Docket No. R2001-1

USPS LR-J-83

PRC Version/Development of ECR Mail Processing Saturation Savings

Table of Contents

I. *Introduction*

This library reference provides the supporting documentation and analyses used to estimate Standard Mail Enhanced Carrier Route (ECR) and NECR mail processing saturation savings using the Postal Rate Commission's (PRC) costing methodology. This is a Category 5 library reference. The estimation process described in this library reference is similar to the one described in USPS-LR-J-59. The difference between the two is that the estimation process presented in this library reference uses the PRC cost methodology to estimate mail processing volume-variable costs for clerks and mailhandlers, while USPS-LR-J-59 uses the Postal Service's methodology.

This library reference relies on other witnesses' library references in this docket. The following sources are used:

- USPS-LR-J-74 for BY00 PRC CRA costs
- USPS-LR-J-82 for the PRC's volume-variable cost methodology
- USPS-LR-J-10 for the IOCS Data Set
- USPS-LR-J-112 for volumes by shape and weight increment
- USPS-LR-J-88 for nontransportation unit cost avoidance
- USPS-LR-J-80 for base year and test year cost factors

II. Organization

The main results are presented in the Excel workbook 'LR83ECR PRC.xls' in the spreadsheet 'Table 1.' Table 1, which is presented below, is analogous to Table 1 in USPS-LR-J-59. The methodology used to develop the estimates in Table 1 is the same as that used in USPS-LR-J-59, with the exception the PRC cost methodology is used in its development. To develop the model for this library reference, the following data were used: base year and test year cost data

(from USPS-LR-J-80), base year IOCS (USPS-LR-J-10) and volume data (USPS-LR-J-112), and the test year estimate for nontransportation unit cost avoidance (USPS-LR-J-88). Data sources are referenced in each spreadsheet in LR83ECR PRC.xls. The underlying mail processing volume-variable costs for clerks and mailhandlers are estimated using the PRC's cost distribution methodology. The programs used to estimate these costs are described in Appendix A: Program Documentation. Appendix B contains the Fortran and Sort/Merge program listings. Electronic copies of all Excel workbooks and text files with the Fortran and Sort/Merge programs are provided on the accompanying CD-ROM.

Table 1: FY00 Dropship Adjusted Unit Costs (cents)
PRC Version

ECR Rate Category	ECR (in cents)
Auto Basic Letters	1.625
Basic Letters	2.987
High Density Letters	0.684
Saturation Letters	0.684
Basic Flats	3.374
Basic Parcels	249.513
Total Basic Nonletters	3.649
High Density/Saturation Flats	1.168
High Density/Saturation Parcels	86.997
Total High Density/Saturation Nonletters	1.189

Appendix A: Program Documentation

A. Computer Hardware and Software

The IOCS data processing is performed on a Data General AViiON minicomputer with four Pentium Pro microprocessors and one gigabyte of RAM, running the DGUX version of UNIX operating system. Source programs with an ".f" file extension are FORTRAN programs and programs with a ".sm" file extension are SORT/MERGE programs. The remaining processing is performed in Excel workbooks (file extension '.xls') on PCs running the Windows NT 4 and Windows 2000 operating systems and Microsoft Office.

B. Preparation of the IOCS Data

The following programs are used to extract, code, and process the 2000 IOCS data set in preparation for the Postal Rate Commission (PRC) volume-variable cost distribution for clerks and mailhandler mail processing costs.

Program: ***cadoc00prc.f*** - *Divides IOCS clerk/mailhandler tallies by office group* (MODS 1&2, BMCs, Non-MODS) and assigns the tallies to cost pools

Input: **FY00 IOCS Data** (USPS-LR-J-10)
mods_usps.00 - List of MODS 1&2 finance numbers used to identify MODS 1&2 offices (USPS-LR-J-82)

Output: **mods12_mp00prc.dat** – IOCS mail processing tallies for MODS 1&2 offices
mods12_aw00prc.dat – IOCS administrative and window service tallies for MODS 1&2 offices
bmcs_mp00prc.dat – IOCS mail processing tallies for BMCs
bmcs_aw00prc.dat – IOCS administrative and window service tallies for BMCs
nonmods_mp00prc.dat – IOCS mail processing tallies for Non-MODS offices
nonmods_aw00prc.dat – IOCS administrative and window service tallies for Non-MODS offices
nonmods_op88prc.dat – IOCS expedited delivery tallies for Non-MODS offices

C. Postal Rate Commission (PRC) Method Volume-Variable Cost Estimates – Clerks and Mailhandlers, Mail Processing

These volume-variable cost distribution FORTRAN programs replicate the function of the mail processing cost distribution SAS programs documented in USPS-LR-J-82, which use the Postal Rate Commission's (PRC) cost distribution methodology. The FORTRAN programs described below divide the cost estimates by Standard ECR rate category, cost pool, and shape.

Program: ***modsproc00prc_wgt2.f*** *– Estimates mail processing volume-variable costs for* MODS 1&2 offices by activity code, cost pool, and weight increment using the PRC methodology

Input: **mods12_mp00prc.dat** – IOCS mail processing tallies for MODS 1&2 offices
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
mixclass.intl – List of class specific mixed mail activity codes
mxmail.intl.dat – Maps the direct activity codes to their respective class specific mixed mail activity codes
costpools.00.new - List of MODS 1&2 cost pools and cost pool dollars for MODS 1&2 offices (USPS-LR-J-82)

Output: **mods00prc.data** – Estimated PRC mail processing volume-variable costs by cost pool, activity code, and weight increment for MODS 1&2 offices

Program: **sumclass_mod_ecr.f** - Rolls up the output from modsproc00prc_wgt2.f from activity code to Standard Mail ECR and NECR rate category by cost pool and shape

Input: **mods00prc.data** – Estimated PRC mail processing volume-variable costs by cost pool, activity code, and weight increment for MODS 1&2 *offices*
costpools.00.new - List of cost pools for MODS 1&2 offices (USPS-LR-J-82)
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
classes_ecr.old - List of old CRA subclasses
classmap_ecr.old - Maps IOCS activity codes to the appropriate CRA subclass

Output: **mod00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard ECR rate category, cost pool, and shape

Workbook: **mod00 all PRC.xls** – Summarizes the PRC mail processing volume-variable cost estimates for MODS 1&2 offices by cost pool and shape for Standard Mail ECR and NECR rate categories (Basic, WSS/WSH, and Automation)

Input: **mod00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard ECR rate category, cost pool, and shape
FY00 PRC mail processing volume-variable costs – Parcel Post mail processing volume-variable costs for clerks/mailhandlers by cost pool using the PRC method cost distribution (USPS-LR-J-82)

Program: **bmcproc00prc_wgt2.f** – Estimates PRC mail processing volume-variable costs for BMCs by activity code, cost pool, and weight increment

Input: **bmcs_mp00prc.dat** – IOCS mail processing tallies for BMCs
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
mixclass.intl – List of class specific mixed mail activity codes
mxmail.intl.dat – Maps the direct activity codes to their respective class specific mixed mail activity codes
costpools.00.bmc.619 – List of cost pools for BMCs (USPS-LR-J-82)

Output: **bmc00prc.data** – Estimated PRC mail processing volume-variable costs by cost pool, activity code, and weight increment for BMCs

Program: **sumclass_bmc_ecr.f** - Rolls up the output from bmcproc00prc_wgt2.f from activity code to Standard Mail ECR and NECR rate categories by cost pool and shape

Input: **bmc00prc.data** – Estimated PRC mail processing volume-variable costs by cost pool, activity code, and weight increment for BMCs
costpools.00.bmc.619 – List of cost pools for BMCs (USPS-LR-J-82)
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
classes_ecr.old - List of old CRA subclasses
classmap_ecr.old - Maps IOCS activity codes to the appropriate CRA subclasses

Output: **bmc00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Workbook: **bmc00 all PRC.xls** – Summarizes the PRC mail processing volume-variable cost estimates for BMCs by cost pool and shape for Standard Mail ECR and NECR rate categories (Basic, ECR WSS/WSH, and Automation)

Input: **bmc00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape
FY00 PRC mail processing volume-variable costs – Parcel Post mail processing volume-variable costs for clerks/mailhandlers by cost pool using the PRC method cost distribution (USPS-LR-J-82)

Program: **nmodproc00prc_wgt.f** – Estimates PRC mail processing volume-variable costs for Non-MODS offices by activity code, cost pool, and weight increment

Input: **nonmods_mp00prc.dat** – IOCS mail processing tallies for Non-MODS offices
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
mixclass.intl – List of class specific mixed mail activity codes
mxmail.intl.dat – Maps the direct activity codes to their respective class specific mixed mail activity codes
costpools.00.nmod.619 – List of cost pools for Non-MODS offices (USPS-LR-J-82)

Output: ***nmod00prc.data*** – *Estimated PRC mail processing volume-variable* costs by cost pool, activity code, and weight increment

Program: **sumclass_nmod_ecr.f** - Rolls up the output from nmodproc00prc_wgt.f from activity code to Standard Mail ECR and NECR rate categories by cost pool and shape

Input: **nmod00prc.data** – Estimated PRC mail processing volume-variable costs by cost pool, activity code, and weight increment
costpools.00.nmod.619 – List of cost pools for Non-MODS offices (USPS-LR-J-82)
activity00.ecr.cra2 – List of the direct and class specific mixed activity codes
classes_ecr.old - List of old CRA subclasses
classmap_ecr.old - Maps IOCS activity codes to the appropriate CRA subclasses

Output: **nmod00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Workbook: **nmod00 all PRC.xls** – Summarizes the PRC mail processing volume-variable cost estimates for Non-MODS offices by cost pool and shape for Standard Mail ECR and NECR rate categories (Basic, ECR WSS/WSH, and Automation)

Input: **nmod00cra_prc_ecr.csv** – Estimated PRC mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape
FY00 PRC mail processing volume-variable costs – Parcel Post mail processing volume-variable costs for clerks/mailhandlers by cost pool using the PRC method cost distribution (USPS-LR-J-82)

Workbook: **LR83Output.xls** – Summarizes the Standard Mail ECR and NECR costs by rate category, shape, and cost pool for each office type. Has a separate worksheet for each office type; MODS 1&2 offices, BMCs, and Non-MODS offices. Adjusts the FOTRAN PRC cost estimations to match the SAS cost estimations

Input: **mod00 all PRC.xls**
bmcs00 all PRC.xls
nmod00 all PRC.xls

Appendix B: Program Lists

Section I: Preparation of IOCS Data

(Program: cadoc00prc.f)

```
      Program cadoc00prc

c     Purpose:  Divides IOCS Clerk and Mailhandler tallies by office group (MODS 1&2, BMCs, Non-MODS),
c               activity (mail processing, administrative, window service), and cost pool.
c               PRC Methodology

      IMPLICIT NONE

      integer*4    nfin, npool, npool2

      parameter    (nfin=568)   !  # of MODS finance numbers
      parameter    (npool=57)   !  # of cost pools
      parameter    (npool2=75)  !  # of cost pools including BMC and Non-MODS breaks

      include 'iocs2000.h'

      integer*4    pool         !  function to assign cost pool group
      integer*4    rog(nfin)
      integer*4    ct1, ct2, ct3, ctaw1, ctmp1, ctaw2
      integer*4    ctmp2, ctaw3, ctmp3, ctkeep
      integer*4    if262, if260, if257, iw, actv, if244, ct_good
      integer*4    ct_inv, ct_inva
      integer*4    ldc, tally_pool(npool), bmcgrp2
      integer*4    modgrp, nmodgrp, if9805, if9806, if245
      integer*4    keep, hand, bmcgrp, ier, ct, i1, i2, i3, i4, i5
      integer*4    costpool, searchc, i, ct_brk_bmc, ct_brk_nmod
      integer*4    ct_reg_006, ct_reg_after
      integer*4    ct_reg_ldc, ct_reg_60, ct_reg_final
      integer*4    ct_reg_f9606, ct_reg_rpw

      real*8       ct_reg_before
      real*8       rf9250, wgt, dlrs, mp_nmod
      real*8       mp_mod, mp_bmc
      real*8       cost_pool(npool), brk_bmc, brk_nmod
      real*8       cost_win, cost_adm, cost_inq, adm_bmc, adm_non
      real*8       win_bmc, win_non, cost_intl, cost_out
      real*8       cost_mod, cost_bmc, cost_nmod
      real*8       ovh6522_bmc, ovh6522_nmod, ovhfact_nmod

      character*6  fin(nfin)
      character*3  type
      character*4  cf244
      character*1 codes(27)/'A','B','C','D','E','F','G','H','I','J','K',
     &   'L','M','N','O','P','Q','R','S','T','U','V',
     &   'W','X','Y','Z',' '/

c     List of MODS 1&2 offices by finance number

      open(10,file='mods_usps.00')

11    format(a6)
      do i=1,nfin
         read(10,11) fin(i)
         rog(i) = 1
      end do

      print*, 'Read in fin #s '

      close(10)

      open(25,file='iocsdata.2000.new',recl=1167) !  FY2000 IOCS data set
21    format(a1167)
31    format(a1167,f15.5,i2,i2,i3,i5)
      open(30,file='mods12_mp00prc.dat',recl=1200) !  MODS 1&2 mail processing IOCS tallies
      open(35,file='mods12_aw00prc.dat',recl=1200) !  MODS 1&2 admin/window service IOCS tallies
      open(40,file='bmcs_mp00prc.dat',recl=1200) !  BMCs mail processing IOCS tallies
      open(45,file='bmcs_aw00prc.dat',recl=1200) !  BMCs admin/window service IOCS tallies
      open(50,file='nonmods_mp00prc.dat',recl=1200) !  Non-MODS mail processing IOCS tallies
      open(55,file='nonmods_aw00prc.dat',recl=1200) !  Non-MODS admwin/window service IOCS tallies
      open(56,file='nonmods_op88prc.dat',recl=1200) !  Non-MODS expedited delivery tallies

      ier=0
      ct=0
      i1=0
      i2=0
      i3=0
      i4=0
      i5=0
      ct_good = 0
      ct_inv = 0
```

```
      ct_inva = 0
      mp_nmod = 0.0
      mp_mod = 0.0
      mp_bmc = 0.0
      ct1 = 0
      ct2 = 0
      ct3 = 0
      ctaw1 = 0
      ctmp1 = 0
      ctaw2 = 0
      ctmp2 = 0
      ctaw3 = 0
      ctmp3 = 0
      ctkeep = 0
      cost_win = 0.0
      cost_adm = 0.0
      cost_inq = 0.0
      cost_intl = 0.0
      cost_out = 0.0
      adm_bmc = 0.0
      adm_non = 0.0
      win_bmc = 0.0
      win_non = 0.0
      ct_brk_bmc = 0
      ct_brk_nmod = 0
      brk_bmc = 0.0
      brk_nmod = 0.0
      cost_mod = 0.0
      cost_bmc = 0.0
      cost_nmod = 0.0
      ovh6522_bmc = 0.0
      ovh6522_nmod = 0.0
      ovhfact_nmod = 0.0
      ct_reg_before = 0.
      ct_reg_006 = 0
      ct_reg_after = 0
      ct_reg_ldc = 0
      ct_reg_60 = 0
      ct_reg_final = 0
      ct_reg_f9606 = 0
      ct_reg_rpw = 0

99    do while (ier.eq.0)

         keep=0
         hand=0
         type='   '
         modgrp=0
         bmcgrp=0
         nmodgrp=0

         read(25,21,iostat=ier,end=100) rec !  Read in IOCS tallies

         iw = 1

         read(f260,'(i2)') if260
         read(f262,'(i4)') if262
         read(f257,'(i2)') if257
         read(f244,'(i4)') if244
         read(f9250,'(f10.0)') rf9250
         read(f9805,'(i4)') if9805
         read(f9806,'(i4)') if9806

         cf244 = f244

         ct=ct+1

c     Separate out Clerk/Mailhander IOCS tallies

c     Identify MODS 1&2 office tallies
         il=searchc(fin,nfin,f2)

c     Identify Clerk/Mailhandler tallies by roster designation (F257)
         if ((if257.eq.11).or.(if257.eq.12).or.(if257.eq.31).or.(if257.eq.32)
     +       .or.(if257.eq.41).or.(if257.eq.42).or.(if257.eq.61).or.(if257.eq.62)
     +       .or.(if257.eq.81).or.(if257.eq.82)) then
            keep=1
         end if

c     Exclude tallies with a tally dollar weight (F9250) of zero
```

```
      if (rf9250.le.0.0) then
         keep=0
      end if

c     Exclude CAG K offices
      if (f264.eq.'K') then
         keep=0
      end if

      if (keep.eq.1) then
         ctkeep=ctkeep+1
      end if

      wgt=rf9250/100000.

c     Assign office type
      if (keep.eq.1) then
         f1(1:1) = '4'        !  Function 4 offices
         if (f2.eq.'      ') then
            f1 = '  '
         end if
         if ((f263.eq.'333333').or.(f263.eq.'444444').or.(f263.eq.'666666')) then
            f1(1:1) = '1'     !  Function 1 offices
         end if
         if (f263.eq.'666666') then !  BMCs
            type = 'bmc'
            ct1 = ct1 + 1
            cost_bmc = cost_bmc + wgt
         else if (i1.gt.0) then !  MODS 1&2 offices
            if ((rog(i1).eq.1).or.(rog(i1).eq.2)) then
               type = 'mod'
               ct2 = ct2 + 1
               cost_mod = cost_mod + wgt
            end if
         else                  !  Non-MODS offices
            type = 'non'
            ct3 = ct3 + 1
            if (if260.ne.88) then
               cost_nmod = cost_nmod + wgt
            end if
         end if

c     Cost pool assignment for MODS 1&2 offices

         if (type.eq.'mod') then

            modgrp=pool(f114) !  Subroutine that assigns cost pool based on MODs code (F114)

            if (modgrp.eq.100) then
               ct_inv = ct_inv + 1
            else
               ct_good = ct_good + 1
            end if

c     Use various IOCS fields to assign cost pool to those tallies with an invalid MODs code

            if (modgrp.eq.100) then
               if (f1(1:1).eq.'1') then !  Function 1 offices
                  if ((f128.eq.'A').and.(f9211.eq.'A')) then
                     modgrp=12 !  manl
                  else if ((f128.eq.'A').and.(f9211.eq.'B')) then
                     modgrp=11 ! manf
                  else if ((f128.eq.'A').and.(f9211.eq.'C')) then
                     modgrp=13 ! manp
                  else if ((f128.eq.'A').and.(f9211.eq.'D')) then
                     modgrp=17 !  1CancMPP
                  else if ((f128.eq.'A').and.(f9211.eq.'E')) then
                     modgrp=16 !  1Bulk pr
                  else if ((f128.eq.'A').and.(f9211.eq.'F')) then
                     modgrp=19 !  1OpPref
                  else if ((f128.eq.'A').and.(f9211.eq.'G')) then
                     modgrp=21 !  1Pouching
                  else if ((f128.eq.'A').and.(f9211.eq.'H')) then
                     modgrp=20 !  1Platform
                  else if (f128.eq.'B') then
                     modgrp=3 !  OCR
                  else if ((f128.ge.'C').and.(f128.le.'E')) then
                     modgrp=1 !  BCS
                  else if (f128.eq.'F') then
                     modgrp=6 !  LSM
```

```
else if ((f128.ge.'G').and.(f128.le.'H')) then
   modgrp=17 !  1CancMPP
else if (f128.eq.'I') then
   modgrp=10 !  1SackS_m
else if (f128.eq.'J') then
   modgrp=7 !  mecparc
else if (f128.eq.'K') then
   modgrp=4 !  FSM
else if (f128.eq.'L') then
   modgrp=8 !  spbs Oth
else if (f128.eq.'M') then
   modgrp=10 !  1Sacks_m
else if (f128.eq.'N') then
   modgrp=22 !  1Sacks_h
else if (f128.eq.'O') then
   modgrp=19 !  1OPref
else if (f128.eq.'P') then
   modgrp=23 !  1Scan
else if (f128.eq.'Q') then
   modgrp=21 !  1Pouching
else if (f128.eq.'R') then
   modgrp=17 !  1CancMPP
else if (f128.eq.'S') then
   modgrp=15 !  LDC 15
else if ((f128.eq.'T').and.((f9212.ge.'A').and.(f9212.le.'D'))) then !
   modgrp=20 !  1Platform
else if (if260.eq.7) then
   modgrp=42 !  LDC 79
else if (if260.eq.8) then
   modgrp=20 !  1Platform
else if ((f118.eq.'A').or.(f118.eq.'C').or.
+       (f118.eq.'E').or.(f118.eq.'F').or.
+       (f118.eq.'I').or.(f118.eq.'K')) then
   modgrp=17 !  1CancMPP
else if ((f118.eq.'B').or.(f118.eq.'D').or.
+       (f118.eq.'H').or.(f118.eq.'J')) then
   modgrp=19 !  1OPref
else if (f118.eq.'G') then
   modgrp=28 !  Rewrap
else if (((f119.ge.'A').and.(f119.le.'F')).and.
+       (f122.eq.' ').and.(f9602.eq.'A')) then
   modgrp=13 !  manp
else if (((f119.ge.'A').and.(f119.le.'F')).and.
+       (f122.eq.' ').and.(f9602.eq.'C')) then
   modgrp=22 !  1Sacks_h
else if (((f119.ge.'A').and.(f119.le.'F')).and.
+       (f122.eq.' ').and.(f9602.eq.'D')) then
   modgrp=13 !  manp
else if (((f119.ge.'A').and.(f119.le.'G')).and.
+       (f122.eq.' ').and.((f128.eq.'A').and.(f9211.eq.'I'))) then !
   modgrp=30 !  1Misc
else if (((f119.ge.'A').and.(f119.le.'G')).and.
+       (f122.eq.' ').and.(f9602.eq.'B')) then
   modgrp=21 !  1Pouching
else if ((f122.ge.'A').and.(f122.le.'F')) then
   modgrp=21 !  1Pouching
else if ((f122.ge.'I').and.(f122.le.'L')) then
   modgrp=21 !  1Pouching
else if (f122.eq.'G') then
   modgrp=30 !  1Misc
else if (f122.eq.'M') then
   modgrp=30 !  1Misc
else if (f122.eq.'H') then
   modgrp=29 !  1EEqmt
else if (if260.eq.0) then
   modgrp=24 !  Bus Reply
else if (if260.eq.6) then
   modgrp=31 !  1Support
else if (if260.eq.9) then
   modgrp=95 !  1Window
else if (if260.eq.10) then
   modgrp=98 !  2Adm
else if (if260.eq.14) then
   modgrp=41 !  LDC 49
else if (if260.eq.17) then
   modgrp=97 !  2Adm inq
else if (if260.eq.18) then
   modgrp=27 !  Registry
else if (if260.eq.19) then
   modgrp=26 !  Mailgram
```

```
                  else if (if260.eq.20) then
                     modgrp=31 !  1Support
                  else if (if260.eq.21) then
                     modgrp=38 !  LDC48 Oth
                  else if (if260.eq.22) then
                     modgrp=25 !  Express
                  else if (if260.eq.23) then
                     modgrp=38 !  LDC48 Oth
                  else if ((if260.ge.24).and.(if260.le.26)) then
                     modgrp=95 !  Window
                  else
                     modgrp=30 !  1Misc
                  end if
               end if

               if (f1(1:1).eq.'4') then !  Function 4 offices
                  modgrp = 98 !  2Adm
                  if ((f128.ge.'B').and.(f128.le.'E')) then
                     modgrp=33 !  LDC 41
                  else if ((f128.eq.'F').or.(f128.eq.'K')) then
                     modgrp=34 !  LDC 42
                  else if (if260.eq.0) then
                     modgrp=40 !  LDC48 Sp Serv
                  else if (if260.eq.6) then
                     modgrp=40 !  LDC48 Sp Serv
                  else if ((if260.ge.11).and.(if260.le.13).or.
     +                (if260.eq.20)) then
                     modgrp=36 !  LDC 44
                  else if ((if260.eq.9).or.((if260.ge.24).and.
     +                (if260.le.26))) then
                     modgrp=95 !  Window
                  else if (if260.eq.10) then
                     modgrp=98 !  2Adm
                  else if (if260.eq.14) then
                     modgrp=41 !  LDC 49
                  else if (if260.eq.17) then
                     modgrp=97 !  2Adm inq
                  else if (if260.eq.18) then
                     modgrp=40 !  LDC48 Sp Serv
                  else if (if260.eq.19) then
                     modgrp=26 !  Mailgram
                  else if (if260.eq.21) then
                     modgrp=40 !  LDC 48 Sp Serv
                  else if (if260.eq.22) then
                     modgrp=37 !  LDC48 Exp
                  else if (if260.eq.23) then
                     modgrp=40 !  LDC48 Sp Serv
                  else
                     modgrp=35 !  LDC43
                  end if
               end if
            end if

            if (modgrp.eq.100) then
               ct_inva = ct_inva + 1
               print*, 'Cost pool not assigned ', f114
            end if

c     Assigns LDC to cost pools

            ldc = 0

            if ((modgrp.ge.1).and.(modgrp.le.3)) then
               ldc = 11
            else if ((modgrp.ge.4).and.(modgrp.le.6)) then
               ldc = 12
            else if ((modgrp.ge.7).and.(modgrp.le.10)) then
               ldc = 13
            else if ((modgrp.ge.11).and.(modgrp.le.14)) then
               ldc = 14
            else if (modgrp.eq.15) then
               ldc = 15
            else if ((modgrp.ge.16).and.(modgrp.le.23)) then
               ldc = 17
            else if ((modgrp.ge.24).and.(modgrp.le.31)) then
               ldc = 18
            else if (modgrp.eq.33) then
               ldc = 41
            else if (modgrp.eq.34) then
               ldc = 42
```

```
            else if (modgrp.eq.35) then
               ldc = 43
            else if (modgrp.eq.36) then
               ldc = 44
            else if ((modgrp.ge.37).and.(modgrp.le.40)) then
               ldc = 48
            else if (modgrp.eq.41) then
               ldc = 49
            else if (modgrp.eq.42) then
               ldc = 79
            else
               ldc = 0
            end if


c     MODS-based encirclement rule

c     For domestic mail
c       PRC version uses old-style MODS encirclement

            if (((if9806.ge.10).and.(if9806.le.300)).and.(f9214.eq.' ')) then

               if (if9806.eq.60) then
                  actv = if9806
               else if (if9806.eq.190) then
                  if ((f9606.eq.'A').or.(f9606.eq.'B')) then
                     actv = if9806
                  else if ((f9806.eq.'0190').and.(f9805(2:4).eq.'510')) then
                     actv = if9806
                  else if (((f246.eq.'   ').or.(f247.eq.'   ').or.
     &                 (f248.eq.'   ').or.(f249.eq.'   ')).and.
     &                 ((modgrp.eq.42).or.(modgrp.eq.95).or.(modgrp.eq.35).or. !  LD79, Window, LD43
     &                 (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.39).or. !  LD48 Oth, LD48 SSV, LD48 Adm
     &                 (modgrp.ge.97).or.(modgrp.eq.30).or.(modgrp.eq.31))) then !  2Adm, 1Misc, 1Support
                     actv = if9806
                  else
                     print *,'possible encirclement error pool=',modgrp
                     actv = 9999
                  end if
               else if ((if9806.eq.300).and.((f9606.eq.'C').or.(f9632.eq.'1'))) then
                  actv = if9806
               else if ((f246.eq.'   ').and.(f247.eq.'   ').and.
     &                 (f248.eq.'   ').and.(f249.eq.'   ')) then
                  actv = if9805
                  if (f9806.eq.'0210') then
                     actv = if9806
                  else if ((f9806.eq.'0090').and.
     &                 ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &                 (modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
     &                 (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
     &                 (modgrp.eq.22).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  1Sacks_h, 1Misc, 1Support
     &                 (modgrp.eq.35).or.(modgrp.eq.39).or.(modgrp.ge.97))) then !  LD43, LD48 Adm, 2Adm
                     actv = if9806
                  else if (((f9806.eq.'0030').or.(f9806.eq.'0070').or.(f9806.eq.'0080')).and.
     &                 ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &                 (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                 (modgrp.eq.95).or. !  Window
     &                 (modgrp.eq.97).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  2Adm Inq, 1Misc, 1Support
     &                 (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                     read(f245,'(i3)') if245
                     actv = if9806
                  else if ((f9806.eq.'0010').and.
     &                 ((modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                 (modgrp.eq.95).or. !  Window
     &                 (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.37).or. !  LD48 Oth, LD48 SSV, LD48 Exp
     &                 (modgrp.eq.25).or.(modgrp.eq.30).or.(modgrp.eq.31))) then !  Express, 1Misc, 1Support
                     actv = if9806
                  else if ((f9806.eq.'0050').and.
     &                 ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &                 (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                 (modgrp.eq.95).or. !  Window
     &                 (modgrp.eq.30).or.(modgrp.eq.31).or. !  1Misc, 1Support
     &                 (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                     actv = if9806
                  else if ((f9806.eq.'0020').and.
     &                 ((modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
     &                 (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
     &                 (modgrp.eq.22).or. !  1SacksS_h
     &                 (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                 (modgrp.eq.95))) then !  Window
```

```
                  actv = if9806
               end if
            else if ((f246.gt.'001').or.(f247.gt.'001').or.
                  (f248.gt.'001').or.(f249.gt.'001')) then
               actv = if9805
               if (((f9806.eq.'0030').or.(f9806.eq.'0070').or.
     &            (f9806.eq.'0080')).and.
     &            ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &            (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &            (modgrp.eq.95).or. !  Window
     &            (modgrp.eq.97).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  2Adm Inq, 1Misc, 1Support
     &            (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  read(f245,'(i3)') if245
                  actv = if9806
               else if ((f9806.eq.'0010').and.
     &                ((modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                (modgrp.eq.95).or. !  Window
     &                (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.37).or. !  LD48 Oth, LD48 SSV, LD48 Exp
     &                (modgrp.eq.25).or.(modgrp.eq.30).or.(modgrp.eq.31))) then !  Express, 1Misc, 1Support
                  actv = if9806
               else if ((f9806.eq.'0050').and.
     &                ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &                (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                (modgrp.eq.95).or. !  Window
     &                (modgrp.eq.30).or.(modgrp.eq.31).or. !  1Misc, 1Support
     &                (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  actv = if9806
               else if ((f9806.eq.'0020').and.
     &                ((modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
     &                (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
     &                (modgrp.eq.22).or. !  1SackS_h
     &                (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
     &                (modgrp.eq.95))) then !  Window
                  actv = if9806
               end if
            end if
         else
            actv = if9806
         end if

c        For international mail

         if ((f9806.eq.'0700').and.((f9805(1:2).eq.'54').or.(((f9805(2:2).ge.'6').and.
     &      (f9805(2:2).le.'8')).and.(if9805.le.4950)))) then
            print *,'0700 candidate in ',modgrp,' ',f245,' ',f246
            if (((f245.ge.'001').and.(f245.le.'030')).or.
     &         ((f246.ge.'001').and.(f246.le.'030'))) then
               if ((f245.eq.'006').or.(f246.eq.'006')) then
                  actv = 700
                  print *,'0700 candidate out ',modgrp,' ',f245,' ',f246
               else if (f245.eq.'019') then
                  if ((f9606.eq.'A').or.(f9606.eq.'B')) then
                     actv = 700
                  else if ((f245.eq.'019').and.(f9805(2:4).eq.'510')) then
                     actv = 700
                  else if (((f246.eq.'   ').or.(f247.eq.'   ').or.
     &                  (f248.eq.'   ').or.(f249.eq.'   ')).and.
     &                  ((modgrp.eq.42).or.(modgrp.eq.95).or.(modgrp.eq.35).or. !  LD79, Window, LD43
     &                  (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.39).or. !  LD48 Oth, LD48 SSV, LD48 Adm
     &                  (modgrp.ge.97).or.(modgrp.eq.30).or. !  2Adm, 1Misc
     &                  (modgrp.eq.31))) then !  1Support
                     actv = 700
                  else
                     actv = if9805
                  end if
               else if ((f245.eq.'030').and.((f9606.eq.'C').or.(f9632.eq.'1'))) then
                  actv = 700
               else if ((f246.eq.'   ').and.(f247.eq.'   ').and.
     &                  (f248.eq.'   ').and.(f249.eq.'   ')) then
                  actv = if9805
                  if (f245.eq.'021') then
                     actv = 700
                  else if ((f245.eq.'009').and.
     &                  ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
     &                  (modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
     &                  (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
     &                  (modgrp.eq.22).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  1SackS_h, 1Misc, 1Support
     &                  (modgrp.eq.35).or.(modgrp.eq.39).or.(modgrp.ge.97))) then !  LD43, LD48 Adm, 2Adm
                     actv = 700
                  else if (((f245.eq.'003').or.(f245.eq.'007').or.(f45.eq.'008')).and.
```

```
&                    ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
&                    (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95).or. !  Window
                     (modgrp.eq.97).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  2Adm Inq, 1Misc, 1Support
                     (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  actv = 700
               else if ((f245.eq.'001').and.
&                    ((modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95).or. !  Window
&                    (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.37).or. !  LD48 Oth, LD48 SSV, LD48 Exp
&                    (modgrp.eq.25).or.(modgrp.eq.30).or.(modgrp.eq.31))) then !  Express, 1Misc, 1Support
                  actv = 700
               else if ((f245.eq.'005').and.
&                    ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
&                    (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95).or. !  Window
&                    (modgrp.eq.30).or.(modgrp.eq.31).or. !  1Misc, 1Support
&                    (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  actv = 700
               else if ((f245.eq.'002').and.
&                    ((modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
&                    (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
&                    (modgrp.eq.22).or. !  1SackS_h
&                    (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95))) then !  Window
                  actv = 700
               end if
            else if ((f246.gt.'001').or.(f247.gt.'001').or.
&                (f248.gt.'001').or.(f249.gt.'001')) then
               actv = if9805
               if (((f245.eq.'003').or.(f245.eq.'007').or.(f245.eq.'008')).and.
&                  ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
&                  (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                  (modgrp.eq.95).or. !  Window
&                  (modgrp.eq.97).or.(modgrp.eq.30).or.(modgrp.eq.31).or. !  2Adm Inq, 1Misc, 1Support
&                  (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  actv = 700
               else if ((f245.eq.'001').and.
&                    ((modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95).or. !  Window
&                    (modgrp.eq.38).or.(modgrp.eq.40).or.(modgrp.eq.37).or. !  LD48 Oth, LD48 SSV, LD48 Exp
&                    (modgrp.eq.25).or.(modgrp.eq.30).or.(modgrp.eq.31))) then !  Express, 1Misc, 1Support
                  actv = 700
               else if ((f245.eq.'005').and.
&                    ((modgrp.eq.24).or.(modgrp.eq.38).or.(modgrp.eq.40).or. !  BusReply, LD48 Oth, LD48 SSV
&                    (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95).or. !  Window
&                    (modgrp.eq.30).or.(modgrp.eq.31).or. !  1Misc, 1Support
&                    (modgrp.eq.39).or.(modgrp.ge.97))) then !  LD48 Adm, 2Adm
                  actv = 700
               else if ((f245.eq.'002').and.
&                    ((modgrp.eq.16).or.(modgrp.eq.23).or.(modgrp.eq.21).or. !  1Bulk pr, 1Scan, 1Pouching
&                    (modgrp.eq.17).or.(modgrp.eq.19).or.(modgrp.eq.18).or. !  1CancMPP, 1OpPref, 1OpBulk
&                    (modgrp.eq.22).or. !  1SackS_h
&                    (modgrp.eq.42).or.(modgrp.eq.20).or.(modgrp.eq.35).or. !  LD79, 1Platform, LD43
&                    (modgrp.eq.95))) then !  Window
                  actv = 700
               end if
            end if
         end if
      end if

      if ((if9806.ge.100).and.(if9806.le.120)) then
         actv = if9806
      end if

      if (modgrp.eq.32) then !  International
         if ((f9805(1:2).eq.'54').or.(((f9805(2:2).ge.'6').and.
&           (f9805(2:2).le.'8')).and.(if9805.le.4950))) then
            ct_reg_before = ct_reg_before + wgt
         end if
      end if

      if (modgrp.eq.27) then !  Registry
         if (actv.eq.60) then
            ct_reg_after = ct_reg_after + 1
         end if
      end if

      if (modgrp.eq.27) then !  Registry
```

```
                  if (actv.eq.60) then
                     ct_reg_ldc = ct_reg_ldc + 1
                  end if
               end if

               if (modgrp.eq.27) then !  Registry
                  if (actv.eq.60) then
                     ct_reg_60 = ct_reg_60 + 1
                  end if
               end if


               if (modgrp.eq.27) then !  Registry
                  if (actv.eq.60) then
                     ct_reg_f9606 = ct_reg_f9606 + 1
                  end if
               end if

               if (modgrp.ge.95) then !  Admin/Window Service
                  actv = if9806
                  ldc = 0
               end if

c     Form Intl/MP cost pool

               if ((f2.eq.'054521').or.(f2.eq.'054522').or.(f2.eq.'056793').or.
     &                (f2.eq.'160049').or.(f2.eq.'115855').or.(f2.eq.'350185').or.
     &                (f2.eq.'482267')) then
                  if (((ldc.ge.11).and.(ldc.le.18)).or.
     &                   ((ldc.ge.41).and.(ldc.le.44)).or.
     &                   ((ldc.ge.48).and.(ldc.le.49)).or.
     &                   (ldc.eq.79)) then
                     modgrp = 32 !  Intl MP
                     ldc = 19
                  else
                     modgrp = 96 !  Intl Admin
                  end if
               end if

               if (modgrp.eq.27) then
                  if (actv.eq.60) then
                     ct_reg_final = ct_reg_final + 1
                  end if
               end if

c  Reassign specific actv codes for expanded subclasses

               if ((actv.eq.1060).or.(actv.eq.2060).or.(actv.eq.3060).or.(actv.eq.4060)) then !  1st SP
                  if (f136.eq.'D') then !  Metered
                     if (actv.eq.1060) actv=1068
                     if (actv.eq.2060) actv=2068
                     if (actv.eq.3060) actv=3068
                     if (actv.eq.4060) actv=4068
                  end if
               end if
               if ((actv.eq.1310).or.(actv.eq.2310).or.(actv.eq.3310).or.(actv.eq.4310)) then !  Reg ECR
                  if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
                     if (actv.eq.1310) actv=1311
                     if (actv.eq.2310) actv=2311
                     if (actv.eq.3310) actv=3311
                     if (actv.eq.4310) actv=4311
                  else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
                     if (actv.eq.1310) actv=1312
                     if (actv.eq.2310) actv=2312
                     if (actv.eq.3310) actv=3312
                     if (actv.eq.4310) actv=4312
                  else if (f9617.eq.'1') then !  ECRLOT
                     actv = actv
                  else
                     actv = actv
                  end if
               end if
               if ((actv.eq.1330).or.(actv.eq.2330).or.(actv.eq.3330).or.(actv.eq.4330)) then !  NP ECR
                  if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
                     if (actv.eq.1330) actv=1331
                     if (actv.eq.2330) actv=2331
                     if (actv.eq.3330) actv=3331
                     if (actv.eq.4330) actv=4331
                  else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
                     if (actv.eq.1330) actv=1332
```

```
                  if (actv.eq.2330) actv=2332
                  if (actv.eq.3330) actv=3332
                  if (actv.eq.4330) actv=4332
               else if (f9617.eq.'1') then !  ECRLOT
                  actv = actv
               else
                  actv = actv
               end if
            end if

            if (modgrp.ge.95) then !  Admin/Window Service cost pools
               if (modgrp.eq.95) then !  Window Service
                  cost_win = cost_win + wgt
               else if (modgrp.eq.99) then ! 2Adm out
                  cost_out = cost_out + wgt
                  dlrs=0.0
               else if (modgrp.eq.96) then !  2Adm intl
                  cost_intl = cost_intl + wgt
               else
                  if (modgrp.eq.97) then !  2Adm inq (claims/inquiry)
                     cost_inq = cost_inq + wgt
                  else if (modgrp.eq.98) then !  2Adm
                     cost_adm = cost_adm + wgt
                  end if
               end if
               if (modgrp.eq.95) then !  Window Service
                  costpool=1
               else
                  costpool=2
               end if
               if ((modgrp.ge.95).and.(modgrp.le.99)) then
                  write(35,31) rec, dlrs, modgrp, costpool, iw, actv !  Admin/Window Service tallies
                  ctaw2=ctaw2+1
               end if
            else
               write(30,31) rec, wgt, modgrp, ldc, iw, actv !  Mail Proc tallies
               ctmp2=ctmp2+1
               mp_mod = mp_mod + wgt
               if ((modgrp.gt.0).and.(modgrp.le.npool)) then
                  cost_pool(modgrp) = cost_pool(modgrp) + wgt
                  tally_pool(modgrp) = tally_pool(modgrp) + 1
               else
                  print*, 'Cost pool not assigned cost = ', wgt
               end if
            end if

         end if              ! Tallies at MODS offices

c     Cost pool assignment for BMCs

         if (type.eq.'bmc') then
            if (((if260.ge.0).and.(if260.le.8)).or.
     +          ((if260.ge.11).and.(if260.le.16)).or.
     +          ((if260.ge.18).and.(if260.le.23)).or.
     +          ((if260.ge.27).and.(if260.le.29)).or.(if260.eq.88)) then !  Mail proc operation codes (F260)
               if (if9806.eq.6521) then
                  bmcgrp = 75 ! Z_breaks
               else if ((f128.eq.'I').and.(f121.eq.'N')) then
                  bmcgrp=47  ! SSM
               else if ((f128.eq.'I').and.(f121.eq.'Y')) then
                  bmcgrp=45  ! SSM_Alli
               else if ((f128.eq.'J').and.(f121.eq.'N')) then
                  bmcgrp=46  ! PSM
               else if ((f128.eq.'J').and.(f121.eq.'Y')) then
                  bmcgrp=45  ! PSM_Alli
               else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                (f128.eq.'M')) then
                  bmcgrp=49  ! NMO
               else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                (f9211.eq.'C').and.(f9602.eq.'C')) then
                  bmcgrp=49  ! NMO
               else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                (f9211.eq.'C').and.(f9602.eq.'D')) then
                  bmcgrp=49  ! NMO
               else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                (f128.eq.'L')) then
                  bmcgrp=48  ! SPB
               else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                (f9602.eq.'A')) then
                  bmcgrp=48  ! SPB
```

```
                else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f9602.eq.'B')) then
                     bmcgrp=48  ! SPB
                  else if (((f116.ge.'A').and.(f116.le.'H')).and.
     +                 (f9209.eq.' ')) then
                     bmcgrp=44  ! Platform
                  else if (((f118.ge.'A').and.(f118.le.'K')).and.
     +                 (f9209.eq.' ')) then
                     bmcgrp=45  ! Mail Prep
                  else
                     bmcgrp=45  ! Other
                  end if
                  if ((f128.eq.'I').and.(f121.eq.'N')) then
                     bmcgrp2=47 ! SSM
                  else if ((f128.eq.'I').and.(f121.eq.'Y')) then
                     bmcgrp2=45 ! SSM_Alli
                  else if ((f128.eq.'J').and.(f121.eq.'N')) then
                     bmcgrp2=46 ! PSM
                  else if ((f128.eq.'J').and.(f121.eq.'Y')) then
                     bmcgrp2=45 ! PSM_Alli
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f128.eq.'M')) then
                     bmcgrp2=49 ! NMO
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f9211.eq.'C').and.(f9602.eq.'C')) then
                     bmcgrp2=49 ! NMO
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f9211.eq.'C').and.(f9602.eq.'D')) then
                     bmcgrp2=49 ! NMO
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f128.eq.'L')) then
                     bmcgrp2=48 ! SPB
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f9602.eq.'A')) then
                     bmcgrp2=48 ! SPB
                  else if (((f119.ge.'A').and.(f119.le.'G')).and.
     +                 (f9602.eq.'B')) then
                     bmcgrp2=48 ! SPB
                  else if (((f116.ge.'A').and.(f116.le.'H')).and.
     +                 (f9209.eq.' ')) then
                     bmcgrp2=44 ! Platform
                  else if (((f118.ge.'A').and.(f118.le.'K')).and.
     +                 (f9209.eq.' ')) then
                     bmcgrp2=45 ! Mail Prep
                  else
                     bmcgrp2=45 ! Other
                  end if
               else
                  bmcgrp=0
               end if

               actv = if9806

c     BMC encirclements

               if (((bmcgrp.ge.44).and.(bmcgrp.le.49)).and.(actv.eq.60)) then
                  if ((f9805(2:4).eq.'510').and.((if9805.ge.1000).and.(if9805.le.4950))) then
                     actv = if9805
                  else if ((f9805(2:2).eq.'8').and.
     &                 ((if9805.ge.1000).and.(if9805.le.4950))) then
                     actv = if9805
                  else if ((f9805(1:3).ge.'545').and.(f9805(1:3).le.'548')) then
                     actv = if9805
                  else
                     actv = 60
                  end if
               end if

c  Reassign specific actv codes for expanded subclasses

               if ((actv.eq.1060).or.(actv.eq.2060).or.(actv.eq.3060).or.(actv.eq.4060)) then !  1st SP
                  if (f136.eq.'D') then !  Metered
                     if (actv.eq.1060) actv=1068
                     if (actv.eq.2060) actv=2068
                     if (actv.eq.3060) actv=3068
                     if (actv.eq.4060) actv=4068
                  end if
               end if
               if ((actv.eq.1310).or.(actv.eq.2310).or.(actv.eq.3310).or.(actv.eq.4310)) then !  Reg ECR
                  if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
```

```
            if (actv.eq.1310) actv=1311
            if (actv.eq.2310) actv=2311
            if (actv.eq.3310) actv=3311
            if (actv.eq.4310) actv=4311
         else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
            if (actv.eq.1310) actv=1312
            if (actv.eq.2310) actv=2312
            if (actv.eq.3310) actv=3312
            if (actv.eq.4310) actv=4312
         else if (f9617.eq.'1') then !  ECRLOT
            actv = actv
         else
            actv = actv
         end if
      end if
      if ((actv.eq.1330).or.(actv.eq.2330).or.(actv.eq.3330).or.(actv.eq.4330)) then !  NP ECR
         if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
            if (actv.eq.1330) actv=1331
            if (actv.eq.2330) actv=2331
            if (actv.eq.3330) actv=3331
            if (actv.eq.4330) actv=4331
         else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
            if (actv.eq.1330) actv=1332
            if (actv.eq.2330) actv=2332
            if (actv.eq.3330) actv=3332
            if (actv.eq.4330) actv=4332
         else if (f9617.eq.'1') then !  ECRLOT
            actv = actv
         else
            actv = actv
         end if
      end if

c     Assigns LDC to cost pools

      if ((bmcgrp.ge.46).and.(bmcgrp.le.48)) then
         ldc = 13
      else if (bmcgrp.eq.49) then
         ldc = 14
      else if ((bmcgrp.ge.44).and.(bmcgrp.le.45)) then
         ldc = 17
      else
         ldc = 0
      end if

      if (bmcgrp.eq.0) then
         if ((if260.eq.9).or.((if260.ge.24).and.(if260.le.26))) then !  Admin/Window Service
            costpool = 1
         else
            costpool = 2
         end if
         dlrs=wgt * 850133./849454. !  Convert to cost pool dollars
         write(45,31) rec, dlrs, bmcgrp, costpool, iw, actv !  Admin/Window Service tallies
         ctawl=ctawl+1
         adm_bmc = adm_bmc + wgt
         if ((if260.eq.9).or.((if260.ge.24).and.(if260.le.26))) then !  Window Service
            win_bmc = win_bmc + wgt
         end if
         if (actv.ne.6522) then
            ovh6522_bmc = ovh6522_bmc + (wgt*850133./849454.) !  Overhead factor
         end if
      else             !  Mail proc
         dlrs=wgt
         write(40,31) rec, dlrs, bmcgrp, ldc, iw, actv
         ctmpl=ctmpl+1
         mp_bmc = mp_bmc + dlrs
         if ((bmcgrp.gt.0).and.(bmcgrp.le.npool)) then
            cost_pool(bmcgrp) = cost_pool(bmcgrp) + wgt
            tally_pool(bmcgrp) = tally_pool(bmcgrp) + 1
         end if
         if ((actv.ne.6522).and.(bmcgrp.le.npool)) then
            ovh6522_bmc = ovh6522_bmc + (wgt*850133./849454.) !  Overhead factor
         end if
         if (bmcgrp.eq.75) then !  Breaks
            ct_brk_bmc = ct_brk_bmc + 1
            dlrs = wgt
            brk_bmc = brk_bmc + dlrs
            if (actv.ne.6522) then
               ovh6522_bmc = ovh6522_bmc + (wgt*850133./849454.) !  Overhead factor
            end if
```

```
            end if
          end if
        end if

c     Cost pool assignment for Non-MODS

        if (type.eq.'non') then
          nmodgrp = 0
          actv = if9806
          if (((if260.ge.0).and.(if260.le.8)).or.
     +        ((if260.ge.11).and.(if260.le.16)).or.
     +        ((if260.ge.18).and.(if260.le.23)).or.
     +        ((if260.ge.27).and.(if260.le.29))) then !  Mail Processing operation codes (F260)

            if (f9806.eq.'6521') then
              nmodgrp = 75 !  Breaks
              if (f262.ne.f9806) then
                print*, 'Non MODS breaks, f262 ', f262, ' f9805 ', f9805, ' f9806 ', f9806
              end if
            else if (f128.eq.'A') then !  Manual
              if (f9211.eq.'A') then
                nmodgrp = 54 !  Manual Letters
              else if (f9211.eq.'B') then
                nmodgrp = 53 !  Manual Flats
              else if (f9211.eq.'C') then
                nmodgrp = 55 !  Manual Parcels
              else
                nmodgrp = 50 !  Allied Labor
              end if
            else if ((f128.ge.'B').and.(f128.le.'F')) then
              nmodgrp = 51 !  Automated distribution
            else if ((f128.ge.'G').and.(f128.le.'I')) then
              nmodgrp = 50 !  Allied Labor
            else if ((f128.ge.'J').and.(f128.le.'M')) then
              nmodgrp = 51 !  Automated distribution
            else if ((f128.ge.'N').and.(f128.le.'R')) then
              nmodgrp = 50 !  Allied Labor
            else if (f128.eq.'S') then
              nmodgrp = 51 ! Automated distribution
            else if ((f128.ge.'T').and.(f128.le.'U')) then
              nmodgrp = 50 !  Allied Labor
            else if (((f116.ge.'A').and.(f116.le.'H')).or.
     &                ((f118.ge.'A').and.(f118.le.'K')).or.(f121.eq.'Y')) then
              nmodgrp = 50 !  Allied Labor
            else if (if260.eq.18) then
              nmodgrp = 56 !  Registry
            else if (if260.eq.22) then
              nmodgrp = 52 !  Express
            else
              nmodgrp = 57 !  Misc & support
            end if

c     Non-MODS Encirclement

            if ((nmodgrp.eq.56).or.(nmodgrp.eq.57)) then !  Registry and Misc
              actv = if9806
            else
              actv = if9805
            end if

            if ((nmodgrp.eq.56).and.(actv.ne.60)) actv=if9805 ! Registry


            if (actv.eq.60) then
              if ((f9805(2:4).eq.'510').and.((if9805.ge.1000).and.(if9805.le.4950))) then
                actv = if9805
              else if ((f9805(2:2).eq.'8').and.((if9805.ge.1000).and.(if9805.le.4950))) then
                actv = if9805
              else if ((f9805(1:3).ge.'545').and.(f9805(1:3).le.'548')) then
                actv = if9805
              else
                actv = 60
              end if
            end if
          end if

c   Reassign specific actv codes for expanded subclasses

          if ((actv.eq.1060).or.(actv.eq.2060).or.(actv.eq.3060).or.(actv.eq.4060)) then !  1st SP
            if (f136.eq.'D') then !  Metered
```

```
                  if (actv.eq.1060) actv=1068
                  if (actv.eq.2060) actv=2068
                  if (actv.eq.3060) actv=3068
                  if (actv.eq.4060) actv=4068
               end if
            end if
            if ((actv.eq.1310).or.(actv.eq.2310).or.(actv.eq.3310).or.(actv.eq.4310)) then !  Reg ECR
               if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
                  if (actv.eq.1310) actv=1311
                  if (actv.eq.2310) actv=2311
                  if (actv.eq.3310) actv=3311
                  if (actv.eq.4310) actv=4311
               else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
                  if (actv.eq.1310) actv=1312
                  if (actv.eq.2310) actv=2312
                  if (actv.eq.3310) actv=3312
                  if (actv.eq.4310) actv=4312
               else if (f9617.eq.'1') then !  ECRLOT
                  actv = actv
               else
                  actv = actv
               end if
            end if
            if ((actv.eq.1330).or.(actv.eq.2330).or.(actv.eq.3330).or.(actv.eq.4330)) then !  NP ECR
               if ((f9618.eq.'1').or.(f9619.eq.'1')) then !  WSH/WSS
                  if (actv.eq.1330) actv=1331
                  if (actv.eq.2330) actv=2331
                  if (actv.eq.3330) actv=3331
                  if (actv.eq.4330) actv=4331
               else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
                  if (actv.eq.1330) actv=1332
                  if (actv.eq.2330) actv=2332
                  if (actv.eq.3330) actv=3332
                  if (actv.eq.4330) actv=4332
               else if (f9617.eq.'1') then !  ECRLOT
                  actv = actv
               else
                  actv = actv
               end if
            end if

            if (if260.eq.0) then
               if260=30
            end if

c     Assigns LDC to cost pools

            if (nmodgrp.eq.50) then
               ldc = 17
            else if (nmodgrp.eq.51) then
               ldc = 11
            else if ((nmodgrp.eq.52).or.((nmodgrp.ge.56).and.(nmodgrp.le.57))) then
               ldc = 18
            else if ((nmodgrp.ge.53).and.(nmodgrp.le.55)) then
               ldc = 14
            else
               ldc = 0
            end if

            if (((if260.ge.9).and.(if260.le.10)).or.
     +         (if260.eq.17).or.((if260.ge.24).and.
     +         (if260.le.26))) then !  Admin/Window Service
               if ((if260.eq.9).or.((if260.ge.24).and.(if260.le.26))) then
                  costpool = 1
               else
                  costpool = 2
               end if
               dlrs=wgt * 4833550./4401822. !  Convert to cost pool dollars
               write(55,31) rec, dlrs, if260, costpool, iw, actv ! Admin/Window Service tallies
               adm_non = adm_non + wgt
               ctaw3=ctaw3+1
               if ((if260.eq.9).or.((if260.ge.24).and.(if260.le.26))) then !  Window Service
                  win_non = win_non + wgt
               end if
               if (actv.ne.6522) then
                  ovh6522_nmod = ovh6522_nmod + (wgt*4833550./4401822.) !  Overhead factor
               end if
            else if (if260.ne.88) then !  exclude expedited delivery
               dlrs=wgt
               if (nmodgrp.gt.0) then
```

```
                write(50,31) rec, dlrs, nmodgrp, ldc, iw, actv !  Mail Proc tallies
                ctmp3=ctmp3+1
                mp_nmod = mp_nmod + dlrs
                if (nmodgrp.le.npool) then
                   cost_pool(nmodgrp) = cost_pool(nmodgrp) + wgt
                   tally_pool(nmodgrp) = tally_pool(nmodgrp) + 1
                   if (actv.ne.6522) then
                      ovh6522_nmod = ovh6522_nmod + (wgt*4833550./4401822.) !  Overhead factor
                   end if
                   if ((actv.ne.6521).and.(actv.ne.6522)) then
                      ovhfact_nmod = ovhfact_nmod + (wgt*4833550./4401822.) !  Overhead factor
                   end if
                end if
             else
                print*, 'Pool not assigned f260 = ', if260
             end if
             if (nmodgrp.eq.75) then !  Z Breaks
                ct_brk_nmod = ct_brk_nmod + 1
                dlrs = wgt
                brk_nmod = brk_nmod + dlrs
                if (actv.ne.6522) then
                   ovh6522_nmod = ovh6522_nmod + (wgt*4833550./4401822.) !  Overhead factor
                end if
             end if
          else             !  Op code 88's now part of C/S 3.4
             write(56,21) rec !  Expedited delivery tallies
          end if
       end if
    end if

    end do

100 print*, 'Read exit error ', ier
    print*, 'Total Records ', ct
    print*, 'Number of obs used ', ctkeep
    print*, 'BMC  Total Obs ', ct1, ' Adm/Win ', ctaw1, ' MP ', ctmp1, ' Breaks ', ct_brk_bmc !
    print*, 'MODS Total Obs ', ct2, ' Adm/Win ', ctaw2, ' MP ', ctmp2 !
    print*, 'NMOD Total Obs ', ct3, ' Adm/Win ', ctaw3, ' MP ', ctmp3, ' Breaks ', ct_brk_nmod !
    print*, ' '
    print*, 'Number of MODS 1&2 tallies with a valid MODS code ', ct_good !
    print*, 'Number of MODS 1&2 tallies with an invalid MODS code ', ct_inv !
    print*, 'After residual pool assignment, number of tallies with invalid MODS codes ', ct_inva !
    print*, ' '
    print*, 'Total MODS 1&2 tally dollar weights ', cost_mod
    print*, 'Total BMCs tally dollar weights ', cost_bmc
    print*, 'Total Non-MODS tally dollar weights ', cost_nmod
    print*, ' '
    print*, 'Total MODS mail proc costs ', mp_mod
    print*, 'Total BMC mail proc costs ', mp_bmc
    print*, 'Total Non-MOD mail proc costs ', mp_nmod
    print*, ' '
    print*, 'Total MODS win tally costs = ', cost_win
    print*, 'Total MODS admin tally costs = ', cost_adm
    print*, 'Total MODS admin inq tally costs = ', cost_inq
    print*, 'Total MODS admin intl tally costs = ', cost_intl
    print*, 'Total MODS admin out tally costs = ', cost_out
    print*, 'Total BMCs admin/win tally costs = ', adm_bmc
    print*, 'Total BMC window costs = ', win_bmc
    print*, 'Total BMC break costs = ', brk_bmc
    print*, 'Total NMods admin/win tally costs = ', adm_non
    print*, 'Total NMods window costs = ', win_non
    print*, 'Total NMods break costs = ', brk_nmod
    print*, ' '
    print*, 'OVH6522 factor for BMCs (denominator) ', ovh6522_bmc
    print*, 'OVH6522 factor for Non-MODS (denominator) ', ovh6522_nmod
    print*, 'OVHFACT factor for Non-MODS (denominator) ', ovhfact_nmod
    print*, ' '
    print*, 'Registry checks '
    print*, 'Total Registry tallies before encirclement ', ct_reg_before
    print*, 'Total Registry tallies with F245, F246 = 006 ', ct_reg_006
    print*, 'Total Registry tallies after initial encirclement ', ct_reg_after
    print*, 'Total Registry tallies after LDC to F9805 encirclement ', ct_reg_ldc
    print*, 'Total Registry tallies after cost pool encirclement ', ct_reg_60
    print*, 'Total Registry tallies after F9606 encirclement ', ct_reg_f9606
    print*, 'Total Registry tallies after RPW encirclement ', ct_reg_rpw
    print*, 'Total Registry tallies after all encirclements ', ct_reg_final

    end

c
```

```
c      pool   tcg 7/10/96
c      assigns pool groups to MODS numbers

       function   pool(mod)

       integer*4    pool
       character*3  mod

       pool = 100

c      OCR OPERATIONS
       if ((mod.eq.'046').or.
     &   ((mod.ge.'830').and.(mod.le.'837')).or.
     &   ((mod.ge.'840').and.(mod.le.'847')).or.
     &   ((mod.ge.'850').and.(mod.le.'857')).or.
     &   ((mod.ge.'880').and.(mod.le.'887'))) then
          pool = 3                ! OCR
       else if ((mod.ge.'301').and.(mod.le.'304')) then
          pool = 3                ! Intl/OCR

c      BCS OPERATIONS
       else if ((mod.eq.'047').or.
     &      ((mod.ge.'241').and.(mod.le.'251')).or.
     &      ((mod.ge.'603').and.(mod.le.'604')).or.
     &      ((mod.ge.'860').and.(mod.le.'869')).or.
     &      ((mod.ge.'870').and.(mod.le.'879')).or.
     &      ((mod.ge.'914').and.(mod.le.'917')).or.
     &      ((mod.ge.'970').and.(mod.le.'979'))) then
          pool = 1                ! BCS
       else if (((mod.ge.'311').and.(mod.le.'312')).or.
     &      ((mod.ge.'315').and.(mod.le.'316'))) then
          pool = 1                ! BCS Intl
       else if (((mod.ge.'260').and.(mod.le.'267')).or.
     &      ((mod.ge.'270').and.(mod.le.'279')).or.
     &      ((mod.ge.'280').and.(mod.le.'287')).or.
     &      ((mod.ge.'290').and.(mod.le.'299')).or.
     &      ((mod.ge.'890').and.(mod.le.'899')).or.
     &      ((mod.ge.'908').and.(mod.le.'911')).or.
     &      ((mod.ge.'918').and.(mod.le.'919')).or.
     &      ((mod.ge.'925').and.(mod.le.'926'))) then
          pool = 2                ! BCS/DBCS
       else if ((mod.eq.'309').or.
     &      ((mod.ge.'313').and.(mod.le.'314')).or.
     &      ((mod.ge.'317').and.(mod.le.'319')).or.
     &      ((mod.ge.'356').and.(mod.le.'357'))) then
          pool = 2                ! BCS/DBCS Intl

c      LSM OPERATIONS
       else if (((mod.ge.'080').and.(mod.le.'089')).or.
     &      (mod.eq.'091').or.
     &      ((mod.ge.'093').and.(mod.le.'099'))) then
          pool=6                  ! LSM
       else if ((mod.eq.'090').or.(mod.eq.'092')) then
          pool=6                  ! LSM Intl

c      FSM OPERATIONS
       else if (((mod.ge.'140').and.(mod.le.'148')).or.
     &      (mod.eq.'191').or.
     &      ((mod.ge.'194').and.(mod.le.'197')).or.
     &      ((mod.ge.'331').and.(mod.le.'338')).or.
     &      ((mod.ge.'421').and.(mod.le.'428')).or.
     &      ((mod.ge.'960').and.(mod.le.'967'))) then
          pool=4                  ! FSM 881
       else if ((mod.eq.'192').or.(mod.eq.'193')) then
          pool=4                  ! FSM Intl
       else if (((mod.ge.'441').and.(mod.le.'448')).or.
     &      (mod.eq.'450').or.(mod.eq.'451').or.
     &      ((mod.ge.'461').and.(mod.le.'468'))) then
          pool=5                  ! FSM 1000
       else if (((mod.ge.'305').and.(mod.le.'308')).or.
     &      ((mod.ge.'452').and.(mod.le.'453'))) then
          pool = 5                ! FSM 1000 Intl

c      Mechanized sort-sack outside
       else if ((mod.ge.'238').and.(mod.le.'239')) then
          pool=10                 ! 1Sacks_m
       else if (mod.eq.'349') then
          pool=10                 ! 1Sacks_m Intl

c      MECHANIZED PARCEL SORTER
```

```
      else if (mod.eq.'105') then
         pool=7                  !  Mecparc
      else if ((mod.ge.'107').and.(mod.le.'108')) then
         pool=7                  !  Mecparc Intl

c      SMALL PARCEL BUNDLE SORTER
      else if (((mod.ge.'134').and.(mod.le.'137')).or.
     &      ((mod.ge.'254').and.(mod.le.'257')).or.
     &      ((mod.ge.'434').and.(mod.le.'437'))) then
         pool=8                  !  SPBS Oth
      else if (((mod.ge.'052').and.(mod.le.'054')).or.
     &      ((mod.ge.'056').and.(mod.le.'058')).or.
     &      ((mod.ge.'346').and.(mod.le.'347'))) then
         pool = 8                !  SPBS Oth Intl
      else if (((mod.ge.'138').and.(mod.le.'139')).or.
     &      ((mod.ge.'258').and.(mod.le.'259')).or.
     &      ((mod.ge.'438').and.(mod.le.'439'))) then
         pool=9                  !  SPBS Prio
      else if ((mod.eq.'104').or.(mod.eq.'106')) then
         pool = 9                !  SPBS Prio Intl

c      MANUAL FLAT OPERATIONS
      else if ((mod.eq.'060').or.
     &      ((mod.ge.'069').and.(mod.le.'070')).or.
     &      ((mod.ge.'070').and.(mod.le.'075')).or.
     &      (mod.eq.'170').or.(mod.eq.'175').or.
     &      ((mod.ge.'178').and.(mod.le.'179'))) then
         pool=11                 !  MANF
      else if ((mod.ge.'062').and.(mod.le.'063')) then
         pool=11                 ! MANF Intl

c      MANUAL LETTERS OPERATIONS
      else if (((mod.ge.'029').and.(mod.le.'030')).or.
     &      ((mod.ge.'040').and.(mod.le.'045')).or.
     &      (mod.eq.'150').or.(mod.eq.'160').or.
     &      ((mod.ge.'168').and.(mod.le.'169'))) then
         pool=12                 !  MANL
      else if ((mod.ge.'032').and.(mod.le.'033')) then
         pool=12                 !  MANL Intl

c      MANUAL PARCEL OPERATIONS
      else if ((mod.eq.'100').or.
     &      (mod.eq.'130').or.(mod.eq.'200')) then
         pool = 13               !  MANP
      else if (((mod.ge.'102').and.(mod.le.'103')).or.
     &      ((mod.ge.'202').and.(mod.le.'207'))) then
         pool = 13               !  MANP Intl

c      MANUAL PRIORITY
      else if ((mod.eq.'050').or.(mod.eq.'055')) then
         pool=14                 !  Priority

c      LDC15
      else if (((mod.ge.'381').and.(mod.le.'386')).or.
     &      (mod.eq.'771').or.
     &      ((mod.ge.'774').and.(mod.le.'776')).or.
     &      (mod.eq.'779')) then
         pool=15                 !   LDC 15

c      ALLIED OPERATIONS

c      ACDCS
      else if ((mod.eq.'064').or.
     &      ((mod.ge.'118').and.(mod.le.'119'))) then
         pool=23                 !  1Scan
      else if (mod.eq.'350') then
         pool = 23               !  1Scan Intl

c      Bulk presort
      else if ((mod.ge.'002').and.(mod.le.'009')) then
         pool=16                 !  1Bulk Pr

       Cancellation/mail prep
      else if ((mod.ge.'010').and.(mod.le.'028')) then
         pool=17                 !  1CancMPP

c      opening unit - pref
      else if (((mod.ge.'110').and.(mod.le.'114')).or.
     &      ((mod.ge.'180').and.(mod.le.'184'))) then
         pool=19                 !  1OpPref
```

```
      else if ((mod.ge.'343').and.(mod.le.'344')) then
         pool = 19                 !  1OpPref Intl
      else if ((mod.ge.'358').and.(mod.le.'359')) then
         pool = 19                 !  1OpPref (1Robotic)

c      opening unit - bbm
      else if (((mod.ge.'115').and.(mod.le.'117')).or.
     &       ((mod.ge.'185').and.(mod.le.'189'))) then
         pool=18                   !  1OpBulk

c      pouching
      else if (((mod.ge.'120').and.(mod.le.'129')).or.
     &       ((mod.ge.'208').and.(mod.le.'209'))) then
         pool=21                   !  1Pouching
      else if (mod.eq.'345') then
         pool = 21                 !  1Pouching Intl

c      platform
      else if ((mod.ge.'210').and.(mod.le.'234')) then
         pool=20                   !  1Platform
      else if (((mod.ge.'351').and.(mod.le.'352')).or.
     &       (mod.eq.'454')) then
         pool = 20                 ! 1Platform Intl

c      manual sack sort
      else if ((mod.ge.'235').and.(mod.le.'237')) then
         pool=22                   !  1Sacks_h
      else if (mod.eq.'348') then
         pool = 22                 !  1Sacks_h Intl

c      DAMAGED PARCEL REWRAP
      else if (mod.eq.'109') then
         pool=28                   !  Rewrap
      else if (mod.eq.'574') then
         pool = 28                 !  Rewrap Intl

c      EXPRESS
      else if ((mod.eq.'131').or.(mod.eq.'669').or.(mod.eq.'793')) then
         pool=25                   !  Express
      else if (mod.eq.'575') then
         pool = 25                 !  Express Intl

c      empty equipment
      else if (mod.eq.'549') then
         pool=29                   !  1EEqmt
      else if (mod.eq.'576') then
         pool = 29                 ! 1EEqmt Intl

c      MAILGRAM
      else if (mod.eq.'584') then
         pool=26                   !  Mailgram

c      MAIL PROCESSING SUPPORT
      else if (((mod.ge.'340').and.(mod.le.'341')).or.
     &       (mod.eq.'547').or.(mod.eq.'548').or.
     &       ((mod.ge.'554').and.(mod.le.'555')).or.
     &       (mod.eq.'607').or.
     &       (mod.eq.'612').or.(mod.eq.'620').or.(mod.eq.'630').or.
     &       (mod.eq.'677').or.(mod.eq.'755').or.(mod.eq.'798')) then
         pool=31                   !  1Support

c      MISCELLANEOUS
      else if ((mod.ge.'560').and.(mod.le.'564')) then
         pool=30                   !  1Misc
      else if ((mod.eq.'132').or.
     &       ((mod.ge.'545').and.(mod.le.'546')).or.
     &       (mod.eq.'577').or.(mod.eq.'580').or.(mod.eq.'681')) then
         pool = 30                 !  1Misc Intl

c      BUSINESS REPLY / POSTAGE DUE
      else if (mod.eq.'930') then
         pool=24                   !  Bus Reply
      else if (mod.eq.'573') then
         pool = 24                 !  Bus Reply Intl

c      REGISTRY
      else if ((mod.ge.'585').and.(mod.le.'590')) then
         pool=27                   !  Registry
      else if (mod.eq.'578') then
         pool = 27                 !  Registry Intl
```

```
c      LDC41 AND LDC42
       else if ((mod.eq.'048').or.(mod.eq.'049').or.
            (mod.eq.'252').or.(mod.eq.'253').or.
            ((mod.ge.'361').and.(mod.le.'362')).or.
     &      ((mod.ge.'364').and.(mod.le.'366')).or.
     &      ((mod.ge.'371').and.(mod.le.'378')).or.
     &      ((mod.ge.'411').and.(mod.le.'417')).or.
     &      ((mod.ge.'605').and.(mod.le.'606')).or.
     &      ((mod.ge.'821').and.(mod.le.'829')).or.
     &      ((mod.ge.'905').and.(mod.le.'907')).or.
     &      ((mod.ge.'912').and.(mod.le.'913')).or.
     &      ((mod.ge.'942').and.(mod.le.'943'))) then
          pool=33                  ! LDC 41
       else if (((mod.ge.'400').and.(mod.le.'407')).or.
     &      ((mod.ge.'801').and.(mod.le.'819'))) then
          pool=34                  !  LDC 42

c      MANUAL DISTRIBUTION - STATION/BRANCH (LDC43)
       else if (mod.eq.'240') then
          pool=35                  ! LDC 43

c      STATION/BRANCH - BOX SECTION (LDC44)
       else if (mod.eq.'769') then
          pool=36                  !  LDC 44

c      WINDOW Service
       else if ((mod.eq.'355').or.(mod.eq.'568')) then
          pool=95

c      LDC48
       else if (mod.eq.'583') then
          pool=37                  !  LDC48 Exp
       else if ((mod.eq.'353').or.(mod.eq.'558').or.(mod.eq.'559').or.
     &      (mod.eq.'608').or.(mod.eq.'621').or.
     &      (mod.eq.'631').or.(mod.eq.'678')) then
          pool=39                  !  LDC48 Adm
       else if ((mod.ge.'542').and.(mod.le.'544')) then
          pool=40                  !  LDC48 SSV
       else if ((mod.eq.'741').or.(mod.eq.'742').or.(mod.eq.'794')) then
          pool=38                  !  LDC48 Oth

c      ADDRESS INFO SYSTEM & CENTRAL MAIL MARK-UP
       else if ((mod.eq.'539').or.
     &      ((mod.ge.'791').and.(mod.le.'792')).or.
     &      ((mod.ge.'795').and.(mod.le.'797'))) then
          pool=41                  !  LDC 49

c      MAILING REQUIREMENTS & BUSINESS MAIL ENTRY
       else if ((mod.eq.'001').or.(mod.eq.'550').or.(mod.eq.'660').or.
     &      (mod.eq.'697')) then
          pool=42                  !  LDC 79

c      invalid mods code for mail processing
       else
          pool = 100
       end if

       if (pool.eq.100) then

c      ADMINISTRATION

c      2adm_out
          if ((mod.eq.'342').or.(mod.eq.'354').or.((mod.ge.'455').and.(mod.le.'459')).or.
     &      ((mod.ge.'471').and.(mod.le.'504')).or.((mod.ge.'599').and.(mod.le.'602')).or.
     &      ((mod.ge.'613').and.(mod.le.'614')).or.(mod.eq.'616').or.(mod.eq.'622').or.
     &      (mod.eq.'624').or.(mod.eq.'632').or.((mod.ge.'634').and.(mod.le.'635')).or.
     &      (mod.eq.'641').or.(mod.eq.'655').or.(mod.eq.'671').or.(mod.eq.'676').or.
     &      ((mod.ge.'698').and.(mod.le.'703')).or.((mod.ge.'609').and.(mod.le.'703')).or.
     &      ((mod.ge.'705').and.(mod.le.'740')).or.((mod.ge.'743').and.(mod.le.'754')).or.
     &      ((mod.ge.'757').and.(mod.le.'762')).or.(mod.eq.'768').or.(mod.eq.'770').or.
     &      ((mod.ge.'920').and.(mod.le.'924')).or.((mod.ge.'927').and.(mod.le.'929')).or.
     &      ((mod.ge.'932').and.(mod.le.'937')).or.((mod.ge.'946').and.(mod.le.'953'))) then
            pool = 99
          end if

c      2Adm
          if (((mod.ge.'505').and.(mod.le.'538')).or.((mod.ge.'540').and.(mod.le.'541')).or.
     &      ((mod.ge.'556').and.(mod.le.'557')).or.(mod.eq.'566').or.
     &      ((mod.ge.'569').and.(mod.le.'572')).or.(mod.eq.'579').or.
```

```
     &       ((mod.ge.'581').and.(mod.le.'582')).or.((mod.ge.'591').and.(mod.le.'596')).or.
     &       ((mod.ge.'610').and.(mod.le.'611')).or.(mod.eq.'615').or.(mod.eq.'617').or.
     &       (mod.eq.'623').or.(mod.eq.'633').or.(mod.eq.'636').or.
             ((mod.ge.'642').and.(mod.le.'643')).or.((mod.ge.'648').and.(mod.le.'654')).or.
             ((mod.ge.'656').and.(mod.le.'659')).or.((mod.ge.'661').and.(mod.le.'666')).or.
     &       (mod.eq.'668').or.(mod.eq.'670').or.((mod.ge.'672').and.(mod.le.'675')).or.
     &       ((mod.ge.'679').and.(mod.le.'680')).or.((mod.ge.'682').and.(mod.le.'687')).or.
     &       (mod.eq.'689').or.((mod.ge.'691').and.(mod.le.'696')).or.(mod.eq.'704').or.
     &       ((mod.ge.'763').and.(mod.le.'766')).or.((mod.ge.'772').and.(mod.le.'773')).or.
     &       (mod.eq.'704').or.((mod.ge.'780').and.(mod.le.'789')).or.
     &       ((mod.ge.'900').and.(mod.le.'904')).or.((mod.ge.'958').and.(mod.le.'959')).or.
     &       ((mod.ge.'968').and.(mod.le.'969')).or.((mod.ge.'980').and.(mod.le.'987'))) then
             pool = 98
          end if

c       2Adm inq
          if ((mod.ge.'551').and.(mod.le.'552')) then
             pool = 97
          end if

        end if


        return
        end
```

Section II: Postal Rate Commission Method Volume-Variable Cost Estimates – Clerks and Mailhandlers, Mail Processing

(Programs: modsproc00prc_wgt2.f, sumclass_mod_ecr.f, bmcproc00prc_wgt2.f, sumclass_bmc_ecr.f, nmodproc00prc_wgt.f, sumclass_nmod_ecr.f)

```
      program modsproc00prc_wgt2

c     Purpose:  Computes distributed volume-variable costs (PRC Method) for MODS 1&2 offices
c               Adds additional dimension for various weight categories

      implicit none

      integer*4 nmod, nw, nmod2, nw2
      integer*4 nact, nshp, nmix, nmixcl, nact2
      integer*4 nitem, nshp2, nshp3, ncsi, ncon, begmail

      parameter (nmod = 43)      !  Number of cost pools (includes the LDC 15 Proxy cost pool)
      parameter (nmod2 = 42)     !  Number of distribution cost pools
      parameter (nw =  22)       !  Number of weight increments (including no weight)
      parameter (nw2 = 21)       !  Number of weight increments
      parameter (nact = 261)     !  Number of direct activity codes
      parameter (nshp = 6)       !  Number of shapes
      parameter (nitem = 16)     !  Number of item types
      parameter (nshp2 = 5)      !  Number of shapes (not including other)
      parameter (nshp3 = 4)      !  Number of shapes for PRC mixed-mail keys (ishp = letter, flat, parcel, all)
      parameter (ncon = 10)      !  Number of container types
      parameter (nmix = 20)      ! Combined activity codes - for dist of counted items
      parameter (ncsi = nshp2 + nitem) !  Number of "identified" container types (loose shapes + items)
      parameter (begmail = 18)   ! Set this to the index of the first non-Spec Serv activity
      parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
      parameter (nact2 = 281)    ! Number of activity codes including class-specific mixed-mail

      include 'iocs2000.h'

      real*8     adols(nw,nmod,nact2,nshp) ! Handling direct single piece
      real*8    adist(nw,nmod,nact2,nshp) ! Workspace for distribution of no weight single pieces
      real*8     bdols(nw,nmod,nitem,nact2) ! Handling identical or top-piece item
      real*8    bdist(nw,nmod,nitem,nact2) ! Workspace for distribution of no weight identical/top-piece items
      real*8    cdist(nw,nmod,nitem,nact2) ! Workspace for distribution of matrix D
      real*8     cdols(nw,nmod,nitem,nact2) ! Workspace for distributed costs from matrix D
      real*8     ddols(nmod,nitem) ! Handling mixed/empty item
      real*8     fdols(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8    fdist(nw,nmod,ncon,nact2) ! Workspace for distribution of no weight identical/top-piece containers
      real*8     hkey(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8     gdols(nmod,ncon,ncsi) ! Handling "identified" container
      real*8    gdist(nw,nmod,ncon,nact2) ! G Matrix distributed to activity code
      real*8     hdols(nmod,ncon) ! Handling uncounted/empty container
      real*8    idols(nw,nmod,nact2) ! Special Service directs excluded from PRC keys
      real*8    result(nw,nmod,nact2) ! Array to hold results
      real*8    result2(nw,nmod,nact2) ! Array to hold results for Spec Service not-handling dist
      real*8    resulta(nw,nmod,nact2) ! Array to hold results for matrix A
      real*8    resultb(nw,nmod,nact2) ! Array to hold results for matrix B, C, D
      real*8    resultf(nw,nmod,nact2) ! Array to hold results for matrix F, G, H
      real*8    resultj(nw,nmod,nact2) ! Array to hold distributed J matrix
      real*8    work(nw,nmod,nact2) ! Array to hold distributed mixed class-specific
      real*8    work2(nw,nmod,nact2) ! Array to hold distributed mixed class-specific
      real*8    mixkey(nw,nmod,nact2,nshp2) ! Array to hold PRC allied mixed-mail keys
      real*8    mixkey2(nw,nmod,nact2,nshp2) ! Array to hold PRC allied not-handling keys
      real*8    mixdist(nw,nmod,nact2,nshp2) !  Array to hold PRC distributed mixed-mail
      real*8    mixallied(nmod,nshp3) ! Array to hold PRC allied mixed-mail
      real*8    jdols(nmod,nshp3) ! Not handling
      real*8    counts(ncsi)
      real*8    actshr(nw,nact2), actshr3(nact), actshr4(nw,nact2,nshp3), actwgt(nw2), actshr2(nw,nact2)
      real*8    dlrs, sum, distsum, rf9250, tot_dol, tot_dol2, check, totj,fixtot
      real*8    atot, btot, ctot, dtot, ftot, gtot, htot, itot, jtot
      real*8    pooldols(nmod), iocsdols(nmod), migrate(nmod)
      real*8    prcadj1(nmod), prcadj2(nmod)
      real*8    variable(nmod), fixed(nmod), total(nmod), varprc(nmod)
      real*8    varcost(nw,nmod,nact)
      real*8    novarcst(nw,nmod,nact)
      real*8    distsum48, cost_cntr, cost_unid, check1, check2

      logical flag

      integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
      integer*4 ind, if114, ldc, k, if9806, ishpprc, shapeprc, mixshp, if9805
      integer*4 cnt, mixclass(nact2), class(nact2)
      integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw
      integer*4 ier, ct_cntr, ct_unid, ishp, l
      integer*4 mapcodes(20)
      integer*4 searchc, searchi, modgrp, hand, actv
      integer*4 mixcodes(nmixcl), class_code(nact2)
      integer*4 acodes(nact2), mixcount(nmixcl)
      integer*4 mixmap(nact,nmixcl)
      integer*4 ldc1(nmod)
```

```
      integer*4 if166, if167, weight, ct_nowgt

      character*14 modcodes(nmod)
      character*1 codes(26)/'A','B','C','D','E','F','G','H','I','J','K',
         'L','M','N','O','P','Q','R','S','T','U','V',
     &   'W','X','Y','Z'/

      logical flag2

      atot = 0.0
      btot = 0.0
      ctot = 0.0
      dtot = 0.0
      ftot = 0.0
      gtot = 0.0
      htot = 0.0
      itot = 0.
      jtot = 0.0
      acnt = 0
      bcnt = 0
      ccnt = 0
      dcnt = 0
      fcnt = 0
      gcnt = 0
      hcnt = 0
      jcnt = 0
      cnt = 0
      ier = 0
      fixtot = 0.

      do i = 1, nmod
         prcadj1(i) = 0.
         prcadj2(i) = 0.
         pooldols(i) = 0.0
         variable(i) = 0.0
         iocsdols(i) = 0.0
         fixed(i) = 0.0
         varprc(i) = 0.
         total(i) = 0.
         migrate(i) = 0.
      end do

      do i = 1, 20
         mapcodes(i) = 0
      end do

      do i = 1, nmixcl
         mixcodes(i) = 0
         mixcount(i) = 0
      end do

C     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(i4,i6,i5,i4)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(i), mixclass(i)
      end do
      print *,'read activity map'
      close(20)

c     Map of class specific mixed-mail activity codes
      open(20,file='mixclass.intl')
      do i = 1,nmixcl
         read (20,21) mixcodes(i)
      end do
      print *,'read mixed item code list'
      close(20)

      do i = 1,nact
         do j = 1,nmixcl
            mixmap(i,j) = 0
         end do
      end do

c     Maps class specific mixed-mail activity codes to appropriate direct activity codes
      open(20,file='mxmail.intl.dat')
23    format(20i4)

      do while (ier.eq.0)
         read (20,23,iostat=ier,end=75) mapcodes
```

```
      i = searchi(mixcodes,nmixcl,mapcodes(1))
      if (i.gt.0) then
         flag = .true.
         ind = 1
         do while ((flag).and.(ind.lt.20))
            ind = ind + 1
            if (mapcodes(ind).gt.0) then
               j = searchi(acodes,nact,mapcodes(ind))
               if (j.gt.0) then
                  mixcount(i) = mixcount(i) + 1
                  mixmap(mixcount(i),i) = j
               else
                  print *,' Direct mail code did not map ',mapcodes(ind)
               end if
            else
               flag = .false.
            end if
         end do
      else
         print *,' Mixed mail code did not map ',mapcodes(1)
      end if
   end do
75 print *,' read mixed-mail map with exit code = ',ier
   close(20)

c  Map of cost pool dollars and variabilities by cost pool
   open(20,file='costpools.00.new')
24 format(4x,a14,i5,f9.0,f7.2)
   do i = 1,nmod
      read(20,24) modcodes(i), ldc1(i), pooldols(i), variable(i)
   end do
   close(20)

C  Initialize matrices

   do iw = 1,nw
      do imod = 1,nmod
         do iact = 1,nact
            varcost(iw,imod,iact) = 0.
            novarcst(iw,imod,iact) = 0.
         end do
      end do
   end do
   do ishp = 1, nshp
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               adols(iw,imod,iact,ishp) = 0.0
               adist(iw,imod,iact,ishp) = 0.0
            end do
         end do
      end do
   end do
   do iact = 1, nact2
      do iitem = 1, nitem
         do imod = 1, nmod
            do iw = 1, nw
               bdols(iw,imod,iitem,iact) = 0.
               bdist(iw,imod,iitem,iact) = 0.
            end do
         end do
      end do
   end do
   do iact = 1, nact2
      do iitem = 1, nitem
         do imod = 1, nmod
            do iw = 1, nw
               cdist(iw,imod,iitem,iact) = 0.
            end do
         end do
      end do
   end do
   do iact = 1, nact2
      do iitem = 1, nitem
         do imod = 1, nmod
            do iw=1,nw
               cdols(iw,imod,iitem,iact) = 0.
            end do
         end do
      end do
```

```
end do
do iitem = 1, nitem
   do imod = 1, nmod
      ddols(imod,iitem) = 0.
   end do
end do
do iact = 1, nact2
   do icon = 1, ncon
      do imod = 1, nmod
         do iw = 1, nw
            fdols(iw,imod,icon,iact) = 0.
            fdist(iw,imod,icon,iact) = 0.
            hkey(iw,imod,icon,iact) = 0.
         end do
      end do
   end do
end do
do icsi = 1, ncsi
   do icon = 1, ncon
      do imod = 1, nmod
         gdols(imod,icon,icsi) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do icon = 1, ncon
      do imod = 1, nmod
         do iw = 1, nw
            gdist(iw,imod,icon,iact) = 0.
         end do
      end do
   end do
end do
do icon = 1, ncon
   do imod = 1, nmod
      hdols(imod,icon) = 0.
   end do
end do
do iw = 1,nw
   do iact = 1, nact2
      do imod = 1, nmod
         idols(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         result(iw,imod,iact) = 0.
         result2(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         resulta(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         resultb(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         resultf(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         work(iw,imod,iact) = 0.
         work2(iw,imod,iact) = 0.
         resultj(iw,imod,iact) = 0.
```

```
         end do
      end do
   end do
   do imod = 1, nmod
      do ishp = 1,nshp3
         jdols(imod,ishp) = 0.
         mixallied(imod,ishp) = 0.
      end do
   end do
   do ishp = 1, nshp2
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               mixkey(iw,imod,iact,ishp) = 0.0
               mixkey2(iw,imod,iact,ishp) = 0.0
               mixdist(iw,imod,iact,ishp) = 0.0
            end do
         end do
      end do
   end do
   print*, 'Matrices initialized '

   open(25,file='mods12_mp00prc.dat',recl=1200) !  MODS 1&2 offices mail proc IOCS data
31 format(a1167,f15.5,i2,i2,i3,i5)

   cnt = 0
   ier = 0
   tot_dol = 0.0
   tot_dol2 = 0.0
   totj =  0.0
   ct_cntr = 0
   ct_unid = 0
   cost_cntr = 0.0
   cost_unid = 0.0
   ct_nowgt = 0

   do while (ier.eq.0)

      read(25,31,iostat=ier,end=100) rec,dlrs,modgrp,ldc,iw,actv

      cnt = cnt + 1
      iw = 1

      read(f114,'(i3)') if114
      read(f9805,'(i4)') if9805
      read(f9806,'(i4)') if9806
      read(f9250,'(f10.0)') rf9250
      read(f166,'(i2)') if166
      read(f167,'(i2)') if167

      dlrs = rf9250/100000.

c     Break out Std A ECR Saturation and High Density into separate activity codes
      if ((actv.eq.1311).or.(actv.eq.2311).or.(actv.eq.3311).or.(actv.eq.4311)) then !  Std A  WSH/WSS
         if (f9619.eq.'1') then !  WSS
            if (actv.eq.1311) actv = 1313
            if (actv.eq.2311) actv = 2313
            if (actv.eq.3311) actv = 3313
            if (actv.eq.4311) actv = 4313
         end if
      end if

      if ((actv.eq.1331).or.(actv.eq.2331).or.(actv.eq.3331).or.(actv.eq.4331)) then !  Std A NP WSH/WSS
         if (f9619.eq.'1') then !  WSS
            if (actv.eq.1331) actv = 1333
            if (actv.eq.2331) actv = 2333
            if (actv.eq.3331) actv = 3333
            if (actv.eq.4331) actv = 4333
         end if
      end if

c     Any "auto" ECR flats or parcels are assumed to be basic ECR
      if (actv.eq.2312) actv = 2310
      if (actv.eq.3312) actv = 3310
      if (actv.eq.4312) actv = 4310
      if (actv.eq.2332) actv = 2330
      if (actv.eq.3332) actv = 3330
      if (actv.eq.4332) actv = 4330

c     Identifies tallies for PRC migrated and fixed activity codes
```

```
      if ((f9806.eq.'6320').or.(f9806.eq.'6330').or.(f9806.eq.'6430').or.
     &    (f9806.eq.'6460').or.(f9806.eq.'6480').or.(f9806.eq.'6495').or.
     &    (f9806.eq.'6500').or.(f9806.eq.'6511').or.(f9806.eq.'6512').or.
          (f9806.eq.'6514').or.(f9806.eq.'6516').or.(f9806.eq.'6519').or.
     &    (f9806.eq.'6610').or.(f9806.eq.'6620').or.(f9806.eq.'6630').or.
     &    (f9806.eq.'6640').or.(f9806.eq.'6650').or.(f9806.eq.'6660').or.
     &    (f9806.eq.'6420').or.(f9806.eq.'6210').or.(f9806.eq.'6220').or.
     &    (f9806.eq.'6230').or.(f9806.eq.'6240')) then
         fixed(modgrp) = fixed(modgrp) + dlrs
         total(modgrp) = total(modgrp) + dlrs
         goto 99
      else if (((actv.ge.5020).and.(actv.le.5195)).or.
     &        ((actv.ge.6000).and.(actv.le.6200)).or.
     &        (((actv.eq.6521).or.(actv.eq.6523)).and.
     &        ((f260.eq.'09').or.(f260.eq.'10').or.
     &        (f260.eq.'17').or.((f260.ge.'24').and.(f260.le.'26'))))) then
         migrate(modgrp) = migrate(modgrp) + dlrs
         goto 99
      else
         total(modgrp) = total(modgrp) + dlrs
      end if

      tot_dol = tot_dol + dlrs

c     International mixed codes are remapped to direct codes

      if (actv.eq.5434) then
         actv = 4755
      else if (actv.eq.5444) then
         actv = 4750
      else if (actv.eq.5454) then
         actv = 4810
      else if (actv.eq.5464) then
         actv = 4850
      else if (actv.eq.5433) then
         actv = 3755
      else if (actv.eq.5443) then
         actv = 3750
      else if (actv.eq.5453) then
         actv = 3810
      else if (actv.eq.5463) then
         actv = 3850
      else if (actv.eq.5432) then
         actv = 2755
      else if (actv.eq.5442) then
         actv = 2750
      else if (actv.eq.5452) then
         actv = 2810
      else if (actv.eq.5462) then
         actv = 2850
      else if (actv.eq.5431) then
         actv = 1755
      else if (actv.eq.5441) then
         actv = 1750
      else if (actv.eq.5451) then
         actv = 1810
      else if (actv.eq.5461) then
         actv = 1850
      end if

      ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape
      ishpprc = shapeprc(actv) !  Subroutine assigns shape for PRC mixed allieddistribution

c     Give class specific mixed mail codes a PRC shape of other
      if ((actv.eq.5430).or.(actv.eq.5440).or.(actv.eq.5450).or.
     &    (actv.eq.5460).or.(actv.eq.5470).or.(actv.eq.5480)) then
         ishpprc = 5
      else if (actv.eq.5340) then
         ishpprc = 5
      else if ((actv.ge.5300).and.(actv.le.5480)) then
         print *,'mixkey actv =',f9806,' in pool=',modgrp
      end if

c     Assign handling category
      if (((actv.ge.1000).and.(actv.le.4950)).or.((actv.ge.5300).and.(actv.le.5464))) then
         hand = 1             ! direct (non-special services)
      else if ((actv.ge.10).and.(actv.lt.1000)) then
         if (f129.eq.'C') then
            hand = 5          !  Not handling (special service)
         else if ((actv.eq.60).or.(actv.eq.190).or.
```

```
     &            ((actv.ge.210).and.(actv.le.300))) then
                hand = 1          ! direct (non-special services)
              else if (actv.eq.90) then
                 if ((modgrp.eq.24).or.(modgrp.eq.25).or.(modgrp.eq.27).or.(modgrp.eq.17) !  Bus Reply, Express, Registry, 1CancMPP
                    .or.(modgrp.eq.30).or.((modgrp.ge.37).and.(modgrp.le.40))) then !  Misc, LDC 48
                    hand = 1      !  direct (special service)
                 else
                    hand = 5      !  Not handling (special service)
                 end if
              else if (((actv.ge.10).and.(actv.le.50)).or.((actv.ge.70).and.(actv.le.80))) then
                 if ((modgrp.eq.24).or.(modgrp.eq.25).or.(modgrp.eq.27) !  Bus Reply, Express, Registry
     &              .or.(modgrp.eq.30).or.((modgrp.ge.37).and.(modgrp.le.40))) then !
                    hand = 1      !  direct (special service)
                 else
                    hand = 5      !  Not handling (special service)
                 end if
              else
                 hand = 5         !  Not handling (special service)
              end if
           else if ((f9214.ge.'A').and.(f9214.le.'P').and.(actv.ne.9999)) then
              hand = 2            ! mixed item
           else if ((f9219.ge.'A').and.(f9219.le.'J').and.(actv.ne.9999)) then
              hand = 3            ! mixed container
           else
              hand = 4            ! not handling mail
           end if

           iitem = searchc(codes,nitem,f9214) !  Assign item type
           icon = searchc(codes,ncon,f9219) !  Assign container type
           iact = searchi(acodes,nact2,actv) !  Activity codes

c     Assign weight increment
           if (hand.eq.1) then
              if (actv.ge.1000) then
                 iw = weight(f165,if166,if167,ct_nowgt,nw) !  Subroutine assigns weight increment
              else
                 iw = nw          !  Special service activities assumed to have no record weight
              end if
           else
              iw = nw
           end if

c          Direct tallies in OSS operations are used for RBCS distribution key
           if ((hand.eq.1).and.(((if114.ge.271).and.(if114.le.278)).or.
     &        ((if114.ge.971).and.(if114.le.978)))) then
              result(iw,nmod,iact) = result(iw,nmod,iact) + dlrs !  LDC 15 Proxy distrib key using BCS, DBCS MODS codes
           end if

           if ((hand.eq.1).and.(iact.eq.0)) then
              print *,'missing direct activity code = ',actv,' modgrp = ',modgrp
           end if

c          Assigns shape codes for PRC allied mixed-mail and not-handling distribution
           if (actv.eq.5610) then
              mixshp = 1          ! mixed letters
           else if (actv.eq.5620) then
              mixshp = 2          ! mixed flats
           else if (actv.eq.5700) then
              mixshp = 3          ! mixed parcels
           else if ((actv.eq.5750).or.(actv.eq.6523)) then
              mixshp = 4          ! mixed all shapes
           else if (hand.eq.4) then
              mixshp = 4          ! mixed all shapes
           else
              mixshp = 0
           end if

           if ((mixshp.eq.0).and.((hand.ge.2).and.(hand.le.4))) then
              print *,'mixshp error pool=',modgrp,' hand =',hand,' actv=',f9806,actv
           end if

c     Mixed allied costs for mixed items and containers
           if (((hand.eq.2).or.(hand.eq.3)).and.(modgrp.lt.nmod2)) then
              if (mixshp.gt.0.) then
                 mixallied(modgrp,mixshp) = mixallied(modgrp,mixshp) + dlrs
              end if
           end if

c          Assigns direct tallies to matrix for PRC mixed and not-handling distribution
           if ((ishpprc.gt.0).and.(ishpprc.le.5)) then
```

```
            mixkey(iw,modgrp,iact,ishpprc) = mixkey(iw,modgrp,iact,ishpprc) + dlrs
         end if

C     Single piece being handled, Assign to A matrix
         if ((hand.eq.1).and.(iitem.eq.0).and.(icon.eq.0)) then
            if (iact.gt.0) then
               if ((modgrp.gt.0).and.(modgrp.le.nmod)) then
                  adols(iw,modgrp,iact,ishp)=adols(iw,modgrp,iact,ishp) + dlrs
                  atot = atot + dlrs
                  acnt = acnt + 1
                  tot_dol2 = tot_dol2 + dlrs
               else
                  print *,' bad MODS in matrix A ',f114, modgrp, dlrs
               end if
            else
               print *,'Bad activity in matrix A = ',actv,' cost pool = ',modgrp
            end if

C*********************************************************************************
C     F129=C special service tallies -- assign to own matrix

         else if (hand.eq.5) then
            if (iact.gt.0) then
               idols(iw,modgrp,iact) = idols(iw,modgrp,iact) + dlrs
               itot = itot + dlrs
            else
               print *,'Bad ssv not-handling tally with code = ',actv,
     &            ' cost pool = ',modgrp
            end if

C*********************************************************************************
C     Not-handling mail tallies -- assign to J matrix
         else if (hand.eq.4) then
            if (modgrp.ne.15) then !  Exclude LDC 15
               jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
               jtot = jtot + dlrs
               jcnt = jcnt + 1
               tot_dol2 = tot_dol2 + dlrs
            else
               totj = totj + dlrs
            end if

C*********************************************************************************
C     Item being handled: separate items with direct activity codes from others

         else if ((f9214.ge.'A').and.(f9214.le.'P')) then
            if (hand.eq.1) then
               imat = 1          ! "B" or matrix - identical, top piece, or counted item
            else if (hand.eq.2) then
               imat = 3          ! "D" matrix - mixed, empty item
            else
               print *,'problem item in modgrp = ',modgrp
               imat = 0
            end if

C     "D" matrix: mixed or empty item
            if (imat.eq.3) then
               ddols(modgrp,iitem) = ddols(modgrp,iitem) + dlrs
               dtot = dtot + dlrs
               dcnt = dcnt + 1
               tot_dol2 = tot_dol2 + dlrs

C     "B" matrix: identical or top piece rule (direct item)
            else if (imat.eq.1) then
               bdols(iw,modgrp,iitem,iact) =
     &            bdols(iw,modgrp,iitem,iact) + dlrs
               btot = btot + dlrs
               bcnt = bcnt + 1
               tot_dol2 = tot_dol2 + dlrs
            end if

C     *************************************End Item*****************************
C     Container being handled: separate containers with direct activity codes from others
         else if (icon.gt.0) then

            if (modgrp.gt.0) then

               ct_cntr = ct_cntr + 1
               cost_cntr = cost_cntr + dlrs
```

```
            flag2=.false.

            if (f9901(1:1).eq.'%') then
               read(rec(340:406),451,iostat=ier) counts
            else
               read(rec(339:406),450,iostat=ier) counts
            end if
450         format(5(1x,f3.0),16f3.0)
451         format(f3.0,4(1x,f3.0),16f3.0)

            if (ier.ne.0) then
               flag2 = .true.
               j = 340
               do i = 1, ncsi
                  counts(i) = 0.
               end do
               ier = 0
            end if

            sum = 0.
            do i = 1, ncsi
               sum = sum + counts(i)
            end do

c     "F" matrix: identical mail in container (direct container)

            if (hand.eq.1) then
               fdols(iw,modgrp,icon,iact) =
     &            fdols(iw,modgrp,icon,iact) + dlrs
               ftot = ftot + dlrs
               fcnt = fcnt + 1
               tot_dol2 = tot_dol2 + dlrs

c     "H" matrix:  Uncounted, empty, or contents read error
            else if ((sum.eq.0.).or.flag2) then
               if (sum.gt.0.) then
                  print *,'G candidate pool =',modgrp,' $=',dlrs
               end if
               hdols(modgrp,icon) = hdols(modgrp,icon) + dlrs
               htot = htot + dlrs
               hcnt = hcnt + 1
               tot_dol2 = tot_dol2 + dlrs
               if (actv.ne.6523) then
                  ct_unid = ct_unid + 1
                  cost_unid = cost_unid + dlrs
               end if

c     "G" matrix:  container contents are "identified"
            else if (sum.gt.0.) then
               do icsi = 1, ncsi
                  gdols(modgrp,icon,icsi) = gdols(modgrp,icon,icsi) +
     &               (counts(icsi)/sum) * dlrs
               end do
               gtot = gtot + dlrs
               gcnt = gcnt + 1
               tot_dol2 = tot_dol2 + dlrs
            end if
         end if

C*************************************End Container*******************************
c     Any remaining tallies considered not handling mail
      else
         print *,'Shouldnt get here resid J'
         jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
         jtot = jtot + dlrs
         jcnt = jcnt + 1
         tot_dol2 = tot_dol2 + dlrs
      end if

99    end do
100   print *,' read exit = ',ier,' with ',cnt,' records ', ' dlrs = ', tot_dol
      print*, 'Total assigned dlrs = ', tot_dol2, ' j dols for LD 15 ', totj
      print*, 'Total container tallies ', ct_cntr, ' cost = ', cost_cntr
      print*, 'Total unidentified cntr tallies ', ct_unid, ' cost = ', cost_unid

C *************************************End Read Loop*********************************
c     Calculate PRC variabilities

      do imod = 1,nmod2
         if (imod.eq.15) then   !  LDC 15
```

```
            varprc(imod) = 1.
            migrate(imod) = 0.
         else
            varprc(imod) = 1 - (fixed(imod)+migrate(imod))/iocsdols(imod)
         end if
         print *,imod,' var prc =',varprc(imod),' var usps=',variable(imod)
      end do
      do imod = 1,nmod2
         print *,imod,' mig prc =',migrate(imod),' fix iocs=',fixed(imod)
         fixtot = fixtot + fixed(imod)*pooldols(imod)/iocsdols(imod)
      end do
      print *,'fixed costs (in cost pool $) =',fixtot

c     Redistribute no weight direct single piece costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + adols(iw,imod,iact,ishp)
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*adols(iw,imod,iact,ishp)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of direct single piece no weight costs

      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod2 !  Distbribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + adols(iw,j,iact,ishp)
                        sum = sum + adols(iw,j,iact,ishp)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  else
                     if (adols(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3a distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + adols(iw,imod,k,ishp)
                                 sum = sum + adols(iw,imod,k,ishp)
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                              adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                              end do
                           end do
                           adols(nw,imod,iact,ishp) = 0.0
```

```
                  else
                     print*,'Level 4a distribution of act = ',acodes(iact)
                     do k = begmail, nact2
                        do iw = 1, nw2
                           actshr2(iw,k) = 0.
                        end do
                     end do
                     do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                        if (class(k).eq.class(iact)) then !  Same subclass
                           do j = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                              do iw = 1, nw2 !  Disbribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + adols(iw,j,k,ishp)
                                 sum = sum + adols(iw,j,k,ishp)
                              end do
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do k = begmail, nact2
                           do iw = 1, nw2
                              adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                           adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                           end do
                        end do
                        adols(nw,imod,iact,ishp) = 0.0
                     else
                        print*,'unable to distribute no weight for ',
     &                     imod,' act = ',acodes(iact), ' cost = ', adols(nw,imod,iact,ishp)
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

   Add in redistributed no weight direct single piece costs
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            do ishp = 1, nshp
               adols(iw,imod,iact,ishp) = adols(iw,imod,iact,ishp) + adist(iw,imod,iact,ishp)
            end do
         end do
      end do
   end do

c  Redistribute no weight identical/top piece item costs
   do iact = begmail, nact2
      do imod = 1, nmod
         do iitem = 1, nitem
            if (bdols(nw,imod,iitem,iact).gt.0) then
               sum = 0.0
               do iw = 1, nw2 !  Distribute over all weight increments
                  sum = sum + bdols(iw,imod,iitem,iact)
               end do
               if (sum.gt.0.0) then
                  do iw = 1, nw2
                     bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                  bdols(nw,imod,iitem,iact)*bdols(iw,imod,iitem,iact)/sum
                  end do
                  bdols(nw,imod,iitem,iact) = 0.0
               end if
            end if
         end do
      end do
   end do

c  Residual distribution of identical/top piece items no weight costs
   do iact = begmail, nact2
      do imod = 1, nmod
         do iitem = 1, nitem
            if (bdols(nw,imod,iitem,iact).gt.0.0) then
               sum = 0.0
               do iw = 1, nw2
                  actwgt(iw) = 0.0
               end do
               do j = 1, nitem !  Distribute over all item types
                  do iw = 1, nw2 !  Distribute over all weight increments
```

```
                  actwgt(iw) = actwgt(iw) + bdols(iw,imod,j,iact)
                  sum = sum + bdols(iw,imod,j,iact)
               end do
            end do
            if (sum.gt.0) then
               do iw = 1, nw2
                  bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &               bdols(nw,imod,iitem,iact)*actwgt(iw)/sum
               end do
               bdols(nw,imod,iitem,iact) = 0.0
            else
               if (bdols(nw,imod,iitem,iact).gt.0.0) then
                  print*, 'Level 3 b distribution of act = ', acodes(iact)
                  do iw = 1, nw2
                     actwgt(iw) = 0.
                  end do
                  do k = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     do j = 1,nitem !  Distribute over all item types
                        do iw = 1, nw2 !  Distribute over all weight increments
                           actwgt(iw) = actwgt(iw) + bdols(iw,k,j,iact)
                           sum = sum + bdols(iw,k,j,iact)
                        end do
                     end do
                  end do
                  if (sum.gt.0.) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                     bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  else
                     print*, 'Level 4 b distribution of act = ', acodes(iact)
                     do iw = 1, nw2
                        actwgt(iw) = 0.
                     end do
                     do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                        if (class(k).eq.class(iact)) then !  Same subclass
                           do j = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                              do l = 1,nitem !  Distribute over all item types
                                 do iw = 1, nw2 !  Disbribute over all weight increments
                                    actwgt(iw) = actwgt(iw) + bdols(iw,j,l,k)
                                    sum = sum + bdols(iw,j,l,k)
                                 end do
                              end do
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do iw = 1, nw2
                           bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                        bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                        end do
                        bdols(nw,imod,iitem,iact) = 0.0
                     else
                        print*,'unable to distribute no weight for b, ',
     &                     imod,' act = ',acodes(iact), ' cost = ', bdols(nw,imod,iitem,iact)
                     end if
                  end if
               end if
            end if
         end if
      end do
   end do
end do

c  Add in redistributed no weight identical/top piece item costs
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            do iitem = 1, nitem
               bdols(iw,imod,iitem,iact) = bdols(iw,imod,iitem,iact) + bdist(iw,imod,iitem,iact)
               bdist(iw,imod,iitem,iact) = 0.0
            end do
         end do
      end do
   end do

c  Residual distribution of identical/top piece container no weight costs
   do iact = begmail, nact2
      do imod = 1, nmod
```

```
      do icon = 1, ncon
         if (fdols(nw,imod,icon,iact).gt.0) then
            sum = 0.0
            do iw = 1, nw2 !  Distribute over all weight increments
               sum = sum + fdols(iw,imod,icon,iact)
            end do
            if (sum.gt.0.0) then
               do iw = 1, nw2
                  fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &               fdols(nw,imod,icon,iact)*fdols(iw,imod,icon,iact)/sum
               end do
               fdols(nw,imod,icon,iact) = 0.0
            end if
         end if
      end do
   end do
end do

c     Residual distribution of identical/top piece container no weight costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do icon = 1, ncon
               if (fdols(nw,imod,icon,iact).gt.0.0) then
                  sum = 0.0
                  check = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod2 !  Distbribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + fdols(iw,j,icon,iact)
                        sum = sum + fdols(iw,j,icon,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                     fdols(nw,imod,icon,iact)*actwgt(iw)/sum
                     end do
                     fdols(nw,imod,icon,iact) = 0.0
                  else
                     if (fdols(nw,imod,icon,iact).gt.0.) then
                        print*,'Level 3 distribution of f act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + fdols(iw,imod,icon,k)
                                 sum = sum + fdols(iw,imod,icon,k)
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                              fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                              end do
                           end do
                           fdols(nw,imod,icon,iact) = 0.0
                        else
                           print*,'Level 4 distribution f of act = ',acodes(iact)
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 actshr2(iw,k) = 0.
                              end do
                           end do
                           do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                              if (mixclass(k).eq.mixclass(iact)) then !  Same subclass
                                 do j = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                                    do iw = 1, nw2 !  Distribute over all weight increments
                                       actshr2(iw,k) = actshr2(iw,k) + fdols(iw,j,icon,iact)
                                       sum = sum + fdols(iw,j,icon,iact)
                                    end do
                                 end do
                              end if
```

```
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                              fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                              end do
                           end do
                           fdols(nw,imod,icon,iact) = 0.0
                        else
                           print*,'unable to distribute no weight f for ',
     &                        imod,' act = ',acodes(iact), ' cost = ', fdols(nw,imod,icon,iact)
                        end if
                     end if
                  end if
               end if
            end do
         end do
      end do

c     Add in redistributed no weight identical/top piece container costs
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do icon = 1, ncon
                  fdols(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + fdist(iw,imod,icon,iact)
                  fdist(iw,imod,icon,iact) = 0.0
               end do
            end do
         end do
      end do

c     Redistribution of no weight for mixkey matrix
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp2
               if (mixkey(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + mixkey(iw,imod,iact,ishp)
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp)*mixkey(iw,imod,iact,ishp)/sum
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of mixkey matrix no weights
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp2
               if (mixkey(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + mixkey(iw,j,iact,ishp)
                        sum = sum + mixkey(iw,j,iact,ishp)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  else
                     if (mixkey(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3 distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
```

```
            actshr2(iw,k) = 0.
         end do
      end do
      do k = 1, nact2 !  Distribute over all activity codes within same subclass
         if (class(k).eq.class(iact)) then !  Same subclass
            do iw = 1, nw2 !  Distribute over all weight increments
               actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,imod,k,ishp)
               sum = sum + mixkey(iw,imod,k,ishp)
            end do
         end if
      end do
      if (sum.gt.0.) then
         do k = begmail, nact2
            do iw = 1, nw2
               mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &            mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
            end do
         end do
         mixkey(nw,imod,iact,ishp) = 0.0
      else
         print*,'Level 4 distribution of act = ',acodes(iact)
         do k = begmail, nact2
            do iw = 1, nw2
               actshr2(iw,k) = 0.
            end do
         end do
         do k = begmail, nact2 !  Distribute over all non spec serv  activity codes within same subclass
            if (class(k).eq.class(iact)) then !  Same subclass
               do j = 1,nmod !  Distribute over all cost pools
                  do iw = 1, nw2 !  Distribute over all weight increments
                     actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                     sum = sum + mixkey(iw,j,k,ishp)
                  end do
               end do
            end if
         end do
         if (sum.gt.0.) then
            do k = begmail, nact2
               do iw = 1, nw2
                  mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &               mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
               end do
            end do
            mixkey(nw,imod,iact,ishp) = 0.0
         else
            if (ishp.eq.1) then !  assign card directely to < 1/2 oz increment
               print*, 'Assign card directly to < 1/2 oz increment' !
               mixdist(1,imod,iact,ishp) = mixdist(1,imod,iact,ishp) +
     &            mixkey(nw,imod,iact,ishp)
               mixkey(nw,imod,iact,ishp) = 0.0
            else
               print*,'Level 5 distribution of act = ',acodes(iact)
               do k = begmail, nact2
                  do iw = 1, nw2
                     actshr2(iw,k) = 0.
                  end do
               end do
               do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                  if (class(k).eq.class(iact)) then !  Same subclass
                     do j = 1,nmod !  Distribute over all cost pools
                        do iw = 1, nw2 !  Distribute over all weight increments
                           actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                           sum = sum + mixkey(iw,j,k,ishp)
                        end do
                     end do
                  end if
               end do
               if (sum.gt.0.) then
                  do k = begmail, nact2
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                     end do
                  end do
                  mixkey(nw,imod,iact,ishp) = 0.0
               else
                  print*,'unable to distribute no weight for ',
     &               imod,' act = ',acodes(iact), ' cost = ', mixkey(nw,imod,iact,ishp)
               end if
            end if
```

```
                        end if
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c     Add in redistributed no weight mixed mail key costs
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do ishp = 1, nshp2
                  mixkey(iw,imod,iact,ishp) = mixkey(iw,imod,iact,ishp) + mixdist(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

c     Redistribution of 53XX and 54XX codes in mixed key for PRC method
      do iw = 1,nw
         do imod = 1,nmod
            do iact = 1,nact2
               do ishp = 1,3
                  mixkey2(iw,imod,iact,ishp) =  mixkey(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

      do imod = 1,nmod2
         do iact = 1, nact2
            if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
     &         (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then  !  Intl mixed-mail
               sum = 0.
               do iw = 1,nw
                  do j = 1,nact2
                     do k = 1,3
                        actshr4(iw,j,k) = 0.
                     end do
                  end do
               end do
               if (acodes(iact).eq.5430) then !  Intl mixed U.S. Origin Surface
                  do j = 1,nact2 !  Distribute over all actv codes
                     if (((mixclass(j).ge.18).and.(mixclass(j).le.21)).or.
     &                  ((mixclass(j).ge.32).and.(mixclass(j).le.35))) then !  Intl U.S. Origin Surface direct actvs
                        do iw = 1,nw !  Distribute over all weight increments
                           do k = 1,3 !  Distribute over all PRC shape categories
                              do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                                 sum = sum + mixkey(iw,i,j,k)
                                 actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                              end do
                           end do
                        end do
                     end if
                  end do
               else if (acodes(iact).eq.5440) then !  Intl Mixed U.S. Origin Air Mail
                  do j = 1,nact2 !  Distribute over all actv codes
                     if (((mixclass(j).ge.23).and.(mixclass(j).le.26)).or.
     &                  ((mixclass(j).ge.37).and.(mixclass(j).le.40))) then !  Intl U.S. Origin Air Mail direct actvs
                        do iw = 1,nw !  Distribute over all weight increments
                           do k = 1,3 !  Distribute over all PRC shape categories
                              do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                                 sum = sum + mixkey(iw,i,j,k)
                                 actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                              end do
                           end do
                        end do
                     end if
                  end do
               else if ((acodes(iact).eq.5450).or.(acodes(iact).eq.5470)) then !  Intl mixed Foreign Oridgin Surface
                  do j = 1,nact2 !  Distribute over all actv codes
                     if ((mixclass(j).ge.46).and.(mixclass(j).le.47)) then !  Intl mixed Foreign Oridgin Surface direct actvs
                        do iw = 1,nw !  Distribute over all weight increments
                           do k = 1,3 !  Distribute over all PRC shape categories
                              do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                                 sum = sum + mixkey(iw,i,j,k)
                                 actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                              end do
```

```
                  end do
               end do
            end if
         end do
      else              ! 5460 & 5480 - Intl mixed Foreign Origin Airmail
         do j = 1,nact2 !  Distribute over all actv codes
            if ((mixclass(j).ge.48).and.(mixclass(j).le.50)) then !  Intl Foreign Origin Airmail direct actvs
               do iw = 1,nw !  Distribute over all weight increments
                  do k = 1,3 !  Distribute over all PRC shape categories
                     do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                        sum = sum + mixkey(iw,i,j,k)
                        actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                     end do
                  end do
               end do
            end if
         end do
      end if
      if (sum.gt.0.) then
         do j = 1,nact2
            do iw = 1,nw
               do k = 1,3
                  mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &               actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
               end do
            end do
         end do
      else
         do iw = 1,nw
            if (mixkey(iw,imod,iact,5).gt.0.) then
               print*,'cant redistribute code=',acodes(iact),' pool=',imod
            end if
         end do
      end if
      if (sum.gt.0.) then
         do j = 1,nact2
            do iw = 1,nw
               do k = 1,3
                  mixkey2(iw,imod,j,k) = mixkey2(iw,imod,j,k) +
     &               actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
               end do
            end do
         end do
      else
         do iw = 1,nw
            if (mixkey(iw,imod,iact,5).gt.0.) then
               print*,'cant redistribute code=',acodes(iact),' pool=',imod
            end if
         end do
      end if
   else if (acodes(iact).eq.5340) then !  Std A Mixed

c     Use across-pool key for handling mixed

      sum = 0.
      do iw = 1,nw
         do j = 1,nact2
            do k = 1,3
               actshr4(iw,j,k) = 0.
            end do
         end do
      end do
      do j = 1,nact2   !  Distribute over all actv codes
         if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
            do iw = 1,nw !  Distribute over all weight increments
               do k = 1,3 !  Distribute over all PRC shape categories
                  do i = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     sum = sum + mixkey(iw,i,j,k)
                     actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                  end do
               end do
            end do
         end if
      end do
      if (sum.gt.0.) then
         do j = 1,nact2
            do iw = 1,nw
               do k = 1,3
                  mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &               actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
```

```
                  end do
               end do
            end do
         else
            do iw = 1,nw
               if (mixkey(iw,imod,iact,5).gt.0.) then
                  print*,'cant redistribute code=',acodes(iact),' pool=',imod
               end if
            end do
         end if

c        Use within-pool DK for not-handling

            sum = 0.
            do iw = 1,nw
               do j = 1,nact2
                  do k = 1,3
                     actshr4(iw,j,k) = 0.
                  end do
               end do
            end do
            do j = 1,nact2   !  Distribute over all actv codes
               if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
                  do iw = 1,nw !  Distribute over all weight increments
                     do k = 1,3 !  Distribute over all PRC shape categories
                        sum = sum + mixkey(iw,imod,j,k)
                        actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,imod,j,k)
                     end do
                  end do
               end if
            end do
            if (sum.gt.0.) then
               do j = 1,nact2
                  do iw = 1,nw
                     do k = 1,3
                        mixkey2(iw,imod,j,k) = mixkey2(iw,imod,j,k) +
     &                     actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                     end do
                  end do
               end do
            else
               do iw = 1,nw
                  if (mixkey(iw,imod,iact,5).gt.0.) then
                     print*,'cant redistribute code=',acodes(iact),' pool=',imod
                  end if
               end do
            end if
         end if
      end do
   end do

c  Sum up for all shapes key
   do iw = 1,nw
      do imod = 1,nmod
         do iact = 1,nact2
            do ishp = 1,3
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + mixkey(iw,imod,iact,ishp)
            end do
         end do
      end do
   end do

c  Distribute mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
   do imod = 1, nmod2
      do iitem = 1, nitem
         if (ddols(imod,iitem).gt.0.) then
            sum = 0.
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  sum = sum + bdols(iw,imod,iitem,iact)
               end do
            end do
            if (sum.gt.0) then
               do iact = 1, nact2
                  do iw = 1, nw
                     cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
     &                  ddols(imod,iitem) * bdols(iw,imod,iitem,iact) / sum
                  end do
               end do
               ddols(imod,iitem) = 0.
```

```
            end if
          end if
        end do
      end do

c     Distribute remaining mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix)
      do iitem = 1, nitem
         do imod = 1, nmod2
            if (ddols(imod,iitem).gt.0.) then
               sum = 0
               do iact = 1, nact2
                  do iw = 1, nw
                     actshr(iw,iact) = 0.
                  end do
               end do
               do iact = 1, nact2 !  Distribute over all activity codes
                  do j = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     do iw = 1, nw !  Distribute over all weight increments
                        actshr(iw,iact) = actshr(iw,iact) + bdols(iw,j,iitem,iact)
                        sum = sum + bdols(iw,j,iitem,iact)
                     end do
                  end do
               end do
               if (sum.gt.0.) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
     &                     ddols(imod,iitem) * actshr(iw,iact) / sum
                     end do
                  end do
               else
                  print *,' unable to dist D dols for iitem = ',iitem,', ',ddols(imod,iitem)
               end if
            end if
         end do
      end do

C     Sum distributed mixed/empty item costs in "C" matrix
      print *,' summing B, C, and D '
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = 1, nmod2
               do iw = 1, nw
                  cdols(iw,imod,iitem,iact) = cdols(iw,imod,iitem,iact) + cdist(iw,imod,iitem,iact)
                  cdist(iw,imod,iitem,iact) = 0.
               end do
            end do
         end do
      end do

C     Distribute "identified" container costs ("G" matrix)

      do imod = 1, nmod

C        Identified containers are distributed within
C        the same MODS group if possible; exception: Platform dist across all allied labor.

         do icsi = 1, ncsi
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            if (imod.ne.20) then !  Excludes Platform
               if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
                  do iact = begmail, nact2 !  Distribute over all non special-service activity codes
                     do iw = 1, nw !  Distribute over all weight increments
                        sum = sum + adols(iw,imod,iact,icsi)
                        actshr(iw,iact) = actshr(iw,iact) + adols(iw,imod,iact,icsi)
                     end do
                  end do
               else             ! Items distributed upon direct item costs ("B" matrix)
                  iitem = icsi - nshp2 !  Distribute over all item types
                  do iact = 1, nact2 !  Distribute over all activity codes
                     do iw = 1, nw ! Distribute over all weight increments
                        sum = sum + bdols(iw,imod,iitem,iact)
                        actshr(iw,iact) = actshr(iw,iact) + bdols(iw,imod,iitem,iact)
                     end do
```

```
            end do
         end if
      else                ! Distribute Platform over all ops
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = begmail, nact2 !  Distribute over all non special-service activity codes
               do iw = 1, nw ! Distribute over all weight increments
                  do i = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     if ((i.eq.10).or.((i.ge.16).and.(i.le.23))) then ! Distribute over allied labor cost pools
                        sum = sum + adols(iw,i,iact,icsi)
                        actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                     end if
                  end do
               end do
            end do
         else              ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw ! Distribute over all weight increments
                  do i = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     if ((i.eq.10).or.((i.ge.16).and.(i.le.23))) then ! Distribute over allied labor cost pools
                        sum = sum + bdols(iw,i,iitem,iact)
                        actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                     end if
                  end do
               end do
            end do
         end if
      end if
      if (sum.gt.0.) then
         do icon = 1, ncon
            if (gdols(imod,icon,icsi).gt.0.) then
               do iact = 1, nact2
                  do iw = 1, nw
                     gdist(iw,imod,icon,iact) =
     &                  gdist(iw,imod,icon,iact) +
     &                  gdols(imod,icon,icsi) *
     &                  actshr(iw,iact) / sum
                  end do
               end do
            end if
         end do
      else
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = begmail, nact2 !  Distribute over all non special-service activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     distsum = distsum + adols(iw,i,iact,icsi)
                     actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
         else  ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = 1,nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                     distsum = distsum + bdols(iw,i,iitem,iact)
                     actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
         end if
         if (distsum.gt.0) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        gdist(iw,imod,icon,iact) =
     &                     gdist(iw,imod,icon,iact) +
     &                     gdols(imod,icon,icsi) *
     &                     actshr(iw,iact)/distsum
                     end do
                  end do
               end if
            end do
         else
            print *,'shape distribution empty: mod = ',imod,
     &           ', shape = ',icsi
         end if
      end if
```

```
      end do
    end do                    ! End of "identified" container ("G" matrix) distribution

c   Sum direct container costs and distributed "identified container costs for uncounted container distribution
    do iact = 1, nact2
      do icon = 1, ncon
        do imod = 1, nmod2
          do iw = 1, nw
            hkey(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + gdist(iw,imod,icon,iact)
          end do
        end do
      end do
    end do

c     Distribute uncounted/empty containers ("H" matrix) using direct and distributed "identified"
C     container costs
      do imod = 1, nmod2
        do icon = 1, ncon
          sum = 0.
          do iact = 1, nact2  !  Distribute over all activity codes
            do iw = 1, nw     !  Distribute over all weight increments
              sum = sum + hkey(iw,imod,icon,iact)
            end do
          end do
          if (sum.gt.0) then
            do iact = 1, nact2
              do iw = 1, nw
                gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
     &             hdols(imod,icon) * hkey(iw,imod,icon,iact) / sum
              end do
            end do
            hdols(imod,icon) = 0.
          end if
        end do
      end do

c     Distribute remaining uncounted/empty container costs ("H" matrix) using direct/distributed
C     "identified" container costs ("F" matrix)
      do icon = 1, ncon
        do imod = 1, nmod2
          if (hdols(imod,icon).gt.0.) then
            sum = 0.
            do iact = 1, nact2
              do iw = 1, nw
                actshr(iw,iact) = 0.
              end do
            end do
            do iact = 1, nact2 !  Distribute over all activity codes
              do j = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                do iw = 1, nw !  Distribute over all weight increments
                  actshr(iw,iact) = actshr(iw,iact) + hkey(iw,j,icon,iact)
                  sum = sum + hkey(iw,j,icon,iact)
                end do
              end do
            end do
            if (sum.gt.0.) then
              do iact = 1, nact2
                do iw = 1, nw
                  gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
     &               actshr(iw,iact)/sum * hdols(imod,icon)
                end do
              end do
            else
              print *,' unable to dist h dols for imod = ',imod,
     &           ' icon = ',icon
            end if
          end if
        end do
      end do

c     Sum up all costs (direct and redistributed) except not handling costs ("J" matrix)
c     Pieces
      do ishp = 1, nshp
        do iact = 1, nact2
          do imod = 1, nmod2
            do iw = 1, nw
              result(iw,imod,iact) = result(iw,imod,iact) + adols(iw,imod,iact,ishp)
              result2(iw,imod,iact) = result2(iw,imod,iact) + adols(iw,imod,iact,ishp)
              resulta(iw,imod,iact) = resulta(iw,imod,iact) + adols(iw,imod,iact,ishp)
            end do
```

```
         end do
      end do
   end do
c  Not handling - special services
   do iact = 1, nact2
      do imod = 1, nmod2
         do iw = 1, nw
            result(iw,imod,iact) = result(iw,imod,iact) + idols(iw,imod,iact)
            result2(iw,imod,iact) = result2(iw,imod,iact) + idols(iw,imod,iact)
            resulta(iw,imod,iact) = resulta(iw,imod,iact) + idols(iw,imod,iact)
         end do
      end do
   end do
c  Items
   do imod = 1, nmod2
      if ((imod.eq.10).or.((imod.ge.16).and.(imod.le.23))) then !  Allied cost pools
         do iact = 1, nact2
            do iitem = 1, nitem
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
                  resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
               end do
            end do
         end do
c     Containers
         do iact = 1,nact2
            do icon = 1, ncon
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
               end do
            end do
         end do
      else                    !  Other cost pools
c     Items
         do iact = 1, nact2
            do iitem = 1, nitem
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
                  resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
               end do
            end do
c     Containers
            do icon = 1, ncon
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
               end do
            end do
         end do
      end if

   end do

C  Distribute mixed allied matrix on directs from all pools
   do imod = 1,nmod
      if ((imod.eq.10).or.((imod.ge.16).and.(imod.le.23))) then ! Allied cost pools only
         do ishp = 1,nshp3
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            do iact = 1, nact2 !  Distribute over all activity codes
```

```
                  do iw = 1, nw !  Distribute over all weight increments
                     do j = 1,nmod2 !  Distribute over all cost pools
                        actshr(iw,iact) = actshr(iw,iact) + mixkey(iw,j,iact,ishp)
                        sum = sum + mixkey(iw,j,iact,ishp)
                     end do
                  end do
               end do
               if (sum.gt.0) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        result(iw,imod,iact) = result(iw,imod,iact) +
     &                     mixallied(imod,ishp) * actshr(iw,iact) / sum
                        ishpprc = shapeprc(acodes(iact))
                        if ((ishpprc.ge.1).and.(ishpprc.le.3)) then
                           mixkey2(iw,imod,iact,ishpprc) = mixkey2(iw,imod,iact,ishpprc) +
     &                        mixallied(imod,ishp) * actshr(iw,iact)/sum
                        else if (actshr(iw,iact).gt.0.) then
                           print *,'pool=',imod,' act=',acodes(iact),' key=',actshr(iw,iact)
                        end if
                     end do
                  end do
               else
                  print *,' unable to distribute allied mixed dollars for ',imod,ishp
               end if
            end do
         end if
      end do

c     Generate allied not handling keys
      do ishp = 1, 3
         do iact = 1, nact2
            do imod = 1, nmod
               do iw = 1, nw
                  mixkey2(iw,imod,iact,4) = mixkey2(iw,imod,iact,4) + mixkey2(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

c     Distribute not handling costs ("J" matrix) using all other costs ("results" matrix)
      do imod = 1, nmod2
         sum = 0.
         distsum = 0.
c     Specific MODS cost pools - exclude LDC 15, allied labor, special services, and support cost pools
         if (((imod.le.15).and.(imod.ne.10)).or.((imod.ge.25).and.(imod.le.26)).or.
     &      (imod.eq.28).or.((imod.ge.32).and.(imod.le.37)).or.(imod.eq.42)) then
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            do iact = begmail, nact2 !  Distribute over all non special-service cost pools
               do iw = 1, nw    !  Distribute over all weight increments
                  actshr(iw,iact) = actshr(iw,iact) + result(iw,imod,iact)
                  sum = sum + result(iw,imod,iact)
               end do
            end do
            if (sum.gt.0) then
               do ishp = 1,nshp3
                  do iact = 1, nact2
                     do iw = 1, nw
                        work(iw,imod,iact) = work(iw,imod,iact) +
     &                     jdols(imod,ishp) * actshr(iw,iact) / sum
                        work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                     jdols(imod,ishp) * actshr(iw,iact) / sum
                     end do
                  end do
               end do
            else
               print *,' unable to distribute J dollars for ',imod
            end if

c     Special Service cost pools include special service activity codes
         else if (((imod.eq.24).or.(imod.eq.27)).or.(imod.eq.40).or.(imod.eq.41)) then
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
```

```
                actshr(iw,iact) = 0.
             end do
          end do
          do iact = 1, nact2  !  Distribute over all activity codes
             do iw = 1, nw    !  Distribute all weight increments
                actshr(iw,iact) = actshr(iw,iact) + result(iw,imod,iact)
                sum = sum + result(iw,imod,iact)
             end do
          end do
          if (sum.gt.0) then
             do ishp = 1,nshp3
                do iact = 1, nact2
                   do iw = 1, nw
                      work(iw,imod,iact) = work(iw,imod,iact) +
     &                   jdols(imod,ishp) * actshr(iw,iact) / sum
                      work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                   jdols(imod,ishp) * actshr(iw,iact) / sum
                      check = check +
     &                   jdols(imod,ishp) * actshr(iw,iact) / sum
                   end do
                end do
             end do
          else
             print *,' unable to distribute J dollars for ',imod
          end if

c     Allied not-handling (including cancellation for PRC) is distributed using shape-based keys
          else if (((imod.ge.16).and.(imod.le.23)).or.(imod.eq.10)) then !  Allied cost pools
             do ishp = 1,nshp3
                sum = 0.
                distsum = 0.
                do iact = 1, nact2
                   do iw = 1, nw
                      actshr(iw,iact) = 0.
                   end do
                end do
                do iact = 1, nact2 !  Distribute over all activity codes
                   do iw = 1, nw !  Distribute over all weight increments
                      actshr(iw,iact) = actshr(iw,iact) + mixkey2(iw,imod,iact,ishp)
                      sum = sum + mixkey2(iw,imod,iact,ishp)
                   end do
                end do
                if (sum.gt.0) then
                   do iact = 1, nact2
                      do iw = 1, nw
                         work(iw,imod,iact) = work(iw,imod,iact) +
     &                      jdols(imod,ishp) * actshr(iw,iact) / sum
                      end do
                   end do
                else
                   print *,' unable to distribute J dollars for ',imod
                end if
             end do

c     Alternative track for PRC methodology - allied cost pools
             sum = 0.
             distsum = 0.
             do iact = 1, nact2
                do iw = 1, nw
                   actshr(iw,iact) = 0.
                end do
             end do
             do iact = begmail, nact2 !  Distribute over all non special-service activity codes
                do iw = 1, nw    !  Distribute over all weight increments
                   actshr(iw,iact) = actshr(iw,iact) + result2(iw,imod,iact)
                   sum = sum + result2(iw,imod,iact)
                end do
             end do
             if (sum.gt.0) then
                do ishp = 1,nshp3
                   do iact = 1, nact2
                      do iw = 1, nw
                         work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                      jdols(imod,ishp) * actshr(iw,iact) / sum
                      end do
                   end do
                end do
             else
                print *,' unable to distribute J dollars for ',imod
             end if
```

```
c     1EEQMT is distributed across all MODS pools; distribution includes special services

      else if (imod.eq.29) then !  1EEqmt cost pool
         sum = 0.
         distsum = 0.
         do iact = 1, nact2
            do iw = 1, nw
               actshr(iw,iact) = 0.
            end do
         end do
         do iact = 1, nact2  !  Distribute over all activity codes
            do iw = 1, nw    !  Distribute over all weight increments
               do i = 1,nmod2 !  Distribute over all cost pools
                  actshr(iw,iact) = actshr(iw,iact) + result2(iw,i,iact)
                  sum = sum + result2(iw,i,iact)
               end do
            end do
         end do
         if (sum.gt.0) then
            do ishp = 1,nshp3
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                     work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            end do
         else
            print *,' unable to distribute J dollars for ',imod
         end if

c     1SUPPORT & 1MISC distributed across all Function 1 pools (including SpS)
      else if ((imod.eq.30).or.(imod.eq.31)) then
         sum = 0.
         distsum = 0.
         do iact = 1, nact2
            do iw = 1, nw
               actshr(iw,iact) = 0.
            end do
         end do
         do iact = 1, nact2  !  Distribute over all activity codes
            do iw = 1, nw    !  Distribute over all weight increments
               do i = 1,nmod2 !  Distribute over cost pools
                  if (((ldc1(i).ge.11).and.(ldc1(i).le.19))
     &                .and.(i.ne.24).and.(i.ne.27)) then !  All Function 1 cost pools except BusReply & Registry
                     actshr(iw,iact) = actshr(iw,iact) + result2(iw,i,iact)
                     sum = sum + result2(iw,i,iact)
                  end if
               end do
            end do
         end do
         if (sum.gt.0) then
            do ishp = 1,nshp3
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                     work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            end do
         else
            print *,' unable to distribute J dollars for ',imod
         end if

c     LD48 Oth & Adm distributed over Function 4 pools

      else if ((imod.eq.38).or.(imod.eq.39)) then !  LD48 Oth, LD48_Adm
         sum = 0.
         distsum = 0.
         do iact = 1, nact2
            do iw = 1, nw
               actshr(iw,iact) = 0.
            end do
         end do
         do iact = 1, nact2  !  Distribute over all activity codes
```

```
            do iw = 1, nw     !  Distribute over all weight increments
               do i = 1,nmod2 !  Distribute over cost pools
                  if (((ldc1(i).ge.41).and.(ldc1(i).le.49)).and.
                     (i.ne.40).and.(i.ne.41)) then !  Function 4 cost pools except LD48 SSV & LD49
                     actshr(iw,iact) = actshr(iw,iact) + result(iw,i,iact)
                     sum = sum + result(iw,i,iact)
                  end if
               end do
            end do
         end do
         if (sum.gt.0) then
            do ishp = 1,nshp3
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                     work2(iw,imod,iact) = work2(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            end do
         else
            print *,' unable to distribute J dollars for ',imod
         end if

      end if

   end do

c  Sum distributed not handling costs ("J" matrix) into handling costs ("results" matrix)
   print *,' summing J into all '
   do iact = 1, nact2
      do imod = 1, nmod2
         do iw = 1, nw
            result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
            result2(iw,imod,iact) = result2(iw,imod,iact) + work2(iw,imod,iact)
            resultj(iw,imod,iact) = work(iw,imod,iact)
            work(iw,imod,iact) = 0.
         end do
      end do
   end do

c  Redistribute class-specific mixed mail costs over appropriate class-specific direct activity codes,
C  weight increment, and within cost pools
   print *,' distributing mixed-mail items with class-specific codes'
   do imod = 1,nmod
      do iact = 1,nmixc1
         do iw = 1, nw
            if (result(iw,imod,nact+iact).gt.0.0) then
               sum = 0.
               do i = 1,nact
                  actshr3(i) = 0.
               end do
               do i = 1,nact !  Distribute over all direct activity codes
                  do j = 1,nw !  Distribute over all weight increments
                     if (mixmap(i,iact).gt.0) then
                        sum = sum + result(j,imod,mixmap(i,iact))
                        actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
     &                     + result(j,imod,mixmap(i,iact))
                     end if
                  end do
               end do
               if (sum.gt.0.) then
                  do i = 1,nact
                     if (mixmap(i,iact).gt.0) then
                        work(iw,imod,mixmap(i,iact)) =
     &                     work(iw,imod,mixmap(i,iact)) +
     &                     (result(iw,imod,nact+iact)*
     &                     actshr3(mixmap(i,iact))/sum)
                     end if
                  end do
                  result(iw,imod,nact+iact) = 0.
               else          !  Distribute over all cost pools
                  print*, 'Residual for mix actv code ', acodes(nact+iact)
                  sum = 0.
                  do i = 1,nact
                     actshr3(i) = 0.
                  end do
                  do i = 1, nact !  Distribute over all direct activity codes
                     do j = 1,nw !  Distribute over all weight increments
```

```
                        do k = 1, nmod !  Distribute over all cost pools
                           if (mixmap(i,iact).gt.0) then
                              sum = sum + result(j,k,mixmap(i,iact))
                              actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
                                 + result(j,k,mixmap(i,iact))
                           end if
                        end do
                     end do
                  end do
                  if (sum.gt.0.) then
                     do i = 1, nact
                        if (mixmap(i,iact).gt.0) then
                           work(iw,imod,mixmap(i,iact)) =
     &                        work(iw,imod,mixmap(i,iact)) +
     &                        (result(iw,imod,nact+iact)*
     &                        actshr3(mixmap(i,iact))/sum)
                        end if
                     end do
                     result(iw,imod,nact+iact) = 0.
                  else
                     print*, 'Mix actv code not distributed ', acodes(nact+iact),
     &                    ' cost = ', result(iw,imod,nact+iact), ' pool ', modcodes(imod)
                  end if
               end if
            end if
         end do
      end do
   end do

c     Sum distributed class-specific mixed-mail costs into all other costs
      do iact = 1, nact
         do imod = 1, nmod
            do iw = 1, nw
               result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
               work(iw,imod,iact) = 0.
            end do
         end do
      end do

c     Redistributes items with class-specific activity codes again for result2 matrix
      print *,' distributing mixed-mail items with class-specific codes'
      do imod = 1,nmod
         do iact = 1,nmixcl
            do iw = 1, nw
               if (result2(iw,imod,nact+iact).gt.0.0) then
                  sum = 0.
                  do i = 1,nact
                     actshr3(i) = 0.
                  end do
                  do i = 1,nact !  Distribute over all direct activity codes
                     do j = 1,nw !  Distribute over all weight increments
                        if (mixmap(i,iact).gt.0) then
                           sum = sum + result2(j,imod,mixmap(i,iact))
                           actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
     &                        + result2(j,imod,mixmap(i,iact))
                        end if
                     end do
                  end do
                  if (sum.gt.0.) then
                     do i = 1,nact
                        if (mixmap(i,iact).gt.0) then
                           work(iw,imod,mixmap(i,iact)) =
     &                        work(iw,imod,mixmap(i,iact)) +
     &                        (result2(iw,imod,nact+iact)*
     &                        actshr3(mixmap(i,iact))/sum)
                        end if
                     end do
                     result2(iw,imod,nact+iact) = 0.
                  else          !  Distribute over all cost pools
                     print*, 'Residual for mix actv code=', acodes(nact+iact),imod
                     sum = 0.
                     do i = 1,nact
                        actshr3(i) = 0.
                     end do
                     do i = 1, nact !  Distribute over all direct activity codes
                        do j = 1,nw !  Distribute over all weight increments
                           do k = 1, nmod !  Distribute over all cost pools
                              if (mixmap(i,iact).gt.0) then
                                 sum = sum + result2(j,k,mixmap(i,iact))
                                 actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
```

```
     &                              + result2(j,k,mixmap(i,iact))
                                 end if
                              end do
                           end do
                        end do
                        if (sum.gt.0.) then
                           do i = 1, nact
                              if (mixmap(i,iact).gt.0) then
                                 work(iw,imod,mixmap(i,iact)) =
     &                              work(iw,imod,mixmap(i,iact)) +
     &                              (result2(iw,imod,nact+iact)*
     &                              actshr3(mixmap(i,iact))/sum)
                              end if
                           end do
                           result2(iw,imod,nact+iact) = 0.
                        else
                           print*, 'Mix actv code not distributed ', acodes(nact+iact),
     &                        ' cost = ', result2(iw,imod,nact+iact), ' pool ', modcodes(imod)
                        end if
                     end if
                  end if
               end do
            end do
         end do

c        Sum distributed class-specific mixed-mail costs into all other costs - result2
         do iact = 1, nact
            do imod = 1, nmod
               do iw = 1, nw
                  result2(iw,imod,iact) = result2(iw,imod,iact) + work(iw,imod,iact)
                  work(iw,imod,iact) = 0.
               end do
            end do
         end do

c        Allied cost pool adjustment
         do imod = 1,nmod
            if ((imod.eq.10).or.((imod.ge.16).and.(imod.le.23))) then
               do iw = 1,nw
                  do iact = begmail,nact
                     prcadj1(imod) = prcadj1(imod) + result2(iw,imod,iact)*pooldols(imod)/iocsdols(imod)
                     prcadj2(imod) = prcadj2(imod) + result(iw,imod,iact)*pooldols(imod)/iocsdols(imod)
                  end do
               end do
               check1 = 0.
               check2 = 0.
               do iw = 1,nw
                  do iact = 1,nact
                     check1 = check1 + result2(iw,imod,iact)*pooldols(imod)/iocsdols(imod)
                     check2 = check2 + result(iw,imod,iact)*pooldols(imod)/iocsdols(imod)
                  end do
               end do
               print *,imod,prcadj1(imod),prcadj2(imod)
               print *,imod,check1,check2
            end if
         end do

C        Replace LDC 15 w/proxy distribution key
         do iact = 1, nact
            do iw = 1, nw
               result(iw,15,iact) = result(iw,nmod,iact)
            end do
         end do

c        Compute volume-variable costs for all cost pools except Support Fcn 1 & 4
         distsum = 0.
         distsum48 = 0.
         do imod = 1,nmod2
            sum = 0.
            if ((imod.eq.10).or.((imod.ge.16).and.(imod.le.23))) then !  Allied cost pools
               do iact = 1,nact
                  do iw = 1,nw
                     if (iact.lt.begmail) then !  Special service activity codes
                        varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &                     pooldols(imod)*varprc(imod)*result2(iw,imod,iact)/iocsdols(imod)
                        novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &                     pooldols(imod)*result2(iw,imod,iact)/iocsdols(imod)
                     else          !  Non special service activity codes
                        varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &                     pooldols(imod)*(varprc(imod)*result(iw,imod,iact)/iocsdols(imod))*
```

```
     &                  (prcadj1(imod)/prcadj2(imod))
                     novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &                  pooldols(imod)*(result(iw,imod,iact)/iocsdols(imod))*
                        (prcadj1(imod)/prcadj2(imod))
                  end if
               end do
            end do
         else if (imod.ne.15) then !  All other cost pools except LDC 15
            do iact = 1,nact
               do iw = 1,nw
                  varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &               pooldols(imod)*varprc(imod)*result(iw,imod,iact)/iocsdols(imod)
                  novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &               pooldols(imod)*result(iw,imod,iact)/iocsdols(imod)
               end do
            end do
         else                    !  LDC 15
            do iact = 1,nact
               do iw = 1,nw
                  sum = sum + result(iw,imod,iact)
               end do
            end do
            if (sum.gt.0.) then
               do iact = 1,nact
                  do iw = 1,nw
                     varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &                  pooldols(imod)*result(iw,imod,iact)/sum
                     novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &                  pooldols(imod)*result(iw,imod,iact)/sum
                  end do
               end do
            else
               print *,'unable to distribute $ = ',pooldols(imod),
     &            ' for mods pool ',modcodes(imod)
            end if
         end if
      end do

C     Write out results to a file
      open(80,file='mods00prc.data')
81    format(i3,i4,i3,8f18.9)

      do imod = 1, nmod2
         do iact = 1, nact
            do iw = 1, nw
               write (80,81) ldc1(imod), iact, iw, varcost(iw,imod,iact),
     &            novarcst(iw,imod,iact), result(iw,imod,iact),
     &            resulta(iw,imod,iact), resultb(iw,imod,iact),
     &            resultf(iw,imod,iact), resultj(iw,imod,iact),work(iw,imod,iact)
            end do
         end do
      end do

      Print *, ' Total Count and Dollars by Matrix '
      write (*,'(2x,a1,i6,f15.2)') 'A', acnt, atot
      write (*,'(2x,a1,i6,f15.2)') 'B', bcnt, btot
      write (*,'(2x,a1,i6,f15.2)') 'C', ccnt, ctot
      write (*,'(2x,a1,i6,f15.2)') 'D', dcnt, dtot
      write (*,'(2x,a1,i6,f15.2)') 'F', fcnt, ftot
      write (*,'(2x,a1,i6,f15.2)') 'G', gcnt, gtot
      write (*,'(2x,a1,i6,f15.2)') 'H', hcnt, htot
      write (*,'(2x,a1,6x,f15.2)') 'I', itot
      write (*,'(2x,a1,i6,f15.2)') 'J', jcnt, jtot

      end

C ------------------------------------------------------------------------------------
c     Assign PRC shape

      function shapeprc(actv)

      integer*4 shapeprc,actv

      shapeprc = 0

      if ((actv.lt.1000).or.(actv.ge.5610)) then
         shapeprc = 0     ! special service and mixed-mail
      else if ((actv.ge.1000).and.(actv.lt.2000)) then
         shapeprc = 1     ! letter
      else if ((actv.ge.2000).and.(actv.lt.3000)) then
```

```
         shapeprc = 2      ! flat
      else if ((actv.ge.3000).and.(actv.lt.5000)) then
         shapeprc = 3      ! parcel
      else
         shapeprc = 0      ! other?
      end if

      return
      end

C -------------------------------------------------------------------------------------
c     Assign shape

      function shapeind(actv,f9635,f9805)

      integer*4 shapeind, actv
      character*1  f9635
      character*4 f9805

      if (((actv.ge.1000).and.(actv.lt.2000)).or.(actv.eq.5431).or.(actv.eq.5441)
     &   .or.(actv.eq.5451).or.(actv.eq.5461)) then
         if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
            shapeind = 1         ! cards
         else
            shapeind = 2         ! letters
         end if
      else if (((actv.ge.2000).and.(actv.lt.3000)).or.(actv.eq.5432).or.(actv.eq.5442)
     &      .or.(actv.eq.5452).or.(actv.eq.5462)) then
         shapeind = 3            ! flats
      else if (((actv.ge.3000).and.(actv.lt.4000)).or.(actv.eq.5433).or.(actv.eq.5443)
     &      .or.(actv.eq.5453).or.(actv.eq.5463)) then
         shapeind = 4            ! IPPs
      else if (((actv.ge.4000).and.(actv.lt.5000)).or.(actv.eq.5434).or.(actv.eq.5444)
     &      .or.(actv.eq.5454).or.(actv.eq.5464)) then
         shapeind = 5            ! parcels
      else
         shapeind = 6            ! other?
      end if

      if (actv.eq.5340) then
         shapeind = 6  !  other
         if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
            shapeind = 1         ! cards
         end if
         if (f9635.eq.'A') then
            shapeind = 2  !  letters
         end if
         if ((f9635.eq.'D').or.(f9635.eq.'E')) then
            shapeind = 3  !  flats
         end if
         if ((f9635.eq.'F').or.(f9635.eq.'G').or.(f9635.eq.'J')) then
            shapeind = 4  !  IPPs
         end if
         if ((f9635.eq.'H').or.(f9635.eq.'I')) then
            shapeind = 5  !  parcels
         end if
      end if

      if ((actv.ge.10).and.(actv.lt.1000)) then
         if ((f9805(1:1).eq.'1').and.(((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K'))) then
            shapeind = 1  !  cards
         else if (f9805(1:1).eq.'1') then
            shapeind = 2  !  letters
         else if (f9805(1:1).eq.'2') then
            shapeind = 3  !  flats
         else if (f9805(1:1).eq.'3') then
            shapeind = 4  !  IPPs
         else if (f9805(1:1).eq.'4') then
            shapeind = 5  !  parcels
         else
            shapeind = 6  !  other
         end if
      end if

      return
      end

C -------------------------------------------------------------------------------------
c     Assign weight increment
```

```
function weight(f165,if166,if167,ct_nowgt,nw)

character*1  f165
integer*4    if166, if167, weight, ct_nowgt, nw

weight = 0

if (f165.eq.'A') then
   weight = 1              !  < 1/2 ounce
else if (f165.eq.'B') then
   weight = 2              !  1 ounces
else if (f165.eq.'C') then
   weight = 3              !  1 1/2 ounces
else if (f165.eq.'D') then
   weight = 4              !  2 ounces
else if (f165.eq.'E') then
   weight = 5              !  2 1/2 ounces
else if (f165.eq.'F') then
   weight = 6              !  3 ounces
else if (f165.eq.'G') then
   weight = 7              !  3 1/2 ounces
else if (f165.eq.'H') then
   weight = 8              !  4 ounces
else if (f165.eq.'I') then
   if (if166.eq.0) then    !  < 1 lb
      if (if167.gt.0) then
         weight = if167 + 4
      else
         weight = nw
         ct_nowgt = ct_nowgt + 1
      end if
   else if ((if166.eq.1).and.(if167.eq.0)) then ! 1 lb. 0 oz.
      weight = 20
   else if ((if166.gt.1).or.((if166.eq.1).and.(if167.gt.0))) then  ! >= 1 lb. 1 oz.
      weight = 21
   else
      weight = nw
      ct_nowgt = ct_nowgt + 1
   end if
else
   weight = nw
   ct_nowgt = ct_nowgt + 1
end if

return
end
```

```
      program sumclass_mod_ecr

c     Purpose:  Sum PRC methodology distributed volume-variable mail processing costs
C               for MODS 1&2 offices to subclass
c               Costs are calculated in the Fortran program modsproc00prc_wgt2.f
c               Breaks out ECR costs by Basic, Automated, Walk Sequence Saturation, and
C               Walk Sequence High Density

      implicit none

      integer*4 nact, ncl, nmod, nshp, nmat, nshp2, nw

      parameter (nmod = 42)      !  Number of cost pools
      parameter (nact = 261)     !  Number of activity codes
      parameter (ncl = 80)       !  Number of subclasses
      parameter (nshp = 3)       !  Number of shapes
      parameter (nmat = 8)       !  Number of cost categories
      parameter (nshp2 = 5)      !  Number of shapes (class map)
      parameter (nw = 22)        !  Number of weight increments

      real*8    dollars(nmat,nw,nmod,nact)
      real*8   cdols(nmat,nmod,ncl,nshp)

      integer*4 imod, iact, icl, i, j, k, shape, is
      integer*4 ier, shp(nact), iw, imod2
      integer*4 clmap(nact), mod(nmod), ldcl(nmod)

      character*14 grp(nmod)
      character*9  class(ncl), clcode
      character*9  class2(ncl)
      character*10 class3(ncl)
      character*4  acodes(nact), temp, acin(nshp2)
      character*5  shapetype(nshp)/'1Ltr ','2Flt ','3Pcl '/

      ier = 0

c     Map of cost pools
      open(30,file='costpools.00.new')
3.    format(i4,a14,i5)

      do i = 1, nmod
         read(30,32) mod(i), grp(i), ldcl(i)
      end do
      print *, 'Mod groups read'
      close(30)

c     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(a4)

      do i = 1, nact
         read (20,21) acodes(i)
         is = shape(acodes(i))
         shp(i) = is
      end do
      print*, 'Read in activity codes '
      close(20)

c     Map of subclasses
      open(33,file='classes_ecr.old')
34    format(a9)
      do i = 1, ncl
         read(33,34) class(i)
         class2(i) = class(i)
      end do
      print*, 'Read in classes '
      close(33)

c     Maps activity codes to subclass
      open(35,file='classmap_ecr.old')
36    format(a9,3x,a4,4(4x,a4))
      do i = 1, nact
         clmap(i) =  0
      end do
      do while (ier.eq.0)
         read(35,36,iostat=ier,end=101) clcode, acin
         do i = 1, nshp2
            j = 0
            if (acin(i).ne.'    ') then
               do iact = 1,nact
```

```
                  if (acodes(iact).eq.acin(i)) then
                     j = iact
                  end if
               end do
               if (j.gt.0) then
                  temp = acin(i)
                  if ((temp(2:2).eq.'6').or.(temp(2:2).eq.'7').or.
     &                 (temp(2:2).eq.'8').or.(temp(1:2).eq.'54')) then
                     clmap(j) = 37
                  else
                     k = 0
                     do icl = 1,ncl
                        if (class2(icl).eq.clcode) then
                           k=icl
                        end if
                     end do
                     if (k.gt.0) then
                        clmap(j) = k
                     else
                        print *,' bad class code = ',clcode,' ',clcode
                     end if
                  end if
               else
                  print *,' activity code not found ',acin(i)
               end if
            end if
         end do
      end do
101 print *,' read exit of classmap = ',ier
      ier = 0
      close(35)

C  Initialize matrices
      do imod = 1, nmod
         do icl = 1, ncl
            do j = 1, nmat
               do is = 1, nshp
                  cdols(j,imod,icl,is) = 0.
               end do
            end do
         end do
      end do

c     Read in distributed cost data
      open(40,file='mods00prc.data')
41    format(10x,8f18.9)

      do imod = 1, nmod
         do iact = 1, nact
            do iw = 1, nw
               read (40,41) (dollars(j,iw,imod,iact),j=1,nmat)
            end do
         end do
      end do

C  Sum data to classes

      do j = 1, nmat
         do imod = 1, nmod
            do iact = 1, nact
               do iw = 1, nw
                  icl = clmap(iact) !  Subclass for corresponding activity code
                  is = shp(iact) !  Assign shape
                  if (icl.eq.2) icl = 1 !  Combine 1SP
                  if (icl.eq.7) icl = 6 !  Combine SP Cards
                  if ((icl.eq.3).or.(icl.eq.4)) icl = 5 !  1st PreL
                  if ((icl.eq.8).or.(icl.eq.9)) icl = 10 !  Pre Cds
                  if (icl.eq.20) icl = 19 !  Std A ECR WSS/WSH
                  if (icl.eq.22) icl = 23 !  Std A Non-ECR
                  if (icl.eq.26) icl = 25 !  Std A NP ECR WSS/WSH
                  if (icl.eq.28) icl = 29 !  Std A NP Non-ECR
                  if (icl.eq.31) icl = 30 !  4th ZPP
                  imod2 = imod
                  if (icl.gt.0) then
                     cdols(j,imod2,icl,is) = cdols(j,imod2,icl,is)
     &                  + dollars(j,iw,imod2,iact)
                  else
                     print *,' activity ',acodes(iact),' not in class map ', iact
                  end if
               end do
```

```
            end do
         end do
      end do

C     Write out to file for comparison with previous process

      open(50,file='mod00cra_prc_ecr.csv')
51    format(i2,',',i2,',',a16,',',a10,',',i2,',',a5,', ',f18.9)

      do icl = 1, ncl
         class3(icl) = class(icl)
      end do

      class3(5) = 'PreL'
      class3(10) = 'PreC'
      class3(19) = 'ECR WSS/H'
      class3(23) = 'BRO'
      class3(25) = 'NECR WSS/H'
      class3(29) = 'NPO'

      do imod = 1, nmod
         do icl = 1, ncl
            do is = 1, nshp
               if ((icl.eq.18).or.(icl.eq.19).or.(icl.eq.21).or.(icl.eq.24).or.
     &             (icl.eq.25).or.(icl.eq.27)) then
                  write (50,51) imod,ldc1(imod), grp(imod),class3(icl), icl, shapetype(is),
     &                cdols(2,imod,icl,is)
               end if
            end do
         end do
      end do

      end

C-------------------------------------------------------------------------------
C     Assign shape

      function shape(act)

      integer*4    shape
      character*4  act

      if (act(1:1).eq.'1') then
         shape = 1  !  Letters
      else if (act(1:1).eq.'2') then
         shape = 2  !  Flats
      else if ((act(1:1).eq.'3').or.(act(1:1).eq.'4')) then
         shape = 3  !  IPPs/Parcels
      else
         shape = 3  !  Other (Special Service)
         if (act.gt.'1000') then
            print*, 'No shape for actv ', act
         end if
      end if

      return
      end
```

```
      program bmcproc00prc_wgt2

c     Purpose:  Computes distributed volume-variable costs (PRC Method) for MODS 1&2 offices
c               Adds additional dimension for various weight categories

      implicit none

      integer*4 nmod, nw, begmod, nshp3, nw2
      integer*4 nact, ishp, nshp, nmix, nmixcl, nact2
      integer*4 nitem, nshp2, ncsi, ncon, begmail

      parameter (nmod = 6)       !  Number of cost pools
      parameter (begmod = 1)     !  Beginning position for BMC cost pools within map
      parameter (nw =  22)       !  Number of weight increments (including no weight)
      parameter (nw2 = 21)       !  Number of weight increments
      parameter (nact = 261)     !  Number of direct activity codes
      parameter (nshp = 6)       !  Number of shapes
      parameter (nitem = 16)     !  Number of item types
      parameter (nshp2 = 5)      !  Number of shapes (not including other)
      parameter (nshp3 = 4)      ! Number of shapes for PRC mixed-mail keys (ishp = letter, flat, parcel, all)
      parameter (ncon = 10)      !  Number of container types
      parameter (nmix = 20)      ! Combined activity codes - for dist of counted items
      parameter (ncsi = nshp2 + nitem) !  Number of "identified" container types (loose shapes + items)
      parameter (begmail = 17)   ! Set this to the index of the first non-Spec Serv activity
      parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
      parameter (nact2 = 281)    ! Number of activity codes including class-specific mixed-mail

      include 'iocs2000.h'

      real*8     adols(nw,nmod,nact2,nshp) ! Handling direct single piece
      real*8    adist(nw,nmod,nact2,nshp) ! Workspace for distribution of no weight single pieces
      real*8     bdols(nw,nmod,nitem,nact2) ! Handling identical or top-piece item
      real*8    bdist(nw,nmod,nitem,nact2) ! Workspace for distribution of no weight identical/top-piece items
      real*8    cdist(nw,nmod,nitem,nact2) ! Workspace for distribution of matrix D
      real*8    ddist(nw,nmod,nact2) ! Workspace for Level 3 D matrix distribution
      real*8    dir9806(nw,nmod,nact2) ! Holds f9806 direct tallies matrix
      real*8     cdols(nw,nmod,nitem,nact2) ! Workspace for distributed costs from matrix D
      real*8     ddols(nmod,nitem) ! Handling mixed/empty item
      real*8     fdols(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8    fdist(nw,nmod,ncon,nact2) ! Workspace for distribution of no weight identical/top-piece containers
      real*8     gdols(nmod,ncon,ncsi) ! Handling "identified" container
      real*8    gdist(nw,nmod,ncon,nact2) ! G Matrix distributed to activity code
      real*8    hdist(nw,nmod,ncon,nact2) ! H Matrix distributed to activity code
      real*8    hkey(nw,nmod,ncon,nact2) ! Matrix to hold distribution key for unidentified container
      real*8     hdols(nmod,ncon) ! Handling uncounted/empty container
      real*8    result(nw,nmod,nact2) ! Array to hold results
      real*8    result2(nw,nmod,nact2) ! Array to hold distribution key data for matrix J
      real*8    resulta(nw,nmod,nact2) ! Array to hold results for matrix A
      real*8    resultb(nw,nmod,nact2) ! Array to hold results for matrix B, C, D
      real*8    resultf(nw,nmod,nact2) ! Array to hold results for matrix F, G, H
      real*8    resultj(nw,nmod,nact2) ! Array to hold distributed J matrix
      real*8    mixkey(nw,nmod,nact2,nshp2) ! Array to hold PRC mixed-mail keys
      real*8    mixdist(nw,nmod,nact2,nshp2) !  Array to hold PRC distributed mixed-mail
      real*8    work(nw,nmod,nact2) ! Array to hold distributed mixed class-specific
      real*8    mixallied(nmod,nshp3) ! Array to hold PRC allied mixed-mail
      real*8    jdols(nmod,nshp3) ! Not Handling
      real*8    counts(ncsi)
      real*8    actshr(nw,nact2), actshr3(nact), actshr4(nw,nact2,nshp3), actwgt(nw2), actshr2(nw,nact2)
      real*8     dlrs, sum, distsum, rf9250, check, deovh6522
      real*8    count1,count2
      real*8    bmix(nmod,nact2)
      real*8    atot, btot, ctot, dtot, ftot, gtot, htot, jtot
      real*8    pooldols(nmod)
      real*8    fixed(nmod),total(nmod),variable(nmod),varprc(nmod)
      real*8    varcost(nw,nmod,nact)
      real*8    dlrsin, dlrsout, dlrs5340in, dlrs5340out
      real*8    novarcst(nw,nmod,nact)
      real*8    dlrs_brk(nmod), dlrs_nobrk(nmod), nowgt_key(nw,nmod,nact2)

      logical flag

      integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
      integer*4 ind, ldc, if9806,iact2
      integer*4 cnt,npl,npnl,counted
      integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw, shapeprc, ishpprc
      integer*4 ier, k, l
      integer*4 mapcodes(20)
      integer*4 searchc, searchi, modgrp, hand, actv, mixshp
      integer*4 mixcodes(nmixcl)
      integer*4 acodes(nact2), mixclass(nact2), mixcount (nmixcl)
```

```
      integer*4 mixmap(nact,nmixcl), class_code(nact2)
      integer*4 ldc1(nmod), class(nact2)
      integer*4 modclass, pool, poolcode(nmod)
      integer*4 if166, if167, weight, ct_nowgt

      character*14 modcodes(nmod)
      character*1 codes(26)/'A','B','C','D','E','F','G','H','I','J','K',
     &   'L','M','N','O','P','Q','R','S','T','U','V',
     &   'W','X','Y','Z'/

      logical flag2

      atot = 0.0
      btot = 0.0
      ctot = 0.0
      dtot = 0.0
      ftot = 0.0
      gtot = 0.0
      htot = 0.0
      jtot = 0.0
      acnt = 0
      bcnt = 0
      ccnt = 0
      dcnt = 0
      fcnt = 0
      gcnt = 0
      hcnt = 0
      jcnt = 0
      cnt = 0
      ier = 0
      count1 = 0.0
      count2 = 0.0
      dlrsin = 0.0
      dlrsout = 0.0
      dlrs5340in = 0.0
      dlrs5340out = 0.0
      npl = 0
      npn1 = 0
      counted =0

c     Factor to exclude activity code 6522 from pool costs
      deovh6522 = 830289./850133.

      do i = 1, nmod
         pooldols(i) = 0.0
         fixed(i) = 0.0
         total(i) = 0.
         variable(i) = 0.
         varprc(i) = 0.
         dlrs_brk(i) = 0.0
         dlrs_nobrk(i) = 0.0
      end do

      do i = 1, 20
         mapcodes(i) = 0
      end do

      do i = 1, nmixcl
         mixcodes(i) = 0
         mixcount(i) = 0
      end do

C     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(i4,i6,i5,i4)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(i), mixclass(i)
      end do
      print *,'read activity map'
      close(20)

c     Map of class specific mixed-mail activity codes
      open(20,file='mixclass.intl')
      do i = 1,nmixcl
         read (20,21) mixcodes(i)
      end do
      print *,'read mixed item code list'
      close(20)

      do i = 1,nact
```

```
         do j = 1,nmixcl
            mixmap(i,j) = 0
         end do
      end do

c     Maps class specific mixed-mail activity codes to appropriate direct activity codes
      open(20,file='mxmail.intl.dat')
23    format(20i4)

      do while (ier.eq.0)
         read (20,23,iostat=ier,end=75) mapcodes
         i = searchi(mixcodes,nmixcl,mapcodes(1))
         if (i.gt.0) then
            flag = .true.
            ind = 1
            do while ((flag).and.(ind.lt.20))
               ind = ind + 1
               if (mapcodes(ind).gt.0) then
                  j = searchi(acodes,nact,mapcodes(ind))
                  if (j.gt.0) then
                     mixcount(i) = mixcount(i) + 1
                     mixmap(mixcount(i),i) = j
                  else
                     print *,' Direct mail code did not map ',mapcodes(ind)
                  end if
               else
                  flag = .false.
               end if
            end do
         else
            print *,' Mixed mail code did not map ',mapcodes(1)
         end if
      end do
75    print *,' read mixed-mail map with exit code = ',ier
      close(20)

c     Map of cost pool dollars and variabilities by cost pool
      open(20,file='costpools.00.bmc.619')
2     format(i4,a14,i5,f9.0,f7.2)
      do i = 1,nmod
         read(20,24) poolcode(i), modcodes(i), ldc1(i), pooldols(i), variable(i)
      end do
      close(20)

C  Initialize matrices

      do iw = 1,nw
         do imod = 1,nmod
            do iact = 1,nact
               varcost(iw,imod,iact) = 0.
               novarcst(iw,imod,iact) = 0.
            end do
         end do
      end do
      do ishp = 1, nshp
         do iact = 1, nact2
            do imod = 1, nmod
               do iw = 1, nw
                  adols(iw,imod,iact,ishp) = 0.0
                  adist(iw,imod,iact,ishp) = 0.0
               end do
            end do
         end do
      end do
      do ishp = 1, nshp2
         do iact = 1, nact2
            do imod = 1, nmod
               do iw = 1, nw
                  mixkey(iw,imod,iact,ishp) = 0.0
                  mixdist(iw,imod,iact,ishp) = 0.0
               end do
            end do
         end do
      end do
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = 1, nmod
               do iw = 1, nw
                  bdols(iw,imod,iitem,iact) = 0.
                  bmix(imod,iact) = 0.0
```

```
            bdist(iw,imod,iitem,iact) = 0.
         end do
      end do
   end do
nd do
do iact = 1, nact2
   do iitem = 1, nitem
      do imod = 1, nmod
         do iw = 1, nw
            cdist(iw,imod,iitem,iact) = 0.
         end do
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         ddist(iw,imod,iact) = 0.
         dir9806(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do iitem = 1, nitem
      do imod = 1, nmod
         do iw=1,nw
            cdols(iw,imod,iitem,iact) = 0.
         end do
      end do
   end do
end do
do iitem = 1, nitem
   do imod = 1, nmod
      ddols(imod,iitem) = 0.
   end do
end do
do iact = 1, nact2
   do icon = 1, ncon
      do imod = 1, nmod
         do iw = 1, nw
            fdols(iw,imod,icon,iact) = 0.
            fdist(iw,imod,icon,iact) = 0.
            hkey(iw,imod,icon,iact) = 0.
         end do
      end do
   end do
end do
do icsi = 1, ncsi
   do icon = 1, ncon
      do imod = 1, nmod
         gdols(imod,icon,icsi) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do icon = 1, ncon
      do imod = 1, nmod
         do iw = 1, nw
            gdist(iw,imod,icon,iact) = 0.
            hdist(iw,imod,icon,iact) = 0.
         end do
      end do
   end do
end do
do icon = 1, ncon
   do imod = 1, nmod
      hdols(imod,icon) = 0.
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
         result(iw,imod,iact) = 0.
         result2(iw,imod,iact) = 0.
      end do
   end do
end do
do iact = 1, nact2
   do imod = 1, nmod
      do iw = 1, nw
```

```
            resulta(iw,imod,iact) = 0.
         end do
      end do
   end do
   lo iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            resultb(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            resultf(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            work(iw,imod,iact) = 0.
            resultj(iw,imod,iact) = 0.
            nowgt_key(iw,imod,iact) = 0.
         end do
      end do
   end do
   do imod = 1, nmod
      do ishp = 1,nshp3
         jdols(imod,ishp) = 0.
         mixallied(imod,ishp) = 0.
      end do
   end do

   print*, 'Matrices initialized '

   open(25,file='bmcs_mp00prc.dat',recl=1200) !  BMCs mail processing IOCS tallies
3  format(a1167,f15.5,i2,i2,i3,i5)
   cnt = 0
   ier = 0
   ct_nowgt = 0

   do while (ier.eq.0)

      read(25,31,iostat=ier,end=100) rec,dlrs,pool,ldc,iw,actv

      cnt = cnt + 1

      iw = 1

      modgrp = searchi(poolcode,nmod,pool) !  Assign cost pool code

      if ((modgrp.lt.begmod).or.(modgrp.gt.nmod)) then
         if (pool.ne.75) then
            print*, 'Invalid cost pool ', modgrp
         end if
         goto 99
      end if

      read(f9250,'(f10.0)') rf9250
      read(f9806,'(i4)') if9806
      read(f166,'(i2)') if166
      read(f167,'(i2)') if167

      dlrs = rf9250/100000.

c     Break out Std A ECR into Basic, Automation, Saturation and High Density activity codes
      if ((if9806.eq.1310).or.(if9806.eq.2310).or.(if9806.eq.3310).or.(if9806.eq.4310)) then !  Reg ECR
         if (f9618.eq.'1') then !  WSH
            if (if9806.eq.1310) if9806=1311
            if (if9806.eq.2310) if9806=2311
            if (if9806.eq.3310) if9806=3311
            if (if9806.eq.4310) if9806=4311
         else if (f9619.eq.'1') then !  WSS
            if (if9806.eq.1310) if9806 = 1313
            if (if9806.eq.2310) if9806 = 2313
            if (if9806.eq.3310) if9806 = 3313
            if (if9806.eq.4310) if9806 = 4313
         else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
            if (if9806.eq.1310) if9806=1312
```

```
            if (if9806.eq.2310) if9806=2312
            if (if9806.eq.3310) if9806=3312
            if (if9806.eq.4310) if9806=4312
         else if (f9617.eq.'1') then !  ECRLOT
            if9806 = if9806
         else
            if9806 = if9806
         end if
      end if
      if ((if9806.eq.1330).or.(if9806.eq.2330).or.(if9806.eq.3330).or.(if9806.eq.4330)) then !  NP ECR
         if (f9618.eq.'1') then !  WSH
            if (if9806.eq.1330) if9806=1331
            if (if9806.eq.2330) if9806=2331
            if (if9806.eq.3330) if9806=3331
            if (if9806.eq.4330) if9806=4331
         else if (f9619.eq.'1') then !  WSS
            if (if9806.eq.1330) if9806 = 1333
            if (if9806.eq.2330) if9806 = 2333
            if (if9806.eq.3330) if9806 = 3333
            if (if9806.eq.4330) if9806 = 4333
         else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
            if (if9806.eq.1330) if9806=1332
            if (if9806.eq.2330) if9806=2332
            if (if9806.eq.3330) if9806=3332
            if (if9806.eq.4330) if9806=4332
         else if (f9617.eq.'1') then !  ECRLOT
            if9806 = if9806
         else
            if9806 = if9806
         end if
      end if

c     Any "auto" ECR flats or parcels are assumed to be basic ECR
      if (if9806.eq.2312) if9806 = 2310
      if (if9806.eq.3312) if9806 = 3310
      if (if9806.eq.4312) if9806 = 4310
      if (if9806.eq.2332) if9806 = 2330
      if (if9806.eq.3332) if9806 = 3330
      if (if9806.eq.4332) if9806 = 4330

c     Identifies tallies for PRC migrated and fixed activity codes
      if ((f9806.eq.'6320').or.(f9806.eq.'6330').or.(f9806.eq.'6430').or.
     &    (f9806.eq.'6460').or.(f9806.eq.'6480').or.(f9806.eq.'6495').or.
     &    (f9806.eq.'6500').or.(f9806.eq.'6511').or.(f9806.eq.'6512').or.
     &    (f9806.eq.'6514').or.(f9806.eq.'6516').or.(f9806.eq.'6519').or.
     &    (f9806.eq.'6610').or.(f9806.eq.'6620').or.(f9806.eq.'6630').or.
     &    (f9806.eq.'6640').or.(f9806.eq.'6650').or.(f9806.eq.'6660').or.
     &    (f9806.eq.'6420').or.(f9806.eq.'6210').or.(f9806.eq.'6220').or.
     &    (f9806.eq.'6230').or.(f9806.eq.'6240')) then
         fixed(modgrp) = fixed(modgrp) + dlrs
         total(modgrp) = total(modgrp) + dlrs
         goto 99
      else
         total(modgrp) = total(modgrp) + dlrs
      end if


      dlrs_brk(modgrp) = dlrs_brk(modgrp) + dlrs
      if (actv.ne.6521) then !  Exclude breaks
         dlrs_nobrk(modgrp) = dlrs_nobrk(modgrp) + dlrs
      end if

c     PRC method uses F9806 for activity code, no encirclement

      actv = if9806

      ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape
      ishpprc = shapeprc(actv) !  Subroutine assigns shape for PRC mixed allieddistribution

c     Assign PRC shape to Intl mixed-mail activity codes
      if ((actv.eq.5434).or.(actv.eq.5444).or.(actv.eq.5454).or.(actv.eq.5464)) then
         ishpprc = 3          !  parcels
      else if ((actv.eq.5433).or.(actv.eq.5443).or.(actv.eq.5453).or.(actv.eq.5463)) then
         ishpprc = 3          !  parcels
      else if ((actv.eq.5432).or.(actv.eq.5442).or.(actv.eq.5452).or.(actv.eq.5462)) then
         ishpprc = 2          !  flats
      else if ((actv.eq.5431).or.(actv.eq.5441).or.(actv.eq.5451).or.(actv.eq.5461)) then
         ishpprc = 1          !  letters
      else if ((actv.eq.5430).or.(actv.eq.5440).or.(actv.eq.5450).or.
     &        (actv.eq.5460).or.(actv.eq.5470).or.(actv.eq.5480)) then
```

```
         ishpprc = 5          !  other
      else if (actv.eq.5340) then
         ishpprc = 5          !  other
      else if ((actv.ge.5300).and.(actv.le.5480)) then
         print *,'actv =',f9806,' in pool=',modgrp
      end if

      iitem = searchc(codes,nitem,f9214) !  Assign item type
      icon = searchc(codes,ncon,f9219) !  Assign container type
      iact = searchi(acodes,nact2,actv) !  Activity codes
      iact2 = searchi(acodes,nact2,if9806) !  Activity codes

      if ((actv.eq.20).or.(actv.eq.60)) then
         print *,'actv=',f9806,' pool=',modgrp,' iitem=',iitem,' $=',dlrs
         if ((actv.eq.60).and.(iitem.gt.0)) then
            fixed(modgrp) = fixed(modgrp) + dlrs
            goto 99
         end if
      end if

      dlrsin = dlrsin + dlrs
      modclass=1

c     Sum direct tally dollar weights by weight increment, cost pool, and F9806 activity code
      if (((if9806.ge.10).and.(if9806.le.4950)).or.((if9806.ge.5300).and.(if9806.le.5480))) then
         if (iact2.gt.0) then
            dir9806(iw,modgrp,iact2) = dir9806(iw,modgrp,iact2) + dlrs ! direct
         else
            print *,'iact2 = 0; f9806 = ',if9806
         end if
      end if

c     Assign handling category
      if (((actv.ge.10).and.(actv.le.4950)).or.((actv.ge.5300).and.(actv.le.5480))) then
         hand = 1             ! direct
      else if ((f9214.ge.'A').and.(f9214.le.'P')) then
         hand = 2             ! mixed item
      else if ((f9219.ge.'A').and.(f9219.le.'J')) then
         hand = 3             ! mixed container
      else
         hand = 4             ! not handling mail
      end if

c        Assigns shape codes for PRC allied mixed-mail and not-handling distribution
      if (actv.eq.5610) then
         mixshp = 1           ! mixed letters
      else if (actv.eq.5620) then
         mixshp = 2           ! mixed flats
      else if (actv.eq.5700) then
         mixshp = 3           ! mixed parcels
      else if ((actv.eq.5750).or.(actv.eq.6523)) then
         mixshp = 4           ! mixed all shapes
      else if (hand.eq.4) then
         mixshp = 4           ! mixed all shapes
      else
         mixshp = 0
      end if

c     Assign weight increment
      if (hand.eq.1) then
         if (actv.ge.1000) then
            iw = weight(f165,if166,if167,ct_nowgt,nw) !  Subroutine assigns weight increment
         else
            iw = nw           !  Special service activities assumed to have no record weight
         end if
      else
         iw = nw
      end if

c        Assigns direct tallies to matrix for PRC mixed and not-handling distribution
      if ((ishpprc.gt.0).and.(ishpprc.le.5)) then
         mixkey(iw,modgrp,iact2,ishpprc) = mixkey(iw,modgrp,iact2,ishpprc) + dlrs
         if (ishpprc.ne.4) then
            mixkey(iw,modgrp,iact2,4) = mixkey(iw,modgrp,iact2,4) + dlrs
         end if
      end if

c     Mixed allied costs for mixed items and containers
      if (((hand.eq.2).or.(hand.eq.3)).and.(modgrp.lt.46)) then ! Allied cost pools only
         if (mixshp.gt.0.) then
```

```
            mixallied(modgrp,mixshp) = mixallied(modgrp,mixshp) + dlrs
         end if
      end if

C     Single piece being handled, Assign to A matrix
      if ((hand.eq.1).and.((iitem.eq.0).and.(icon.eq.0))) then
         if (iact.gt.0) then
            if ((modgrp.gt.0).and.(modgrp.le.nmod)) then
               adols(iw,modgrp,iact,ishp)=adols(iw,modgrp,iact,ishp) + dlrs
               atot = atot + dlrs
               acnt = acnt + 1
            else
               print *,' bad MODS in matrix A ',f114, modgrp, dlrs
            end if
         else
            print *,' bad activity in matrix A ',actv
            if (modgrp.gt.0) then
               jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
               jtot = jtot + dlrs
               jcnt = jcnt + 1
            end if
         end if

C*****************************************************************************
C     Not-handling mail tallies -- assign to J matrix
      else if (hand.eq.4) then
         jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
         jtot = jtot + dlrs
         jcnt = jcnt + 1

C*****************************************************************************
C     Item being handled: separate items with direct activity codes from others

      else if ((f9214.ge.'A').and.(f9214.le.'P')) then
         if (hand.eq.1) then
            imat = 1          ! "B" matrix - identical, top piece, or counted item
         else if (hand.eq.2) then
            imat = 3          ! "D" matrix - mixed, empty item
         else
            print *,'problem item in modgrp = ',modgrp
            imat = 0
         end if

C     "D" matrix: mixed or empty item
         if (imat.eq.3) then
            ddols(modgrp,iitem) = ddols(modgrp,iitem) + dlrs
            dtot = dtot + dlrs
            dcnt = dcnt + 1

C     "B" matrix: identical or top piece rule (direct item)
         else if (imat.eq.1) then
            bdols(iw,modgrp,iitem,iact) =
     &          bdols(iw,modgrp,iitem,iact) + dlrs
            btot = btot + dlrs
            bcnt = bcnt + 1
         else                 !  Dump remaining into not handling
            print *,' imat 0 in modgrp = ',actv
            jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 1
         end if

C*****************************************End Item*****************************
C     Container being handled: separate containers with direct activity codes from others
      else if (icon.gt.0) then

         if (modgrp.gt.0) then

            flag2=.false.

            if (f9901(1:1).eq.'%') then
               read(rec(340:406),451,iostat=ier) counts
            else
               read(rec(339:406),450,iostat=ier) counts
            end if
450         format(5(1x,f3.0),16f3.0)
451         format(f3.0,4(1x,f3.0),16f3.0)

            if (ier.ne.0) then
               flag2 = .true.
```

```
                  j = 340
                  do i = 1, ncsi
                     counts(i) = 0.
                  end do
                  ier = 0
               end if

               sum = 0.
               do i = 1, ncsi
                  sum = sum + counts(i)
               end do

c     "F" matrix: identical mail in container (direct container)

               if (hand.eq.1) then
                  fdols(iw,modgrp,icon,iact) =
     &               fdols(iw,modgrp,icon,iact) + dlrs
                  ftot = ftot + dlrs
                  fcnt = fcnt + 1

c     "H" matrix:  Uncounted, empty, or contents read error
               else if ((sum.eq.0.).or.flag2) then
                  hdols(modgrp,icon) = hdols(modgrp,icon) + dlrs
                  htot = htot + dlrs
                  hcnt = hcnt + 1

c     "G" matrix:  container contents are "identified"
               else if (sum.gt.0.) then
                  do icsi = 1, ncsi
                     gdols(modgrp,icon,icsi) = gdols(modgrp,icon,icsi) +
     &                  (counts(icsi)/sum) * dlrs
                  end do
                  gtot = gtot + dlrs
                  gcnt = gcnt + 1
               end if
            end if

C*****************************************End Container*****************************************
c     Any remaining tallies considered not handling mail
         else
            jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 1
            print *,'bad hand/mat in modgrp = ',modgrp
         end if

99    end do
100   print *,' read exit = ',ier,' with ',cnt,' records '

C ***********************************End Read Loop*********************************
c     Calculate PRC variabilities

      do imod = begmod,nmod
         varprc(imod) = 1 - (fixed(imod)*850133./849454.)/(pooldols(imod)*deovh6522)
         print *,imod,' var prc =',varprc(imod),' var usps=',variable(imod)
      end do

c     Generate a distribution key to distribute no weight tallies over all direct pieces, identical/top piece
C     items, and identical/top piece containers - Only used when all other distribution attempts fail
c     Direct pieces
      do imod = 1, nmod
         do iact = 1, nact2
            do iw = 1, nw2
               do ishp = 1, nshp
                  nowgt_key(iw,imod,iact) = nowgt_key(iw,imod,iact) + adols(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do
c     Identical/top piece items
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw2
               do iitem = 1, nitem
                  nowgt_key(iw,imod,iact) = nowgt_key(iw,imod,iact) + bdols(iw,imod,iitem,iact)
               end do
            end do
         end do
      end do
c     Identical/top piece containers
```

```
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw2
               do icon = 1, ncon
                  nowgt_key(iw,imod,iact) = nowgt_key(iw,imod,iact) + fdols(iw,imod,icon,iact)
               end do
            end do
         end do
      end do

c     Redistribute no weight direct single piece costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + adols(iw,imod,iact,ishp)
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*adols(iw,imod,iact,ishp)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of direct single piece no weight costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distbribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + adols(iw,j,iact,ishp)
                        sum = sum + adols(iw,j,iact,ishp)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  else
                     if (adols(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3 distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = 1, nact2 !  Distribute over all activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + adols(iw,imod,k,ishp)
                                 sum = sum + adols(iw,imod,k,ishp)
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
                                    adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                              end do
                           end do
                           adols(nw,imod,iact,ishp) = 0.0
                        else
                           print*,'Level 4 distribution of act = ',acodes(iact)
                           do k = begmail, nact2
                              do iw = 1, nw2
```

```
                        actshr2(iw,k) = 0.
                     end do
                  end do
                  do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                     if (class(k).eq.class(iact)) then !  Same subclass
                        do j = 1,nmod !  Distribute over all cost pools
                           do iw = 1, nw2 !  Disbribute over all weight increments
                              actshr2(iw,k) = actshr2(iw,k) + adols(iw,j,k,ishp)
                              sum = sum + adols(iw,j,k,ishp)
                           end do
                        end do
                     end if
                  end do
                  if (sum.gt.0.) then
                     do k = begmail, nact2
                        do iw = 1, nw2
                           adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                        adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                        end do
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  else
                     if (ishp.eq.1) then !  assign card directely to < 1/2 oz increment
                        print*, 'Assign card directly to < 1/2 oz increment' !
                        adist(1,imod,iact,ishp) = adist(1,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)
                        adols(nw,imod,iact,ishp) = 0.0
                     else
                        print*,'Level 5 distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class_code(k).eq.class_code(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Disbribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + adols(iw,j,k,ishp)
                                    sum = sum + adols(iw,j,k,ishp)
                                 end do
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                              adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                              end do
                           end do
                           adols(nw,imod,iact,ishp) = 0.0
                        else
                           print*,'unable to distribute no weight for ',
     &                        imod,' act = ',acodes(iact), ' cost = ', adols(nw,imod,iact,ishp)
                        end if
                     end if
                  end if
               end if
            end if
         end if
      end do
   end do
end do

c     Use no wgt key (direct pieces, items, & containers) to distribute remaining piece no weight tallies
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0) then
                  print*, 'a dols nowgt key dist for ', acodes(iact)
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + nowgt_key(iw,imod,iact)
                  end do
                  if (sum.gt.0.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*nowgt_key(iw,imod,iact)/sum
```

```
                  end do
                  adols(nw,imod,iact,ishp) = 0.0
               else
                  print*, 'Level 2 distrib for a, nowgt key ', acodes(iact)
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + nowgt_key(iw,j,iact)
                        sum = sum + nowgt_key(iw,j,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  else
                     if (adols(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3 nowgt_key distribution of a act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Distribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + nowgt_key(iw,j,k)
                                    sum = sum + nowgt_key(iw,j,k)
                                 end do
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                              adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                              end do
                           end do
                           adols(nw,imod,iact,ishp) = 0.0
                        else
                           print*, 'Level 5 b distribution of act = ', acodes(iact)
                           do iw = 1, nw2
                              actwgt(iw) = 0.
                           end do
                           do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                              if (class_code(k).eq.class_code(iact)) then !  Same subclass
                                 do j = 1, nmod !  Distribute over all cost pools
                                    do iw = 1, nw2 !  Disbribute over all weight increments
                                       actwgt(iw) = actwgt(iw) + nowgt_key(iw,j,k)
                                       sum = sum + nowgt_key(iw,j,k)
                                    end do
                                 end do
                              end if
                           end do
                           if (sum.gt.0.) then
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                              adols(nw,imod,iact,ishp) * actwgt(iw) / sum
                              end do
                              adols(nw,imod,iact,ishp) = 0.0
                           else
                              print*,'unable to distribute no weight for a, ',
     &                           imod,' act = ',acodes(iact), ' cost = ', adols(nw,imod,iact,ishp)
                           end if
                        end if
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do
```

```
c     Add in redistributed no weight direct single piece costs
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do ishp = 1, nshp
                  adols(iw,imod,iact,ishp) = adols(iw,imod,iact,ishp) + adist(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

c     Redistribute no weight identical/top piece item costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do iitem = 1, nitem
               if (bdols(nw,imod,iitem,iact).gt.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + bdols(iw,imod,iitem,iact)
                  end do
                  if (sum.gt.0.0) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                     bdols(nw,imod,iitem,iact)*bdols(iw,imod,iitem,iact)/sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of identical/top piece items no weight costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do iitem = 1, nitem
               if (bdols(nw,imod,iitem,iact).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nitem !  Distribute over all item types
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + bdols(iw,imod,j,iact)
                        sum = sum + bdols(iw,imod,j,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                     bdols(nw,imod,iitem,iact)*actwgt(iw)/sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  else
                     if (bdols(nw,imod,iitem,iact).gt.0.0) then
                        print*, 'Level 3 b distribution of act = ', acodes(iact)
                        do iw = 1, nw2
                           actwgt(iw) = 0.
                        end do
                        do k = 1, nmod !  Distribute over all cost pools
                           do j = 1,nitem !  Distribute over all item types
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actwgt(iw) = actwgt(iw) + bdols(iw,k,j,iact)
                                 sum = sum + bdols(iw,k,j,iact)
                              end do
                           end do
                        end do
                        if (sum.gt.0.) then
                           do iw = 1, nw2
                              bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                           bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                           end do
                           bdols(nw,imod,iitem,iact) = 0.0
                        else
                           print*, 'Level 4 b distribution of act = ', acodes(iact)
                           do iw = 1, nw2
                              actwgt(iw) = 0.
                           end do
                           do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                              if (class(k).eq.class(iact)) then !  Same subclass
```

```
                        do j = 1, nmod !  Distribute over all cost pools
                           do l = 1,nitem !  Distribute over all item types
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actwgt(iw) = actwgt(iw) + bdols(iw,j,l,k)
                                 sum = sum + bdols(iw,j,l,k)
                              end do
                           end do
                        end do
                     end if
                  end do
                  if (sum.gt.0.) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                     bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  else
                     print*, 'Level 5 b distribution of act = ', acodes(iact)
                     do iw = 1, nw2
                        actwgt(iw) = 0.
                     end do
                     do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                        if (class_code(k).eq.class_code(iact)) then !  Same subclass
                           do j = 1, nmod !  Distribute over all cost pools
                              do l = 1,nitem !  Distribute over all item types
                                 do iw = 1, nw2 !  Disbribute over all weight increments
                                    actwgt(iw) = actwgt(iw) + bdols(iw,j,l,k)
                                    sum = sum + bdols(iw,j,l,k)
                                 end do
                              end do
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do iw = 1, nw2
                           bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                        bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                        end do
                        bdols(nw,imod,iitem,iact) = 0.0
                     else
                        print*,'unable to distribute no weight for b, ',
     &                     imod,' act = ',acodes(iact), ' cost = ', bdols(nw,imod,iitem,iact)
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c     Final no weight redistribution for identical/top piece items - using no wgt key
      do iact = begmail, nact2
         do imod = 1, nmod
            do iitem = 1, nitem
               if (bdols(nw,imod,iitem,iact).gt.0) then
                  print*, 'b dols nowgt key dist for ', acodes(iact)
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + nowgt_key(iw,imod,iact)
                  end do
                  if (sum.gt.0.0) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,iitem,icon,iact) +
     &                     bdols(nw,imod,iitem,iact)*nowgt_key(iw,imod,iact)/sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  else
                     print*, 'Level 2 distrib for b, nowgt key ', acodes(iact)
                     sum = 0.0
                     do iw = 1, nw2
                        actwgt(iw) = 0.0
                     end do
                     do j = 1, nmod !  Distribute over all cost pools
                        do iw = 1, nw2 !  Distribute over all weight increments
                           actwgt(iw) = actwgt(iw) + nowgt_key(iw,j,iact)
                           sum = sum + nowgt_key(iw,j,iact)
                        end do
                     end do
```

```
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
                           bdols(nw,imod,iitem,iact)*actwgt(iw)/sum
                     end do
                     bdols(nw,imod,iitem,iact) = 0.0
                  else
                     if (bdols(nw,imod,iitem,iact).gt.0.) then
                        print*,'Level 3 nowgt_key distribution of b act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Distribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + nowgt_key(iw,j,k)
                                    sum = sum + nowgt_key(iw,j,k)
                                 end do
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                              bdols(nw,imod,iitem,iact) * actshr2(iw,k) / sum
                              end do
                           end do
                           bdols(nw,imod,iitem,iact) = 0.0
                        else
                           print*, 'Level 5 b distribution of act = ', acodes(iact)
                           do iw = 1, nw2
                              actwgt(iw) = 0.
                           end do
                           do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                              if (class_code(k).eq.class_code(iact)) then !  Same subclass
                                 do j = 1, nmod !  Distribute over all cost pools
                                    do iw = 1, nw2 !  Disbribute over all weight increments
                                       actwgt(iw) = actwgt(iw) + nowgt_key(iw,j,k)
                                       sum = sum + nowgt_key(iw,j,k)
                                    end do
                                 end do
                              end if
                           end do
                           if (sum.gt.0.) then
                              do iw = 1, nw2
                                 bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                              bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                              end do
                              bdols(nw,imod,iitem,iact) = 0.0
                           else
                              print*,'unable to distribute no weight for b, ',
     &                           imod,' act = ',acodes(iact), ' cost = ', bdols(nw,imod,iitem,iact)
                           end if
                        end if
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c     Add in redistributed no weight identical/top piece item costs
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do iitem = 1, nitem
                  bdols(iw,imod,iitem,iact) = bdols(iw,imod,iitem,iact) + bdist(iw,imod,iitem,iact)
                  bdist(iw,imod,iitem,iact) = 0.0
               end do
            end do
         end do
      end do

c     Residual distribution of identical/top piece container no weight costs
      do iact = begmail, nact2
```

```
      do imod = 1, nmod
         do icon = 1, ncon
            if (fdols(nw,imod,icon,iact).gt.0) then
               sum = 0.0
               do iw = 1, nw2
                  sum = sum + fdols(iw,imod,icon,iact)
               end do
               if (sum.gt.0.0) then
                  do iw = 1, nw2 !  Distribute over all weight increments
                     fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                  fdols(nw,imod,icon,iact)*fdols(iw,imod,icon,iact)/sum
                  end do
                  fdols(nw,imod,icon,iact) = 0.0
               end if
            end if
         end do
      end do
   end do

c  Residual distribution of identical/top piece container no weight costs
   do iact = begmail, nact2
      do imod = 1, nmod
         do icon = 1, ncon
            if (fdols(nw,imod,icon,iact).gt.0.0) then
               sum = 0.0
               check = 0.0
               do iw = 1, nw2
                  actwgt(iw) = 0.0
               end do
               do j = 1, nmod !  Distbribute over all cost pools
                  do iw = 1, nw2 !  Distribute over all weight increments
                     actwgt(iw) = actwgt(iw) + fdols(iw,j,icon,iact)
                     sum = sum + fdols(iw,j,icon,iact)
                  end do
               end do
               if (sum.gt.0) then
                  do iw = 1, nw2
                     fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
                        fdols(nw,imod,icon,iact)*actwgt(iw)/sum
                  end do
                  fdols(nw,imod,icon,iact) = 0.0
               else
                  if (fdols(nw,imod,icon,iact).gt.0.) then
                     print*,'Level 3 distribution of f act = ',acodes(iact)
                     do k = begmail, nact2
                        do iw = 1, nw2
                           actshr2(iw,k) = 0.
                        end do
                     end do
                     do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                        if (class(k).eq.class(iact)) then !  Same subclass
                           do iw = 1, nw2 !  Distribute over all weight increments
                              actshr2(iw,k) = actshr2(iw,k) + fdols(iw,imod,icon,k)
                              sum = sum + fdols(iw,imod,icon,k)
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do k = begmail, nact2
                           do iw = 1, nw2
                              fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                           fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                           end do
                        end do
                        fdols(nw,imod,icon,iact) = 0.0
                     else
                        print*,'Level 4 distribution f of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Distribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + fdols(iw,j,icon,k)
                                    sum = sum + fdols(iw,j,icon,k)
                                 end do
                              end do
```

```
                        end if
                     end do
                     if (sum.gt.0.) then
                        do k = begmail, nact2
                           do iw = 1, nw2
                              fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                           fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                           end do
                        end do
                        fdols(nw,imod,icon,iact) = 0.0
                     else
                        print*,'Level 5 distribution f of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class_code(k).eq.class_code(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Distribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + fdols(iw,j,icon,k)
                                    sum = sum + fdols(iw,j,icon,k)
                                 end do
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                              fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                              end do
                           end do
                           fdols(nw,imod,icon,iact) = 0.0
                        end if
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c  Final no weight redistribution for identical/top piece containers - using no wgt key
   do iact = begmail, nact2
      do imod = 1, nmod
         do icon = 1, ncon
            if (fdols(nw,imod,icon,iact).gt.0) then
               print*, 'f dols nowgt key dist for ', acodes(iact)
               sum = 0.0
               do iw = 1, nw2 !  Distribute over all weight increments
                  sum = sum + nowgt_key(iw,imod,iact)
               end do
               if (sum.gt.0.0) then
                  do iw = 1, nw2
                     fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                  fdols(nw,imod,icon,iact)*nowgt_key(iw,imod,iact)/sum
                  end do
                  fdols(nw,imod,icon,iact) = 0.0
               else
                  print*, 'Level 2 distrib for f, nowgt key ', acodes(iact)
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + nowgt_key(iw,j,iact)
                        sum = sum + nowgt_key(iw,j,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                     fdols(nw,imod,icon,iact)*actwgt(iw)/sum
                     end do
                     fdols(nw,imod,icon,iact) = 0.0
                  else
```

```
                  if (fdols(nw,imod,icon,iact).gt.0.) then
                     print*,'Level 3 nowgt_key distribution of f act = ',acodes(iact)
                     do k = begmail, nact2
                        do iw = 1, nw2
                           actshr2(iw,k) = 0.
                        end do
                     end do
                     do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                        if (class(k).eq.class(iact)) then !  Same subclass
                           do j = 1,nmod !  Distribute over all cost pools
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + nowgt_key(iw,j,k)
                                 sum = sum + nowgt_key(iw,j,k)
                              end do
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do k = begmail, nact2
                           do iw = 1, nw2
                              fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                           fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                           end do
                        end do
                        fdols(nw,imod,icon,iact) = 0.0
                     else
                        print*,'unable to distribute no weight f for ',
     &                     imod,' act = ',acodes(iact), ' cost = ', fdols(nw,imod,icon,iact)
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c  Add in redistributed no weight identical/top piece container costs
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            do icon = 1, ncon
               fdols(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + fdist(iw,imod,icon,iact)
               fdist(iw,imod,icon,iact) = 0.0
            end do
         end do
      end do
   end do

c  Distribute no weight tallies for mix key
   do iact = begmail, nact2
      do imod = 1, nmod
         do ishp = 1, nshp2
            if (mixkey(nw,imod,iact,ishp).gt.0.0) then
               sum = 0.0
               do iw = 1, nw2 !  Distribute over all weight increments
                  sum = sum + mixkey(iw,imod,iact,ishp)
               end do
               if (sum.gt.0) then
                  do iw = 1, nw2
                     mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                  mixkey(nw,imod,iact,ishp)*mixkey(iw,imod,iact,ishp)/sum
                  end do
                  mixkey(nw,imod,iact,ishp) = 0.0
               end if
            end if
         end do
      end do
   end do

c  Residual distribution of no weight tallies for mix key
   do iact = begmail, nact2
      do imod = 1, nmod
         do ishp = 1, nshp2
            if (mixkey(nw,imod,iact,ishp).gt.0.0) then
               sum = 0.0
               do iw = 1, nw2
                  actwgt(iw) = 0.0
               end do
               do j = 1, nmod !  Distribute over all cost pools
```

```
            do iw = 1, nw2 !  Distribute over all weight increments
               actwgt(iw) = actwgt(iw) + mixkey(iw,j,iact,ishp)
               sum = sum + mixkey(iw,j,iact,ishp)
            end do
         end do
         if (sum.gt.0) then
            do iw = 1, nw2
               mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &            mixkey(nw,imod,iact,ishp)*actwgt(iw)/sum
            end do
            mixkey(nw,imod,iact,ishp) = 0.0
         else
            if (mixkey(nw,imod,iact,ishp).gt.0.) then
               print*,'Level 3 distribution of act = ',acodes(iact)
               do k = begmail, nact2
                  do iw = 1, nw2
                     actshr2(iw,k) = 0.
                  end do
               end do
               do k = 1, nact2 !  Distribute over all non sp serv activity codes within same subclass
                  if (class(k).eq.class(iact)) then !  Same subclass
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,imod,k,ishp)
                        sum = sum + mixkey(iw,imod,k,ishp)
                     end do
                  end if
               end do
               if (sum.gt.0.) then
                  do k = begmail, nact2
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                     end do
                  end do
                  mixkey(nw,imod,iact,ishp) = 0.0
               else
                  print*,'Level 4 distribution of act = ',acodes(iact)
                  do k = begmail, nact2
                     do iw = 1, nw2
                        actshr2(iw,k) = 0.
                     end do
                  end do
                  do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                     if (class(k).eq.class(iact)) then !  Same subclass
                        do j = 1,nmod !  Distribute over all cost pools
                           do iw = 1, nw2 !  Distribute over all weight increments
                              actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                              sum = sum + mixkey(iw,j,k,ishp)
                           end do
                        end do
                     end if
                  end do
                  if (sum.gt.0.) then
                     do k = begmail, nact2
                        do iw = 1, nw2
                           mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                        mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                        end do
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  else
                     if (ishp.eq.1) then !  assign card directely to < 1/2 oz increment
                        print*, 'Assign card directly to < 1/2 oz increment' !
                        mixdist(1,imod,iact,ishp) = mixdist(1,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp)
                        mixkey(nw,imod,iact,ishp) = 0.0
                     else
                        print*,'Level 5 distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non sp serv activity codes within same subclass
                           if (class_code(k).eq.class_code(iact)) then !  Same subclass
                              do j = 1,nmod !  Distribute over all cost pools
                                 do iw = 1, nw2 !  Distribute over all weight increments
                                    actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                                    sum = sum + mixkey(iw,j,k,ishp)
                                 end do
```

```
                                    end do
                                 end if
                              end do
                              if (sum.gt.0.) then
                                 do k = begmail, nact2
                                    do iw = 1, nw2
                                       mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                                    mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                                    end do
                                 end do
                                 mixkey(nw,imod,iact,ishp) = 0.0
                              else
                                 print*,'unable to distribute no weight for ',
     &                              imod,' act = ',acodes(iact), ' cost = ', mixkey(nw,imod,iact,ishp)
                              end if
                           end if
                        end if
                     end if
                  end if
               end if
            end do
         end do
      end do

c     Add in redistributed no weight tallies for mix key
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do ishp = 1, nshp2
                  mixkey(iw,imod,iact,ishp) = mixkey(iw,imod,iact,ishp) + mixdist(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

c     Redistribution of 53XX and 54XX codes in mixed key for PRC method
      do imod = begmod,nmod
         do iact = 1, nact2

            if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
     &          (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then
               sum = 0.
               do iw = 1,nw
                  do j = 1,nact2
                     do k = 1,3
                        actshr4(iw,j,k) = 0.
                     end do
                  end do
               end do
               if (acodes(iact).eq.5430) then !  Intl mixed U.S. Origin Surface
                  do j = 1,nact2 !  Distribute over all actv codes
                     if (((mixclass(j).ge.18).and.(mixclass(j).le.21)).or.
     &                   ((mixclass(j).ge.32).and.(mixclass(j).le.35))) then !  Intl U.S. Origin Surface direct actvs
                        do iw = 1,nw !  Distribute over all weight increments
                           do k = 1,3 !  Distribute over all PRC shape categories
                              do i = begmod, nmod !  Distribute over all cost pools
                                 sum = sum + mixkey(iw,i,j,k)
                                 actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                              end do
                           end do
                        end do
                     end if
                  end do
               else if (acodes(iact).eq.5440) then !  Intl Mixed U.S. Origin Air Mail
                  do j = 1,nact2 !  Distribute over all actv codes
                     if (((mixclass(j).ge.23).and.(mixclass(j).le.26)).or.
     &                   ((mixclass(j).ge.37).and.(mixclass(j).le.40))) then !  Intl U.S. Origin Air Mail direct actvs
                        do iw = 1,nw !  Distribute over all weight increments
                           do k = 1,3 !  Distribute over all PRC shape categories
                              do i = begmod, nmod !  Distribute over all cost pools
                                 sum = sum + mixkey(iw,i,j,k)
                                 actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                              end do
                           end do
                        end do
                     end if
                  end do
               else if ((acodes(iact).eq.5450).or.(acodes(iact).eq.5470)) then !  Intl mixed Foreign Oridgin Surface
                  do j = 1,nact2 !  Distribute over all actv codes
```

```
                if ((mixclass(j).ge.46).and.(mixclass(j).le.47)) then !  Intl mixed Foreign Oridgin Surface direct actvs
                   do iw = 1,nw !  Distribute over all weight increments
                      do k = 1,3 !  Distribute over all PRC shape categories
                         do i = begmod, nmod !  Distribute over all cost pools
                            sum = sum + mixkey(iw,i,j,k)
                            actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                         end do
                      end do
                   end do
                end if
             end do
          else             ! 5460 & 5480 - Intl mixed Foreign Origin Airmail
             do j = 1,nact2 !  Distribute over all actv codes
                if ((mixclass(j).ge.48).and.(mixclass(j).le.50)) then !  Intl Foreign Origin Airmail direct actvs
                   do iw = 1,nw !  Distribute over all weight increments
                      do k = 1,3 !  Distribute over all PRC shape categories
                         do i = begmod, nmod !  Distribute over all cost pools
                            sum = sum + mixkey(iw,i,j,k)
                            actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                         end do
                      end do
                   end do
                end if
             end do
          end if
          if (sum.gt.0.) then
             do j = 1,nact2
                do iw = 1,nw
                   do k = 1,3
                      mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                   actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                   end do
                end do
             end do
          else
             print*,'cant redistribute code=',acodes(iact),' pool=',imod
          end if
       else if (acodes(iact).eq.5340) then !  Std A Mixed
c      Use across-pool key for handling mixed
          sum = 0.
          do iw = 1,nw
             do j = 1,nact2
                do k = 1,3
                   actshr4(iw,j,k) = 0.
                end do
             end do
          end do
          do j = 1,nact2   !  Distribute over all actv codes
             if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
                do iw = 1,nw !  Distribute over all weight increments
                   do k = 1,3 !  Distribute over all PRC shape categories
                      do i = begmod,nmod !  Distribute over all cost pools
                         sum = sum + mixkey(iw,i,j,k)
                         actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                      end do
                   end do
                end do
             end if
          end do
          if (sum.gt.0.) then
             do j = 1,nact2
                do iw = 1,nw
                   do k = 1,3
                      mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                   actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                   end do
                end do
             end do
          end if
       end if
    end do
 end do

c   Distribute mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
    do imod = begmod, nmod
       if (imod.ne.1) then    !  Excludes Platform
          do iitem = 1, nitem
             if (iitem.ne.9) then !  Exclude green sacks
                if (ddols(imod,iitem).gt.0.) then
                   sum = 0.
```

```
                  do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
                     do iw = 1, nw !  Distribute over all weight increments
                        sum = sum + bdols(iw,imod,iitem,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iact = begmail, nact2
                        do iw = 1, nw
                           cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
     &                        ddols(imod,iitem) * bdols(iw,imod,iitem,iact) / sum
                        end do
                     end do
                     ddols(imod,iitem) = 0.
                  end if
               end if
            end if
         end do
      end if
   end do

C     Distribute remaining mixed/empty item costs ("D" matrix) over all activity codes, DBMC categories,
C     and cost pools within item type using direct item costs ("B" matrix)
      do iitem = 1, nitem
         if (iitem.ne.9) then   !  Exclude green sacks
            do imod = begmod, nmod
               if (ddols(imod,iitem).gt.0.) then
                  sum = 0
                  do iact = 1, nact2
                     do iw = 1, nw
                        actshr(iw,iact) = 0.
                     end do
                  end do
                  do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
                     do j = begmod, nmod !  Distribute over all cost pools
                        do iw = 1, nw !  Distribute over all weight increments
                           actshr(iw,iact) = actshr(iw,iact) + bdols(iw,j,iitem,iact)
                           sum = sum + bdols(iw,j,iitem,iact)
                        end do
                     end do
                  end do
                  if (sum.gt.0.) then
                     do iact = 1, nact2
                        do iw = 1, nw
                           cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
     &                        ddols(imod,iitem) * actshr(iw,iact) / sum
                        end do
                     end do
                     ddols(imod,iitem) = 0.
                  else
                     print *,'L2 unable to dist D dols for iitem = ',iitem,', ',ddols(imod,iitem)
                  end if
               end if
            end do
         else if (ddols(imod,iitem).gt.0.) then
            print *,'Mxd Green sack pool = ',imod,' $ = ',ddols(imod,iitem)
         end if
      end do

C     Sum distributed mixed/empty item costs in "D" distribution matrix
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = begmod, nmod
               do iw = 1, nw
                  ddist(iw,imod,iact) = ddist(iw,imod,iact) + cdist(iw,imod,iitem,iact)
               end do
            end do
         end do
      end do

C     Distribute "identified" container costs ("G" matrix)

      do imod = begmod, nmod

         do icsi = 1, ncsi
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
```

```
      end do
      if (imod.ne.1) then !  Excludes Platform
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  sum = sum + adols(iw,imod,iact,icsi)
                  actshr(iw,iact) = actshr(iw,iact) + adols(iw,imod,iact,icsi)
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  sum = sum + bdols(iw,imod,iitem,iact)
                  actshr(iw,iact) = actshr(iw,iact) + bdols(iw,imod,iitem,iact)
               end do
            end do
         end if
      else                ! Distribute Platform over all ops
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = begmod,nmod !  Distribute over all cost pools
                     sum = sum + adols(iw,i,iact,icsi)
                     actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = begmod,nmod !  Distribute over all cost pools
                     sum = sum + bdols(iw,i,iitem,iact)
                     actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
         end if
      end if
      if (sum.gt.0.) then
         do icon = 1, ncon
            if (gdols(imod,icon,icsi).gt.0.) then
               do iact = 1, nact2
                  do iw = 1, nw
                     gdist(iw,imod,icon,iact) =
     &                  gdist(iw,imod,icon,iact) +
     &                  gdols(imod,icon,icsi) *
     &                  actshr(iw,iact) / sum
                  end do
               end do
            end if
         end do
      else                ! level 2 distribution over all pools
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = begmod,nmod !  Distribute over all cost pools
                     distsum = distsum + adols(iw,i,iact,icsi)
                     actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = begmod,nmod !  Distribute over all cost pools
                     distsum = distsum + bdols(iw,i,iitem,iact)
                     actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
         end if
         if (distsum.gt.0) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do iact = begmail, nact2
                     do iw = 1, nw
                        gdist(iw,imod,icon,iact) =
```

```
     &                          gdist(iw,imod,icon,iact) +
     &                          gdols(imod,icon,icsi) *
     &                          actshr(iw,iact)/distsum
                             end do
                          end do
                       end if
                    end do
                 else
                    print *,'shape distribution empty: mod = ',imod,
     &                 ', shape = ',icsi
                 end if
              end if
           end do
        end do

c       Distributed costs adjustments
        do iact = 1,nact2
           do iw = 1,nw
              do iitem = 1,nitem
                 cdist(iw,1,iitem,iact) = cdist(iw,1,iitem,iact)*36326.0/(36326.0-75.1)
                 cdist(iw,2,iitem,iact) = cdist(iw,2,iitem,iact)*37640.0/(37640.0-393.4-49.5)
              end do
              do icon = 1,ncon
                 gdist(iw,1,icon,iact) = gdist(iw,1,icon,iact)*36326.0/(36326.0-75.1)
                 gdist(iw,2,icon,iact) = gdist(iw,2,icon,iact)*37640.0/(37640.0-393.4-49.5)
              end do
           end do
        end do

c       Sum up distribution factor for distribution of uncounted/empty costs ("H" matrix)
        do iact = 1, nact2
           do icon = 1, ncon
              do imod = begmod, nmod
                 do iw = 1, nw
                    hkey(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + gdist(iw,imod,icon,iact)
                 end do
              end do
           end do
        end do

c       Distribute uncounted/empty containers ("H" matrix) using direct and distributed "identified"
C       container costs
        do imod = begmod, nmod
           do icon = 1, ncon
              sum = 0.
              do iact = 1, nact2  !  Distribute over all activity codes
                 do iw = 1, nw    !  Distribute over all weight increments
                    sum = sum + hkey(iw,imod,icon,iact)
                 end do
              end do
              if (sum.gt.0) then
                 do iact = 1, nact2
                    do iw = 1, nw
                       hdist(iw,imod,icon,iact) = hdist(iw,imod,icon,iact) +
     &                    hdols(imod,icon) * hkey(iw,imod,icon,iact) / sum
                    end do
                 end do
                 hdols(imod,icon) = 0.
              end if
           end do
        end do

c       Distribute remaining uncounted/empty container costs ("H" matrix) using direct/distributed
C       "identified" container costs ("F" matrix)
        do icon = 1, ncon
           do imod = begmod, nmod
              if (hdols(imod,icon).gt.0.) then
                 sum = 0.
                 do iact = 1, nact2
                    do iw = 1, nw
                       actshr(iw,iact) = 0.
                    end do
                 end do
                 do iact = 1, nact2 !  Distribute over all activity codes
                    do j = begmod, nmod !  Distribute over all cost pools
                       do iw = 1, nw !  Distribute over all weight increments
                          actshr(iw,iact) = actshr(iw,iact) + hkey(iw,j,icon,iact)
                          sum = sum + hkey(iw,j,icon,iact)
                       end do
                    end do
```

```
            end do
            if (sum.gt.0.) then
               do iact = 1, nact2
                  do iw = 1, nw
                     hdist(iw,imod,icon,iact) = hdist(iw,imod,icon,iact) +
     &                  actshr(iw,iact)/sum * hdols(imod,icon)
                  end do
               end do
            else
               print *,' unable to dist h dols for imod = ',imod,
     &              ' icon = ',icon
            end if
         end if
      end do
   end do

C     Distribute mixed allied matrix on directs from all pools
      do imod = 1,2             ! Allied cost pools only
         do ishp = 1,nshp3
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            do iact = 1, nact2  !  Distribute over all activity codes
               do iw = 1, nw    !  Distribute over all weight increments
                  do j = begmod,nmod !  Distribute over all cost pools
                     actshr(iw,iact) = actshr(iw,iact) + mixkey(iw,j,iact,ishp)
                     sum = sum + mixkey(iw,j,iact,ishp)
                  end do
               end do
            end do
            if (sum.gt.0) then
               do iact = 1, nact2
                  do iw = 1, nw
                     result(iw,imod,iact) = result(iw,imod,iact) +
     &                  mixallied(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            else
               print *,' unable to distribute J dollars for ',imod
            end if
         end do
      end do

c     Sum up all costs (direct and redistributed) except not handling costs ("J" matrix)
c     Direct tallies
      do iact = 1, nact2
         do imod = begmod, nmod
            do iw = 1, nw
               result2(iw,imod,iact) = result2(iw,imod,iact) + dir9806(iw,imod,iact)
            end do
         end do
      end do
c     Pieces
      do ishp = 1, nshp
         do iact = 1, nact2
            do imod = begmod, nmod
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + adols(iw,imod,iact,ishp)
                  resulta(iw,imod,iact) = resulta(iw,imod,iact) + adols(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do
c     Items
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = begmod, nmod
               do iw = 1, nw
                  if (imod.ge.2) then !  All non-platform pools
                     result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &                  + cdols(iw,imod,iitem,iact)+cdist(iw,imod,iitem,iact)
                     result2(iw,imod,iact) = result2(iw,imod,iact) + cdols(iw,imod,iitem,iact)
     &                  +cdist(iw,imod,iitem,iact)
                     resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &                  + cdols(iw,imod,iitem,iact) +cdist(iw,imod,iitem,iact)
                  else          !  Platform cost pool
```

```
                  result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + cdols(iw,imod,iitem,iact)
     &               +cdist(iw,imod,iitem,iact)
                  resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact) +cdist(iw,imod,iitem,iact)
               end if
            end do
         end do
      end do
   end do
c  Containers
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = begmod, nmod
            if (imod.ge.2) then !  All non-platform pools
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact) + hdist(iw,imod,icon,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + gdist(iw,imod,icon,iact)
     &               + hdist(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)+ hdist(iw,imod,icon,iact)
               end do
            else              !  Platform cost pool
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact)
                  result2(iw,imod,iact) = result2(iw,imod,iact) + gdist(iw,imod,icon,iact)
     &               + hdist(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)+ hdist(iw,imod,icon,iact)
               end do
            end if
         end do
      end do
   end do

   do ishp = 1, nshp2
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               mixkey(iw,imod,iact,ishp) = 0.0
            end do
         end do
      end do
   end do

c  Generate allied not handling keys
   do imod = 1,2              !  Allied cost pools only
      do iw = 1,nw
         do iact = 1,nact2
            if ((acodes(iact).ge.1000).and.(acodes(iact).lt.2000)) then
               mixkey(iw,imod,iact,1) = mixkey(iw,imod,iact,1) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).ge.2000).and.(acodes(iact).lt.3000)) then
               mixkey(iw,imod,iact,2) = mixkey(iw,imod,iact,2) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).ge.3000).and.(acodes(iact).lt.5000)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5434).or.(acodes(iact).eq.5444).or.
     &             (acodes(iact).eq.5454).or.(acodes(iact).eq.5464)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5433).or.(acodes(iact).eq.5443).or.
     &             (acodes(iact).eq.5453).or.(acodes(iact).eq.5463)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5432).or.(acodes(iact).eq.5442).or.
     &             (acodes(iact).eq.5452).or.(acodes(iact).eq.5462)) then
               mixkey(iw,imod,iact,2) = mixkey(iw,imod,iact,2) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5431).or.(acodes(iact).eq.5441).or.
     &             (acodes(iact).eq.5451).or.(acodes(iact).eq.5461)) then
               mixkey(iw,imod,iact,1) = mixkey(iw,imod,iact,1) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
     &             (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then
               mixkey(iw,imod,iact,5) = mixkey(iw,imod,iact,5) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
```

```
            else if (acodes(iact).eq.5340) then
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
               mixkey(iw,imod,iact,5) = mixkey(iw,imod,iact,5) + result(iw,imod,iact)
            end if
         end do
      end do
   end do

c     Redistribution of 53XX and 54XX codes in mixed key for PRC method
      do imod = 1,2
         do iact = 1, nact2
            if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
     &          (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then
            sum = 0.
            do iw = 1,nw
               do j = 1,nact2
                  do k = 1,3
                     actshr4(iw,j,k) = 0.
                  end do
               end do
            end do
            if (acodes(iact).eq.5430) then !  Intl mixed U.S. Origin Surface
               do j = 1,nact2 !  Distribute over all actv codes
                  if (((mixclass(j).ge.18).and.(mixclass(j).le.21)).or.
     &                ((mixclass(j).ge.32).and.(mixclass(j).le.35))) then !  Intl U.S. Origin Surface direct actvs
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod, nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
            else if (acodes(iact).eq.5440) then !  Intl Mixed U.S. Origin Air Mail
               do j = 1,nact2 !  Distribute over all actv codes
                  if (((mixclass(j).ge.23).and.(mixclass(j).le.26)).or.
     &                ((mixclass(j).ge.37).and.(mixclass(j).le.40))) then !  Intl U.S. Origin Air Mail direct actvs
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod, nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
            else if ((acodes(iact).eq.5450).or.(acodes(iact).eq.5470)) then !  Intl mixed Foreign Oridgin Surface
               do j = 1,nact2 !  Distribute over all actv codes
                  if ((mixclass(j).ge.46).and.(mixclass(j).le.47)) then !  Intl mixed Foreign Oridgin Surface direct actvs
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod, nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
            else             ! 5460 & 5480 - Intl mixed Foreign Origin Airmail
               do j = 1,nact2 !  Distribute over all actv codes
                  if ((mixclass(j).ge.48).and.(mixclass(j).le.50)) then !  Intl Foreign Origin Airmail direct actvs
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod, nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
            end if
            if (sum.gt.0.) then
               do j = 1,nact2
                  do iw = 1,nw
                     do k = 1,3
```

```
                        mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                     actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                     end do
                  end do
               end do
            else
               print*,'cant redistribute code=',acodes(iact),' pool=',imod
            end if
         else if (acodes(iact).eq.5340) then !  Std A Mixed
c     Use across-pool key for handling mixed
            sum = 0.
            do iw = 1,nw
               do j = 1,nact2
                  do k = 1,3
                     actshr4(iw,j,k) = 0.
                  end do
               end do
            end do
            do j = 1,nact2   !  Distribute over all actv codes
               if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
                  do iw = 1,nw !  Distribute over all weight increments
                     do k = 1,3 !  Distribute over all PRC shape categories
                        do i = begmod,nmod !  Distribute over all cost pools
                           sum = sum + mixkey(iw,i,j,k)
                           actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                        end do
                     end do
                  end do
               end if
            end do
            if (sum.gt.0) then
               do j = 1,nact2
                  do iw = 1,nw
                     do k = 1,3
                        mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                     actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                     end do
                  end do
               end do
            end if
         end if
      end do
   end do

c     Adds break costs to not-handling
c     Calculated by taking dollar wgts by pool incl breaks (6521) - dollar wgts by pool without breaks

   jdols(4,4) = jdols(4,4) + 36855.0 - 28944.0 !  SSM
   jdols(2,4) = jdols(2,4) + 273603.0 - 231878.0 !  Allied Oth
   jdols(3,4) = jdols(3,4) + 83398.0 - 70176.0 !  PSM
   jdols(5,4) = jdols(5,4) + 79419.0 - 66552.0 !  SPBS
   jdols(6,4) = jdols(6,4) + 44801.0 - 37357.0 !  NMO
   jdols(1,4) = jdols(1,4) + 217263.0 - 188655.0 !  Platform

C     Distribute not handling costs ("J" matrix) using all other costs ("results" matrix)
   do imod = begmod, nmod
      if (imod.ge.2) then    !  Exclude platform cost pool
         sum = 0.
         distsum = 0.
         do iact = 1, nact2
            do iw = 1, nw
               actshr(iw,iact) = 0.
            end do
         end do
         do iact = begmail, nact2 !  Distribute over all non sp serv activity codes
            do iw = 1, nw    !  Distribute over all weight increments
               actshr(iw,iact) = actshr(iw,iact) + result(iw,imod,iact)
               sum = sum + result(iw,imod,iact)
            end do
         end do
         if (sum.gt.0) then
            do ishp = 1,nshp3
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            end do
         else
```

```
            print *,' unable to distribute J dollars for ',imod
         end if
      else                    !  Distribute platform over all cost pools
         do ishp = 1,nshp3
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  actshr(iw,iact) = actshr(iw,iact) + mixkey(iw,imod,iact,ishp)
                  sum = sum + mixkey(iw,imod,iact,ishp)
               end do
            end do
            if (sum.gt.0) then
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * actshr(iw,iact) / sum
                  end do
               end do
            else
               print *,' unable to distribute J dollars for ',imod
            end if
         end do
      end if
   end do

c     Sum distributed not handling costs ("J" matrix) into handling costs ("results" matrix)
      do iact = 1, nact2
         do imod = begmod, nmod
            do iw = 1, nw
               result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
               result2(iw,imod,iact) = result2(iw,imod,iact) + work(iw,imod,iact)
               resultj(iw,imod,iact) = work(iw,imod,iact)
               work(iw,imod,iact) = 0.
            end do
         end do
      end do

c     Redistribute class-specific mixed mail costs over appropriate class-specific direct activity codes,
C     weight increment, and within cost pools
      do imod = begmod,nmod
         do iact = 1,nmixcl
            do iw = 1, nw
               if (result(iw,imod,nact+iact).gt.0.0) then
                  sum = 0.
                  do i = 1,nact
                     actshr3(i) = 0.
                  end do
                  do i = 1,nact !  Distribute over all direct activity codes
                     do j = 1,nw !  Distribute over all weight increments
                        if (mixmap(i,iact).gt.0) then
                           sum = sum + result(j,imod,mixmap(i,iact))
                           actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
     &                        + result(j,imod,mixmap(i,iact))
                        end if
                     end do
                  end do
                  if (sum.gt.0.) then
                     do i = 1,nact
                        if (mixmap(i,iact).gt.0) then
                           work(iw,imod,mixmap(i,iact)) =
     &                        work(iw,imod,mixmap(i,iact)) +
     &                        (result(iw,imod,nact+iact)*
     &                        actshr3(mixmap(i,iact))/sum)
                        end if
                     end do
                     result(iw,imod,nact+iact) = 0.
                  else
                     sum = 0.
                     do i = 1,nact
                        actshr3(i) = 0.
                     end do
                     do i = 1,nact !  Distribute over all direct activity codes
                        do j = 1,nw !  Distribute over all weight increments
                           do k = begmod, nmod !  Distribute over all cost pools
```

```
                        if (mixmap(i,iact).gt.0) then
                           sum = sum + result(j,k,mixmap(i,iact))
                           actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
                              + result(j,k,mixmap(i,iact))
                        end if
                     end do
                  end do
               end do
               if (sum.gt.0.) then
                  do i = 1,nact
                     if (mixmap(i,iact).gt.0) then
                        work(iw,imod,mixmap(i,iact)) =
     &                     work(iw,imod,mixmap(i,iact)) +
     &                     (result(iw,imod,nact+iact)*
     &                     actshr3(mixmap(i,iact))/sum)
                     end if
                  end do
                  result(iw,imod,nact+iact) = 0.
               end if
            end if
         end if
      end do
   end do
end do

c     Sum distributed class-specific mixed-mail costs into all other costs
      do iact = 1, nact
         do imod = begmod, nmod
            do iw = 1, nw
               result2(iw,imod,iact) = result2(iw,imod,iact) + work(iw,imod,iact)
               result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
            end do
         end do
      end do
      do iact = 1, nact
         do imod = begmod, nmod
            do iw = 1, nw
               dlrsout = dlrsout + result(iw,imod,iact)
            end do
         end do
      end do

c     Compute volume-variable costs for all cost pools
      do imod = begmod, nmod
         sum = 0.
         do iact = 1, nact
            do iw = 1, nw
               sum = sum + result(iw,imod,iact)
            end do
         end do
         if (sum.gt.0.) then
            do iact = 1, nact
               do iw = 1, nw
                  varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &               pooldols(imod)*varprc(imod)*deovh6522*result(iw,imod,iact)/sum
                  novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &               pooldols(imod)*deovh6522*result(iw,imod,iact)/sum
               end do
            end do
         else
            print *,'unable to distribute $ = ',pooldols(imod),
     &         ' for mods pool ',modcodes(imod)
         end if
      end do

c     Write out results to file
      open(80,file='bmc00prc.data')
81    format(i3,i4,i3,8f18.9)

      do imod = begmod, nmod
         do iact = 1, nact
            do iw = 1, nw
               write (80,81) ldc1(imod), iact, iw, varcost(iw,imod,iact),
     &            novarcst(iw,imod,iact), result(iw,imod,iact),
     &            resulta(iw,imod,iact), resultb(iw,imod,iact),
     &            resultf(iw,imod,iact), resultj(iw,imod,iact),work(iw,imod,iact)
            end do
         end do
      end do
```

```
      Print *, ' Total Count and Dollars by Matrix '
      write (*,'(2x,a1,i6,f15.2)') 'A', acnt, atot
      write (*,'(2x,a1,i6,f15.2)') 'B', bcnt, btot
      write (*,'(2x,a1,i6,f15.2)') 'C', ccnt, ctot
      write (*,'(2x,a1,i6,f15.2)') 'D', dcnt, dtot
      write (*,'(2x,a1,i6,f15.2)') 'F', fcnt, ftot
      write (*,'(2x,a1,i6,f15.2)') 'G', gcnt, gtot
      write (*,'(2x,a1,i6,f15.2)') 'H', hcnt, htot
      write (*,'(2x,a1,i6,f15.2)') 'J', jcnt, jtot

      print *,'IOCS $ in  = ',dlrsin
      print *,'IOCS $ out = ',dlrsout
      print *,'5340 $ = ',dlrs5340in,' ',dlrs5340out

      end

C -------------------------------------------------------------------------------------------
c     Assign PRC shape

      function shapeprc(actv)

      integer*4 shapeprc,actv
      character*4 f9806

      shapeprc = 0

      if ((actv.lt.1000).or.(actv.ge.5610)) then
         shapeprc = 0      ! special service and mixed-mail
      else if ((actv.ge.1000).and.(actv.lt.2000)) then
         shapeprc = 1      ! letter
      else if ((actv.ge.2000).and.(actv.lt.3000)) then
         shapeprc = 2      ! flat
      else if ((actv.ge.3000).and.(actv.lt.5000)) then
         shapeprc = 3      ! parcel
      else
         shapeprc = 0      ! other?
      end if

      return
      end

C -------------------------------------------------------------------------------------------
c     Assign shape

      function shapeind(actv,f9635,f9805)

      integer*4 shapeind, actv
      character*1  f9635
      character*4 f9805

      if (((actv.ge.1000).and.(actv.lt.2000)).or.(actv.eq.5431).or.(actv.eq.5441)
     &     .or.(actv.eq.5451).or.(actv.eq.5461)) then
          if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
             shapeind = 1         ! cards
          else
             shapeind = 2         ! letters
          end if
       else if (((actv.ge.2000).and.(actv.lt.3000)).or.(actv.eq.5432).or.(actv.eq.5442)
     &     .or.(actv.eq.5452).or.(actv.eq.5462)) then
          shapeind = 3            ! flats
       else if (((actv.ge.3000).and.(actv.lt.4000)).or.(actv.eq.5433).or.(actv.eq.5443)
     &     .or.(actv.eq.5453).or.(actv.eq.5463)) then
          shapeind = 4            ! IPPs
       else if (((actv.ge.4000).and.(actv.lt.5000)).or.(actv.eq.5434).or.(actv.eq.5444)
     &     .or.(actv.eq.5454).or.(actv.eq.5464)) then
          shapeind = 5            ! parcels
       else
          shapeind = 6            ! other?
       end if

       if (actv.eq.5340) then
          shapeind = 6  !  other
          if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
             shapeind = 1         ! cards
          end if
          if (f9635.eq.'A') then
             shapeind = 2  !  letters
          end if
          if ((f9635.eq.'D').or.(f9635.eq.'E')) then
             shapeind = 3  !  flats
```

```
      end if
      if ((f9635.eq.'F').or.(f9635.eq.'G').or.(f9635.eq.'J')) then
         shapeind = 4  !  IPPs
      end if
      if ((f9635.eq.'H').or.(f9635.eq.'I')) then
         shapeind = 5  !  parcels
      end if
   end if

   if ((actv.ge.10).and.(actv.lt.1000)) then
      if ((f9805(1:1).eq.'1').and.(((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K'))) then
         shapeind = 1  !  cards
      else if (f9805(1:1).eq.'1') then
         shapeind = 2  !  letters
      else if (f9805(1:1).eq.'2') then
         shapeind = 3  !  flats
      else if (f9805(1:1).eq.'3') then
         shapeind = 4  !  IPPs
      else if (f9805(1:1).eq.'4') then
         shapeind = 5  !  parcels
      else
         shapeind = 6  !  other
      end if
   end if

   return
   end

c ------------------------------------------------------------------------------------------
c     Assign weight increment

   function weight(f165,if166,if167,ct_nowgt,nw)

   character*1  f165
   integer*4    if166, if167, weight, ct_nowgt, nw

   weight = 0

   if (f165.eq.'A') then
      weight = 1             !  < 1/2 ounce
   else if (f165.eq.'B') then
      weight = 2             !  1 ounces
   else if (f165.eq.'C') then
      weight = 3             !  1 1/2 ounces
   else if (f165.eq.'D') then
      weight = 4             !  2 ounces
   else if (f165.eq.'E') then
      weight = 5             !  2 1/2 ounces
   else if (f165.eq.'F') then
      weight = 6             !  3 ounces
   else if (f165.eq.'G') then
      weight = 7             !  3 1/2 ounces
   else if (f165.eq.'H') then
      weight = 8             !  4 ounces
   else if (f165.eq.'I') then
      if (if166.eq.0) then   !  < 1 lb
         if (if167.gt.0) then
            weight = if167 + 4
         else
            weight = nw
            ct_nowgt = ct_nowgt + 1
         end if
      else if ((if166.eq.1).and.(if167.eq.0)) then
         weight = 20
      else if ((if166.gt.1).or.((if166.eq.1).and.(if167.gt.0))) then
         weight = 21
      else
         weight = nw
         ct_nowgt = ct_nowgt + 1
      end if
   else
      weight = nw
      ct_nowgt = ct_nowgt + 1
   end if

   return
   end
```

```
      program sumclass_bmc_ecr

c     Purpose:  Sum PRC methodology distributed volume-variable mail processing costs
c               for BMCs to subclass
c               Costs are calculated in the Fortran program bmcproc00prc_wgt2.f
c               Breaks out ECR costs by Basic, Automated, Walk Sequence Saturation, and
C               Walk Sequence High Density

      implicit none

      integer*4 nact, ncl, nmod, nshp, nmat, nshp2, nw

      parameter (nmod = 6)       !  Number of cost pools
      parameter (nact = 261)     !  Number of activity codes
      parameter (ncl = 80)       !  Number of subclasses
      parameter (nshp = 3)       !  Number of shapes
      parameter (nmat = 8)       !  Number of cost categories
      parameter (nshp2 = 5)      !  Number of shapes (class map)
      parameter (nw = 22)        !  Number of weight increments

      real*8    dollars(nmat,nw,nmod,nact)
      real*8   cdols(nmat,nmod,ncl,nshp)

      integer*4 imod, iact, icl, i, j, k, shape, is
      integer*4 ier, shp(nact), iw
      integer*4 clmap(nact), mod(nmod), ldc1(nmod)

      character*14  grp(nmod)
      character*9 class(ncl), clcode, class2(ncl)
      character*10 class3(ncl)
      character*4 temp
      character*4 acodes(nact), acin(nshp2)
      character*5 shapetype(nshp)/'1Ltr ','2Flt ','3Pcl '/


      ier = 0

      Map of cost pools
      open(30,file='costpools.00.bmc.619')
32    format(i4,a14,i5)

      do i = 1, nmod
         read(30,32) mod(i), grp(i), ldc1(i)
      end do
      print *, 'BMC groups read'
      close(30)

c     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(a4)

      do i = 1, nact
         read (20,21) acodes(i)
         is = shape(acodes(i))
         shp(i) = is
      end do
      print*, 'Read in activity codes '
      close(20)

c     Map of subclasses
      open(33,file='classes_ecr.old')
34    format(a9)
      do i = 1, ncl
         read(33,34) class(i)
         class2(i) = class(i)
      end do
      print*, 'Read in classes '
      close(33)

c     Maps activity codes to subclass
      open(35,file='classmap_ecr.old')
      format(a9,3x,a4,4(4x,a4))
      do i = 1, nact
         clmap(i) =  0
      end do
      do while (ier.eq.0)
         read(35,36,iostat=ier,end=101) clcode, acin
         do i = 1, nshp2
            j = 0
            if (acin(i).ne.'    ') then
```

```
            do iact = 1,nact
               if (acodes(iact).eq.acin(i)) then
                  j = iact
               end if
            end do
            if (j.gt.0) then
               temp = acin(i)
               if (((temp(2:2).eq.'6').or.(temp(2:2).eq.'7').or.
     &             (temp(2:2).eq.'8').or.(temp(1:2).eq.'54'))) then
                  clmap(j) = 37
               else
                  k = 0
                  do icl = 1,ncl
                     if (class2(icl).eq.clcode) then
                        k=icl
                     end if
                  end do
                  if (k.gt.0) then
                     clmap(j) = k
                  else
                     print *,' bad class code = ',clcode,' ',clcode
                  end if
               end if
            else
               print *,' activity code not found ',acin(i)
            end if
         end if
      end do
   end do
101 print *,' read exit of classmap = ',ier
   ier = 0
   close(35)

C  Initialize matrices
   do imod = 1, nmod
      do icl = 1, ncl
         do j = 1, nmat
            do is = 1, nshp
               cdols(j,imod,icl,is) = 0.
            end do
         end do
      end do
   end do

c     Read in distributed cost data
   open(40,file='bmc00prc.data')
41    format(10x,8f18.9)

   do imod = 1, nmod
      do iact = 1, nact
         do iw = 1, nw
            read (40,41) (dollars(j,iw,imod,iact),j=1,nmat)
         end do
      end do
   end do

C  Sum data to classes

   do j = 1, nmat
      do imod = 1, nmod
         do iact = 1, nact
            do iw = 1, nw
               icl = clmap(iact) !  Subclass for corresponding activity code
               is = shp(iact) !  Assign shape
               if (icl.eq.2) icl = 1 !  Combine 1SP
               if (icl.eq.7) icl = 6 !  Combine SP Cards
               if ((icl.eq.3).or.(icl.eq.4)) icl = 5 !  1st PreL
               if ((icl.eq.8).or.(icl.eq.9)) icl = 10 !  Pre Cds
               if (icl.eq.20) icl = 19 !  Std A ECR WSS/WSH
               if (icl.eq.22) icl = 23 !  Std A Non-ECR
               if (icl.eq.26) icl = 25 !  Std A NP ECR WSS/WSH
               if (icl.eq.28) icl = 29 !  Std A NP Non-ECR
               if (icl.eq.31) icl = 30 !  4th ZPP
               if (icl.gt.0) then
                  cdols(j,imod,icl,is) = cdols(j,imod,icl,is)
     &               + dollars(j,iw,imod,iact)
               else
                  print *,' activity ',acodes(iact),' not in class map '
               end if
            end do
```

```
            end do
         end do
      end do

c     Write out to file for comparison with previous process

      do icl = 1, ncl
         class3(icl) = class(icl)
      end do

      class3(5) = 'PreL'
      class3(10) = 'PreC'
      class3(19) = 'ECR WSS/H'
      class3(23) = 'BRO'
      class3(25) = 'NECR WSS/H'
      class3(29) = 'NPO'

      open(50,file='bmc00cra_prc_ecr.csv')
51    format(i2,',',i2,',',a14,',',a10,',',i2,',',a5,',',f18.9)

      do imod = 1, nmod
         do icl = 1, ncl
            do is = 1, nshp
               if ((icl.eq.18).or.(icl.eq.19).or.(icl.eq.21).or.(icl.eq.24).or.
     &             (icl.eq.25).or.(icl.eq.27)) then
                  write (50,51) imod, ldcl(imod), grp(imod), class3(icl), icl, shapetype(is),
     &                cdols(1,imod,icl,is)
               end if
            end do
         end do
      end do

      end

c-----------------------------------------------------------------------
c     Assign shape

      function shape(act)

      integer*4    shape
      character*4  act

      if (act(1:1).eq.'1') then
         shape = 1  !  Letters
      else if (act(1:1).eq.'2') then
         shape = 2  !  Flats
      else if ((act(1:1).eq.'3').or.(act(1:1).eq.'4')) then
         shape = 3  !  IPPs/Parcels
      else
         shape = 3  !  Other (Special Service)
         if (act.gt.'1000') then
            print*, 'No shape for actv ', act
         end if
      end if

      return
      end
```

```
      program nmodproc00prc_wgt

c     Purpose:  Computes distributed volume-variable costs (PRC Method) for Non-MOD offices
                Adds additional dimension for various weight categories

      implicit none

      integer*4 nmod, nw, begmod, nw2
      integer*4 nact, ishp, nshp, nmix, nmixcl, nact2
      integer*4 nitem, nshp2, ncsi, ncon, begmail
      integer*4 nshp3

      parameter (nmod = 8)       !  Number of cost pools
      parameter (begmod = 1)     !  Begining positionf for Non-MODS cost pools within map
      parameter (nw =  22)       !  Number of weight increments (including no weight)
      parameter (nw2 = 21)       !  Number of weight increments
      parameter (nact = 261)     !  Number of direct activity codes
      parameter (nshp = 6)       !  Number of shapes
      parameter (nitem = 16)     !  Number of item types
      parameter (nshp2 = 5)      !  Number of shapes (not including other)
      parameter (ncon = 10)      !  Number of container types
      parameter (nmix = 20)      ! Combined activity codes - for dist of counted items
      parameter (ncsi = nshp2 + nitem) !  Number of "identified" container types (loose shapes + items)
      parameter (begmail = 18)   ! Set this to the index of the first non-Spec Serv activity
      parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
      parameter (nact2 = 281)    ! Number of activity codes including class-specific mixed-mail
      parameter (nshp3 = 4)      ! Number of shapes for PRC mixed-mail keys (ishp = letter, flat, parcel, all)

      include 'iocs2000.h'

      real*8     adols(nw,nmod,nact2,nshp) ! Handling direct single piece
      real*8    adist(nw,nmod,nact2,nshp) ! Workspace for distribution of no weight single pieces
      real*8     bdols(nw,nmod,nitem,nact2) ! Handling identical or top-piece item
      real*8    bdist(nw,nmod,nitem,nact2) ! Workspace for distribution of no weight identical/top-piece items
      real*8    cdist(nw,nmod,nitem,nact2) ! Workspace for distribution of matrix D
      real*8     cdols(nw,nmod,nitem,nact2) ! Workspace for distributed costs from matrix D
      real*8     ddols(nmod,nitem) ! Handling mixed/empty item
      real*8     fdols(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8    fdist(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8     hkey(nw,nmod,ncon,nact2) ! Handling identical or top-piece container
      real*8     gdols(nmod,ncon,ncsi) ! Handling "identified" container
      real*8    gdist(nw,nmod,ncon,nact2) ! G Matrix distributed to activity code
      real*8     hdols(nmod,ncon) ! Handling uncounted/empty container
      real*8    result(nw,nmod,nact2) ! Array to hold results
      real*8    resulta(nw,nmod,nact2) ! Array to hold results for matrix A
      real*8    resultb(nw,nmod,nact2) ! Array to hold results for matrix B, C, D
      real*8    resultf(nw,nmod,nact2) ! Array to hold results for matrix F, G, H
      real*8    resultj(nw,nmod,nact2) ! Array to hold distributed J matrix
      real*8    work(nw,nmod,nact2) ! Array to hold distributed mixed class-specific
      real*8    jdols(nmod,nshp3) ! Not Handling
      real*8    counts(ncsi)
      real*8    actshr(nw,nact2), actshr3(nact), actwgt(nw2), actshr2(nw,nact2)
      real*8    dlrs, sum, distsum, rf9250, tot_dol, tot_dol2, tot_dol3, check
      real*8    bmix(nmod,nact2)
      real*8    atot, btot, ctot, dtot, ftot, gtot, htot, jtot
      real*8    pooldols(nmod),iocsdols(nmod)
      real*8    variable(nmod)
      real*8    varcost(nw,nmod,nact)
      real*8    novarcst(nw,nmod,nact)
      real*8    tot_fdols, tot_gdist
      real*8    gfy,ovhfact,ovh6522,wgt,wgt6521,wgtall
      real*8    mixkey(nw,nmod,nact2,nshp2) ! Array to hold PRC alliedmixed-mail keys
      real*8    mixallied(nmod,nshp3) ! Array to hold PRC allied mixed-mail
      real*8    mixdist(nw,nmod,nact2,nshp2) !  Array to hold PRC distributed mixed-mail
      real*8    fixed(nmod),varprc(nmod)
      real*8    deovh6522, actshr4(nw,nact2,nshp3)

      logical flag

      integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
      integer*4 ind, ldc, l, pool
      integer*4 cnt,npl,npnl,counted, class(nact2)
      integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw
      integer*4 ier, k, poolcode(nmod), class_code(nact2)
      integer*4 mapcodes(20)
      integer*4 searchc, searchi, modgrp, hand, actv
      integer*4 mixcodes(nmixcl)
      integer*4 acodes(nact2), mixcount(nmixcl)
      integer*4 mixmap(nact,nmixcl)
      integer*4 ldcl(nmod)
```

```
      integer*4 mixclass(nact2), mixshp, shapeprc, ishpprc,if9806
      integer*4 if166, if167, weight, ct_nowgt

      character*14 modcodes(nmod)
      character*1 codes(26)/'A','B','C','D','E','F','G','H','I','J','K',
     &   'L','M','N','O','P','Q','R','S','T','U','V',
     &   'W','X','Y','Z'/

      logical flag2

      atot = 0.0
      btot = 0.0
      ctot = 0.0
      dtot = 0.0
      ftot = 0.0
      gtot = 0.0
      htot = 0.0
      jtot = 0.0
      acnt = 0
      bcnt = 0
      ccnt = 0
      dcnt = 0
      fcnt = 0
      gcnt = 0
      hcnt = 0
      jcnt = 0
      cnt = 0
      npl = 0
      npnl = 0
      counted = 0
      ier = 0
      gfy = 0.
      ovhfact = 0.
      ovh6522 = 0.
      wgt = 0.
      wgt6521 = 0.
      wgtall = 0.
      check = 0.
      distsum = 0.
      tot_fdols = 0.
      tot_gdist = 0.
      tot_dol3 = 0.

      deovh6522 = 830289./850133.

      do i = 1, nmod
         pooldols(i) = 0.0
         fixed(i) = 0.0
         variable(i) = 0.0
         iocsdols(i) = 0.0
         varprc(i) = 0.
      end do

      do i = 1, 20
         mapcodes(i) = 0
      end do

      do i = 1, nmixcl
         mixcodes(i) = 0
         mixcount(i) = 0
      end do

C     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(i4,i6,i5,i4)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(i), mixclass(i)
      end do
      print *,'read activity map'
      close(20)

C     Map of class specific mixed-mail activity codes
      open(20,file='mixclass.intl')
      do i = 1,nmixcl
         read (20,21) mixcodes(i)
      end do
      print *,'read mixed item code list'
      close(20)

      do i = 1,nact
```

```
         do j = 1,nmixcl
            mixmap(i,j) = 0
         end do
      end do

c     Maps class specific mixed-mail activity codes to appropriate direct activity codes
      open(20,file='mxmail.intl.dat')
23    format(20i4)

      do while (ier.eq.0)
         read (20,23,iostat=ier,end=75) mapcodes
         i = searchi(mixcodes,nmixcl,mapcodes(1))
         if (i.gt.0) then
            flag = .true.
            ind = 1
            do while ((flag).and.(ind.lt.20))
               ind = ind + 1
               if (mapcodes(ind).gt.0) then
                  j = searchi(acodes,nact,mapcodes(ind))
                  if (j.gt.0) then
                     mixcount(i) = mixcount(i) + 1
                     mixmap(mixcount(i),i) = j
                  else
                     print *,' Direct mail code did not map ',mapcodes(ind)
                  end if
               else
                  flag = .false.
               end if
            end do
         else
            print *,' Mixed mail code did not map ',mapcodes(1)
         end if
      end do
75    print *,' read mixed-mail map with exit code = ',ier
      close(20)

c     Map of cost pool dollars and variabilities by cost pool
      open(20,file='costpools.00.nmod.619')
      format(i4,a14,i5,f9.0,f7.2)
      do i = 1,nmod
         read(20,24) poolcode(i), modcodes(i), ldc1(i), pooldols(i), variable(i)
      end do
      close(20)

C  Initialize matrices

      do iw = 1,nw
         do imod = 1,nmod
            do iact = 1,nact
               varcost(iw,imod,iact) = 0.
               novarcst(iw,imod,iact) = 0.
            end do
         end do
      end do
      do ishp = 1, nshp
         do iact = 1, nact2
            do imod = 1, nmod
               do iw = 1, nw
                  adols(iw,imod,iact,ishp) = 0.0
                  adist(iw,imod,iact,ishp) = 0.0
               end do
            end do
         end do
      end do
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = 1, nmod
               do iw = 1, nw
                  bdols(iw,imod,iitem,iact) = 0.
                  bdist(iw,imod,iitem,iact) = 0.
                  bmix(imod,iact) = 0.0
               end do
            end do
         end do
      end do
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = 1, nmod
               do iw = 1, nw
                  cdist(iw,imod,iitem,iact) = 0.
```

```
            end do
         end do
      end do
   end do
   do iact = 1, nact2
      do iitem = 1, nitem
         do imod = 1, nmod
            do iw=1,nw
               cdols(iw,imod,iitem,iact) = 0.
            end do
         end do
      end do
   end do
   do iitem = 1, nitem
      do imod = 1, nmod
         ddols(imod,iitem) = 0.
      end do
   end do
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = 1, nmod
            do iw = 1, nw
               fdols(iw,imod,icon,iact) = 0.
               fdist(iw,imod,icon,iact) = 0.
               hkey(iw,imod,icon,iact) = 0.
            end do
         end do
      end do
   end do
   do icsi = 1, ncsi
      do icon = 1, ncon
         do imod = 1, nmod
            gdols(imod,icon,icsi) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = 1, nmod
            do iw = 1, nw
               gdist(iw,imod,icon,iact) = 0.
            end do
         end do
      end do
   end do
   do icon = 1, ncon
      do imod = 1, nmod
         hdols(imod,icon) = 0.
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            result(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            resulta(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            resultb(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
         do iw = 1, nw
            resultf(iw,imod,iact) = 0.
         end do
      end do
   end do
   do iact = 1, nact2
      do imod = 1, nmod
```

```
            do iw = 1, nw
               work(iw,imod,iact) = 0.
               resultj(iw,imod,iact) = 0.
            end do
         end do
      end do
      do imod = 1, nmod
         do ishp = 1,nshp3
            jdols(imod,ishp) = 0.
            mixallied(imod,ishp) = 0.
         end do
      end do
      do ishp = 1, nshp2
         do iact = 1, nact2
            do imod = 1, nmod    ! a matrix
               do iw = 1, nw
                  mixkey(iw,imod,iact,ishp) = 0.
                  mixdist(iw,imod,iact,ishp) = 0.0
               end do
            end do
         end do
      end do
      print*, 'Matrices initialized '

      open(25,file='nonmods_mp00prc.dat',recl=1200) !  Non-MODS mail processing IOCS tallies
31    format(a1167,f15.5,i2,i2,i3,i5)

      cnt = 0
      ier = 0
      tot_dol = 0.0
      tot_dol2 = 0.0
      ct_nowgt = 0

      do while (ier.eq.0)

         read(25,31,iostat=ier,end=100) rec,dlrs,pool,ldc,iw,actv

         cnt = cnt + 1
         iw = 1

         modgrp = searchi(poolcode,nmod,pool) !  Assign cost pool code

         if ((modgrp.lt.begmod).or.(modgrp.gt.nmod)) then
            goto 99
         end if

         read(f9250,'(f10.0)') rf9250
         read(f9806,'(i4)') if9806
         read(f166,'(i2)') if166
         read(f167,'(i2)') if167

c     Break out Std A ECR into Basic, Automation, Saturation and High Density activity codes
         if ((if9806.eq.1310).or.(if9806.eq.2310).or.(if9806.eq.3310).or.(if9806.eq.4310)) then !  Reg ECR
            if (f9618.eq.'1') then !  WSH
               if (if9806.eq.1310) if9806=1311
               if (if9806.eq.2310) if9806=2311
               if (if9806.eq.3310) if9806=3311
               if (if9806.eq.4310) if9806=4311
            else if (f9619.eq.'1') then !  WSS
               if (if9806.eq.1310) if9806 = 1313
               if (if9806.eq.2310) if9806 = 2313
               if (if9806.eq.3310) if9806 = 3313
               if (if9806.eq.4310) if9806 = 4313
            else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
               if (if9806.eq.1310) if9806=1312
               if (if9806.eq.2310) if9806=2312
               if (if9806.eq.3310) if9806=3312
               if (if9806.eq.4310) if9806=4312
            else if (f9617.eq.'1') then !  ECRLOT
               if9806 = if9806
            else
               if9806 = if9806
            end if
         end if
         if ((if9806.eq.1330).or.(if9806.eq.2330).or.(if9806.eq.3330).or.(if9806.eq.4330)) then !  NP ECR
            if (f9618.eq.'1') then !  WSH
               if (if9806.eq.1330) if9806=1331
               if (if9806.eq.2330) if9806=2331
               if (if9806.eq.3330) if9806=3331
               if (if9806.eq.4330) if9806=4331
```

```
            else if (f9619.eq.'1') then !  WSS
               if (if9806.eq.1330) if9806 = 1333
               if (if9806.eq.2330) if9806 = 2333
               if (if9806.eq.3330) if9806 = 3333
               if (if9806.eq.4330) if9806 = 4333
            else if ((f9612.eq.'1').or.(f9613.eq.'1').or.(f9614.eq.'1')) then !  AutoECR
               if (if9806.eq.1330) if9806=1332
               if (if9806.eq.2330) if9806=2332
               if (if9806.eq.3330) if9806=3332
               if (if9806.eq.4330) if9806=4332
            else if (f9617.eq.'1') then !  ECRLOT
               if9806 = if9806
            else
               if9806 = if9806
            end if
         end if

c     Any "auto" ECR flats or parcels are assumed to be basic ECR
         if (if9806.eq.2312) if9806 = 2310
         if (if9806.eq.3312) if9806 = 3310
         if (if9806.eq.4312) if9806 = 4310
         if (if9806.eq.2332) if9806 = 2330
         if (if9806.eq.3332) if9806 = 3330
         if (if9806.eq.4332) if9806 = 4330

c        PRC method uses F9806 for activity code, no encirclement
         actv = if9806

         gfy = 4833549./4402680.
         ovhfact = 2553568./2328659.
         ovh6522 = 4402680./4340556.
         wgt = rf9250/100000.

c        PRC version doesn't use 6522 factor

         dlrs = wgt*gfy*ovhfact

c        Identifies tallies for PRC migrated and fixed activity codes
         if ((f9806.eq.'6320').or.(f9806.eq.'6330').or.(f9806.eq.'6430').or.
     &       (f9806.eq.'6460').or.(f9806.eq.'6480').or.(f9806.eq.'6495').or.
     &       (f9806.eq.'6500').or.(f9806.eq.'6511').or.(f9806.eq.'6512').or.
     &       (f9806.eq.'6514').or.(f9806.eq.'6516').or.(f9806.eq.'6519').or.
     &       (f9806.eq.'6610').or.(f9806.eq.'6620').or.(f9806.eq.'6630').or.
     &       (f9806.eq.'6640').or.(f9806.eq.'6650').or.(f9806.eq.'6660').or.
     &       (f9806.eq.'6420').or.(f9806.eq.'6210').or.(f9806.eq.'6220').or.
     &       (f9806.eq.'6230').or.(f9806.eq.'6240')) then
            fixed(modgrp) = fixed(modgrp) + dlrs
            goto 99
         end if

         if (actv.ne.6521) then
            wgt6521 = wgt6521 + dlrs
            wgtall = wgtall + dlrs
         else
            wgtall = wgtall + dlrs
         end if


         tot_dol = tot_dol + dlrs

c        International mixed codes are remapped to direct codes

         if (actv.eq.5434) then
            actv = 4755
         else if (actv.eq.5444) then
            actv = 4750
         else if (actv.eq.5454) then
            actv = 4810
         else if (actv.eq.5464) then
            actv = 4850
         else if (actv.eq.5433) then
            actv = 3755
         else if (actv.eq.5443) then
            actv = 3750
         else if (actv.eq.5453) then
            actv = 3810
         else if (actv.eq.5463) then
            actv = 3850
         else if (actv.eq.5432) then
            actv = 2755
```

```
      else if (actv.eq.5442) then
         actv = 2750
      else if (actv.eq.5452) then
         actv = 2810
      else if (actv.eq.5462) then
         actv = 2850
      else if (actv.eq.5431) then
         actv = 1755
      else if (actv.eq.5441) then
         actv = 1750
      else if (actv.eq.5451) then
         actv = 1810
      else if (actv.eq.5461) then
         actv = 1850
      end if

      ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape
      ishpprc = shapeprc(actv) !  Subroutine assigns shape for PRC mixed allieddistribution

c     Give class specific mixed mail codes a PRC shape of other
      if ((actv.eq.5430).or.(actv.eq.5440).or.(actv.eq.5450).or.
     &    (actv.eq.5460).or.(actv.eq.5470).or.(actv.eq.5480)) then
         ishpprc = 5
      else if (actv.eq.5340) then
         ishpprc = 5
      else if ((actv.ge.5300).and.(actv.le.5480)) then
         print *,'actv =',f9806,' in pool=',modgrp
      end if

c     Assign handling category
      if (((actv.ge.1000).and.(actv.le.4950)).or.((actv.ge.5300).and.(actv.le.5480))) then
         hand = 1             ! direct (non-special services)
      else if ((actv.ge.10).and.(actv.lt.1000)) then
         if (((f9805.ge.'1000').and.(f9805.le.'4950')).or.
     &       ((f9805(1:2).ge.'53').and.(f9805(1:2).le.'54')))then
            hand = 1          ! direct (special services handling)
         else if ((f9635.ge.'A').and.(f9635.le.'K')) then
            hand = 1          ! direct (special services handling)
         else if ((f9214.ge.'A').and.(f9214.le.'P')) then
            hand = 2          ! mixed item
         else if ((f9219.ge.'A').and.(f9219.le.'J')) then
            hand = 3          ! mixed container
         else
            hand = 4          ! not handling mail
         end if
      else if ((f9214.ge.'A').and.(f9214.le.'P')) then
         hand = 2             ! mixed item
      else if ((f9219.ge.'A').and.(f9219.le.'J')) then
         hand = 3             ! mixed container
      else
         hand = 4             ! not handling mail
      end if

      iitem = searchc(codes,nitem,f9214) !  Assign item type
      icon = searchc(codes,ncon,f9219) !  Assign container type
      iact = searchi(acodes,nact2,actv) !  Activity codes

c     Assign weight increment
      if (hand.eq.1) then
         if (actv.ge.1000) then
            iw = weight(f165,if166,if167,ct_nowgt,nw) !  Subroutine assigns weight increment
         else
            iw = nw           !  Special service activities assumed to have no record weight
         end if
      else
         iw = nw
      end if

c     Assigns direct tallies to matrix for PRC mixed and not-handling distribution

      if ((ishpprc.gt.0).and.(ishpprc.le.5).and.(dlrs.gt.0.)) then
         mixkey(iw,modgrp,iact,ishpprc) = mixkey(iw,modgrp,iact,ishpprc) + dlrs
         if (ishpprc.ne.4) then
            mixkey(iw,modgrp,iact,4) = mixkey(iw,modgrp,iact,4) + dlrs
         end if
      end if

c     Assigns shape codes for PRC allied mixed-mail and not-handling distribution
      if (actv.eq.5610) then
         mixshp = 1           ! mixed letters
```

```
         else if (actv.eq.5620) then
            mixshp = 2          ! mixed flats
         else if (actv.eq.5700) then
            mixshp = 3          ! mixed parcels
         else if ((actv.eq.5750).or.(actv.eq.6523)) then
            mixshp = 4          ! mixed all shapes
         else if (hand.eq.4) then
            mixshp = 4          ! mixed all shapes
         else
            mixshp = 0
         end if

c     Mixed allied costs for mixed items and containers
         if ((hand.eq.2).or.(hand.eq.3)) then
            if (mixshp.gt.0.) then
               mixallied(modgrp,mixshp) = mixallied(modgrp,mixshp) + dlrs
            else
               print *,'mixshp error hand=',hand,' actv=',actv,' pool=',modgrp
            end if
         end if

C     Single piece being handled, Assign to A matrix
         if ((hand.eq.1).and.(((iitem.eq.0).and.(icon.eq.0).and.(f9213.eq.'A'))
     &       .or.(f129.eq.'B'))) then
            if (iact.gt.0) then
               if ((modgrp.gt.0).and.(modgrp.le.nmod)) then
                  adols(iw,modgrp,iact,ishp)=adols(iw,modgrp,iact,ishp) + dlrs
                  atot = atot + dlrs
                  acnt = acnt + 1
                  tot_dol2 = tot_dol2 + dlrs
               else
                  print *,' bad MODS in matrix A ',f114, modgrp, dlrs
               end if
            else
               print *,'Not-handling tally with direct code = ',actv,' cost pool = ',modgrp
               if (modgrp.gt.0) then
                  jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
                  jtot = jtot + dlrs
                  jcnt = jcnt + 1
                  tot_dol2 = tot_dol2 + dlrs
               end if
            end if

C*****************************************************************************
C     Not-handling mail tallies -- assign to J matrix
         else if (hand.eq.4) then

            jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 1
            tot_dol2 = tot_dol2 + dlrs

C*****************************************************************************
C     Item being handled: separate items with direct activity codes from others

         else if ((f9214.ge.'A').and.(f9214.le.'P')) then
            if (hand.eq.1) then
               imat = 1         ! "B"  matrix - identical, top piece, or counted item
            else if (hand.eq.2) then
               imat = 3         ! "D" matrix - mixed, empty item
            else
               print *,'problem item in modgrp = ',modgrp
               imat = 0
            end if

C     "D" matrix: mixed or empty item
            if (imat.eq.3) then
               ddols(modgrp,iitem) = ddols(modgrp,iitem) + dlrs
               dtot = dtot + dlrs
               dcnt = dcnt + 1
               tot_dol2 = tot_dol2 + dlrs

C     "B" matrix: identical or top piece rule (direct item)
            else if (imat.eq.1) then
               bdols(iw,modgrp,iitem,iact) =
     &            bdols(iw,modgrp,iitem,iact) + dlrs
               btot = btot + dlrs
               bcnt = bcnt + 1
               tot_dol2 = tot_dol2 + dlrs
            end if
```

```
C*****************************************End Item*************************
C     Container being handled: separate containers with direct activity codes from others
         else if (icon.gt.0) then

            if (modgrp.gt.0) then

               flag2=.false.

               if (f9901(1:1).eq.'%') then
                  read(rec(340:406),451,iostat=ier) counts
               else
                  read(rec(339:406),450,iostat=ier) counts
               end if
450            format(5(1x,f3.0),16f3.0)
451            format(f3.0,4(1x,f3.0),16f3.0)


               if (ier.ne.0) then
                  flag2 = .true.
                  j = 340
                  do i = 1, ncsi
                     counts(i) = 0.
                  end do
                  ier = 0
               end if

               sum = 0.
               do i = 1, ncsi
                  sum = sum + counts(i)
               end do

c     "F" matrix: identical mail in container (direct container)

               if (hand.eq.1) then
                  fdols(iw,modgrp,icon,iact) =
     &               fdols(iw,modgrp,icon,iact) + dlrs
                  ftot = ftot + dlrs
                  fcnt = fcnt + 1
                  tot_dol2 = tot_dol2 + dlrs

c     "H" matrix:  Uncounted, empty, or contents read error
               else if ((sum.eq.0.).or.flag2) then
                  hdols(modgrp,icon) = hdols(modgrp,icon) + dlrs
                  htot = htot + dlrs
                  hcnt = hcnt + 1
                  tot_dol2 = tot_dol2 + dlrs

c     "G" matrix:  container contents are "identified"
               else if (sum.gt.0.) then
                  do icsi = 1, ncsi
                     gdols(modgrp,icon,icsi) = gdols(modgrp,icon,icsi) +
     &                  (counts(icsi)/sum) * dlrs
                  end do
                  gtot = gtot + dlrs
                  gcnt = gcnt + 1
                  tot_dol2 = tot_dol2 + dlrs
               end if
            end if

C*************************************End Container*****************************
c     Any remaining tallies considered not handling mail
         else

            jdols(modgrp,mixshp) = jdols(modgrp,mixshp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 1
            tot_dol2 = tot_dol2 + dlrs
         end if

99    end do
1     print *,' read exit = ',ier,' with ',cnt,' records ', ' dlrs = ', tot_dol
      print*, 'Total assigned dlrs = ', tot_dol2
      print*, 'Total assigned dlrs alt = ', tot_dol3

C ************************************End Read Loop*****************************
c     Calculate PRC variabilities

      do imod = begmod,nmod
         varprc(imod) = 1 - (fixed(imod)*850133./849454.)/(pooldols(imod)*deovh6522)
```

```
      end do


      Redistribute no weight direct single piece costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + adols(iw,imod,iact,ishp)
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*adols(iw,imod,iact,ishp)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of direct single piece no weight costs

      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp
               if (adols(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distbribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + adols(iw,j,iact,ishp)
                        sum = sum + adols(iw,j,iact,ishp)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                     adols(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     adols(nw,imod,iact,ishp) = 0.0
                  else
                     if (adols(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3a distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + adols(iw,imod,k,ishp)
                                 sum = sum + adols(iw,imod,k,ishp)
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                              adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                              end do
                           end do
                           adols(nw,imod,iact,ishp) = 0.0
                        else
                           print*,'Level 4a distribution of act = ',acodes(iact)
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 actshr2(iw,k) = 0.
                              end do
                           end do
                           do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                              if (class(k).eq.class(iact)) then !  Same subclass
                                 do j = 1,nmod !  Distribute over all cost pools
```

```
                                do iw = 1, nw2 !  Disbribute over all weight increments
                                   actshr2(iw,k) = actshr2(iw,k) + adols(iw,j,k,ishp)
                                   sum = sum + adols(iw,j,k,ishp)
                                end do
                             end do
                          end if
                       end do
                       if (sum.gt.0.) then
                          do k = begmail, nact2
                             do iw = 1, nw2
                                adist(iw,imod,iact,ishp) = adist(iw,imod,iact,ishp) +
     &                             adols(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                             end do
                          end do
                          adols(nw,imod,iact,ishp) = 0.0
                       else
                          print*,'unable to distribute no weight for ',
     &                       imod,' act = ',acodes(iact), ' cost = ', adols(nw,imod,iact,ishp)
                       end if
                    end if
                 end if
              end if
           end do
        end do
     end do

c    Add in redistributed no weight direct single piece costs
     do iact = 1, nact2
        do imod = 1, nmod
           do iw = 1, nw
              do ishp = 1, nshp
                 adols(iw,imod,iact,ishp) = adols(iw,imod,iact,ishp) + adist(iw,imod,iact,ishp)
           end do
        end do
     end do

     Redistribute no weight identical/top piece item costs
     do iact = begmail, nact2
        do imod = 1, nmod
           do iitem = 1, nitem
              if (bdols(nw,imod,iitem,iact).gt.0) then
                 sum = 0.0
                 do iw = 1, nw2 !  Distribute over all weight increments
                    sum = sum + bdols(iw,imod,iitem,iact)
                 end do
                 if (sum.gt.0.0) then
                    do iw = 1, nw2
                       bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                    bdols(nw,imod,iitem,iact)*bdols(iw,imod,iitem,iact)/sum
                    end do
                    bdols(nw,imod,iitem,iact) = 0.0
                 end if
              end if
           end do
        end do
     end do

c    Residual distribution of identical/top piece items no weight costs
     do iact = begmail, nact2
        do imod = 1, nmod
           do iitem = 1, nitem
              if (bdols(nw,imod,iitem,iact).gt.0.0) then
                 sum = 0.0
                 do iw = 1, nw2
                    actwgt(iw) = 0.0
                 end do
                 do j = 1, nitem !  Distribute over all item types
                    do iw = 1, nw2 !  Distribute over all weight increments
                       actwgt(iw) = actwgt(iw) + bdols(iw,imod,j,iact)
                       sum = sum + bdols(iw,imod,j,iact)
                    end do
                 end do
                 if (sum.gt.0) then
                    do iw = 1, nw2
                       bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                    bdols(nw,imod,iitem,iact)*actwgt(iw)/sum
                    end do
                    bdols(nw,imod,iitem,iact) = 0.0
```

```
                else
                   if (bdols(nw,imod,iitem,iact).gt.0.0) then
                      print*, 'Level 3 b distribution of act = ', acodes(iact)
                      do iw = 1, nw2
                         actwgt(iw) = 0.
                      end do
                      do k = 1, nmod !  Distribute over all cost pools
                         do j = 1,nitem !  Distribute over all item types
                            do iw = 1, nw2 !  Distribute over all weight increments
                               actwgt(iw) = actwgt(iw) + bdols(iw,k,j,iact)
                               sum = sum + bdols(iw,k,j,iact)
                            end do
                         end do
                      end do
                      if (sum.gt.0.) then
                         do iw = 1, nw2
                            bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                         bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                         end do
                         bdols(nw,imod,iitem,iact) = 0.0
                      else
                         print*, 'Level 4 b distribution of act = ', acodes(iact)
                         do iw = 1, nw2
                            actwgt(iw) = 0.
                         end do
                         do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                            if (class(k).eq.class(iact)) then !  Same subclass
                               do j = 1, nmod !  Distribute over all cost pools
                                  do l = 1,nitem !  Distribute over all item types
                                     do iw = 1, nw2 !  Disbribute over all weight increments
                                        actwgt(iw) = actwgt(iw) + bdols(iw,j,l,k)
                                        sum = sum + bdols(iw,j,l,k)
                                     end do
                                  end do
                               end do
                            end if
                         end do
                         if (sum.gt.0.) then
                            do iw = 1, nw2
                               bdist(iw,imod,iitem,iact) = bdist(iw,imod,iitem,iact) +
     &                            bdols(nw,imod,iitem,iact) * actwgt(iw) / sum
                            end do
                            bdols(nw,imod,iitem,iact) = 0.0
                         else
                            print*,'unable to distribute no weight for b, ',
     &                         imod,' act = ',acodes(iact), ' cost = ', bdols(nw,imod,iitem,iact)
                         end if
                      end if
                   end if
                end if
             end if
          end do
       end do
    end do

c   Add in redistributed no weight identical/top piece item costs
    do iact = 1, nact2
       do imod = 1, nmod
          do iw = 1, nw
             do iitem = 1, nitem
                bdols(iw,imod,iitem,iact) = bdols(iw,imod,iitem,iact) + bdist(iw,imod,iitem,iact)
                bdist(iw,imod,iitem,iact) = 0.0
             end do
          end do
       end do
    end do

c   Residual distribution of identical/top piece container no weight costs
    do iact = begmail, nact2
       do imod = 1, nmod
          do icon = 1, ncon
             if (fdols(nw,imod,icon,iact).gt.0) then
                sum = 0.0
                do iw = 1, nw2 !  Distribute over all weight increments
                   sum = sum + fdols(iw,imod,icon,iact)
                end do
                if (sum.gt.0.0) then
                   do iw = 1, nw2
                      fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                   fdols(nw,imod,icon,iact)*fdols(iw,imod,icon,iact)/sum
```

```
                  end do
                  fdols(nw,imod,icon,iact) = 0.0
               end if
            end if
         end do
      end do
      end do

c     Residual distribution of identical/top piece container no weight costs
      do iact = begmail, nact2
         do imod = 1, nmod
            do icon = 1, ncon
               if (fdols(nw,imod,icon,iact).gt.0.0) then
                  sum = 0.0
                  check = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distbribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + fdols(iw,j,icon,iact)
                        sum = sum + fdols(iw,j,icon,iact)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                     fdols(nw,imod,icon,iact)*actwgt(iw)/sum
                     end do
                     fdols(nw,imod,icon,iact) = 0.0
                  else
                     if (fdols(nw,imod,icon,iact).gt.0.) then
                        print*,'Level 3 distribution of f act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + fdols(iw,imod,icon,k)
                                 sum = sum + fdols(iw,imod,icon,k)
                              end do
                           end if
                        end do
                        if (sum.gt.0.) then
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                              fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                              end do
                           end do
                           fdols(nw,imod,icon,iact) = 0.0
                        else
                           print*,'Level 4 distribution f of act = ',acodes(iact)
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 actshr2(iw,k) = 0.
                              end do
                           end do
                           do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                              if (mixclass(k).eq.mixclass(iact)) then !  Same subclass
                                 do j = 1,nmod !  Distribute over all cost pools
                                    do iw = 1, nw2 !  Distribute over all weight increments
                                       actshr2(iw,k) = actshr2(iw,k) + fdols(iw,j,icon,iact)
                                       sum = sum + fdols(iw,j,icon,iact)
                                    end do
                                 end do
                              end if
                           end do
                           if (sum.gt.0.) then
                              do k = begmail, nact2
                                 do iw = 1, nw2
                                    fdist(iw,imod,icon,iact) = fdist(iw,imod,icon,iact) +
     &                                 fdols(nw,imod,icon,iact) * actshr2(iw,k) / sum
                                 end do
                              end do
                              fdols(nw,imod,icon,iact) = 0.0
                           else
```

```
                        print*,'unable to distribute no weight f for ',
     &                     imod,' act = ',acodes(iact), ' cost = ', fdols(nw,imod,icon,iact)
                     end if
                  end if
               end if
            end if
         end do
      end do
   end do

c     Add in redistributed no weight identical/top piece container costs
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do icon = 1, ncon
                  fdols(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + fdist(iw,imod,icon,iact)
                  fdist(iw,imod,icon,iact) = 0.0
               end do
            end do
         end do
      end do

c     Redistribution of no weight for mixkey matrix
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp2
               if (mixkey(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2 !  Distribute over all weight increments
                     sum = sum + mixkey(iw,imod,iact,ishp)
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp)*mixkey(iw,imod,iact,ishp)/sum
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  end if
               end if
            end do
         end do
      end do

c     Residual distribution of mixkey matrix no weights
      do iact = begmail, nact2
         do imod = 1, nmod
            do ishp = 1, nshp2
               if (mixkey(nw,imod,iact,ishp).gt.0.0) then
                  sum = 0.0
                  do iw = 1, nw2
                     actwgt(iw) = 0.0
                  end do
                  do j = 1, nmod !  Distribute over all cost pools
                     do iw = 1, nw2 !  Distribute over all weight increments
                        actwgt(iw) = actwgt(iw) + mixkey(iw,j,iact,ishp)
                        sum = sum + mixkey(iw,j,iact,ishp)
                     end do
                  end do
                  if (sum.gt.0) then
                     do iw = 1, nw2
                        mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                     mixkey(nw,imod,iact,ishp)*actwgt(iw)/sum
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  else
                     if (mixkey(nw,imod,iact,ishp).gt.0.) then
                        print*,'Level 3 distribution of act = ',acodes(iact)
                        do k = begmail, nact2
                           do iw = 1, nw2
                              actshr2(iw,k) = 0.
                           end do
                        end do
                        do k = 1, nact2 !  Distribute over all activity codes within same subclass
                           if (class(k).eq.class(iact)) then !  Same subclass
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,imod,k,ishp)
                                 sum = sum + mixkey(iw,imod,k,ishp)
                              end do
                           end if
```

```
                  end do
                  if (sum.gt.0.) then
                     do k = begmail, nact2
                        do iw = 1, nw2
                           mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                        mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                        end do
                     end do
                     mixkey(nw,imod,iact,ishp) = 0.0
                  else
                     print*,'Level 4 distribution of act = ',acodes(iact)
                     do k = begmail, nact2
                        do iw = 1, nw2
                           actshr2(iw,k) = 0.
                        end do
                     end do
                     do k = begmail, nact2 !  Distribute over all non spec serv  activity codes within same subclass
                        if (class(k).eq.class(iact)) then !  Same subclass
                           do j = 1,nmod !  Distribute over all cost pools
                              do iw = 1, nw2 !  Distribute over all weight increments
                                 actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                                 sum = sum + mixkey(iw,j,k,ishp)
                              end do
                           end do
                        end if
                     end do
                     if (sum.gt.0.) then
                        do k = begmail, nact2
                           do iw = 1, nw2
                              mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                           mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                           end do
                        end do
                        mixkey(nw,imod,iact,ishp) = 0.0
                     else
                        if (ishp.eq.1) then !  assign card directely to < 1/2 oz increment
                           print*, 'Assign card directly to < 1/2 oz increment' !
                           mixdist(1,imod,iact,ishp) = mixdist(1,imod,iact,ishp) +
     &                        mixkey(nw,imod,iact,ishp)
                           mixkey(nw,imod,iact,ishp) = 0.0
                        else
                           print*,'Level 5 distribution of act = ',acodes(iact)
                           do k = begmail, nact2
                              do iw = 1, nw2
                                 actshr2(iw,k) = 0.
                              end do
                           end do
                           do k = begmail, nact2 !  Distribute over all non spec serv activity codes within same subclass
                              if (class(k).eq.class(iact)) then !  Same subclass
                                 do j = 1,nmod !  Distribute over all cost pools
                                    do iw = 1, nw2 !  Distribute over all weight increments
                                       actshr2(iw,k) = actshr2(iw,k) + mixkey(iw,j,k,ishp)
                                       sum = sum + mixkey(iw,j,k,ishp)
                                    end do
                                 end do
                              end if
                           end do
                           if (sum.gt.0.) then
                              do k = begmail, nact2
                                 do iw = 1, nw2
                                    mixdist(iw,imod,iact,ishp) = mixdist(iw,imod,iact,ishp) +
     &                                 mixkey(nw,imod,iact,ishp) * actshr2(iw,k) / sum
                                 end do
                              end do
                              mixkey(nw,imod,iact,ishp) = 0.0
                           else
                              print*,'unable to distribute no weight for ',
     &                           imod,' act = ',acodes(iact), ' cost = ', mixkey(nw,imod,iact,ishp)
                           end if
                        end if
                     end if
                  end if
               end if
            end if
         end if
      end do
   end do
end do

c     Add in redistributed no weight mixed mail key costs
```

```
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               do ishp = 1, nshp2
                  mixkey(iw,imod,iact,ishp) = mixkey(iw,imod,iact,ishp) + mixdist(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do

      sum = 0.

c     Redistribution of 53XX and 54XX codes in mixed key for PRC method
      do imod = begmod,nmod
         do iact = 1, nact2

            if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
     &         (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then !  Intl mixed-mail
               sum = 0.
               do iw = 1,nw
                  do j = 1,nact2
                     do k = 1,3
                        actshr4(iw,j,k) = 0.
                     end do
                  end do
               end do
               do j = 1,nact2   !  Distribute over all actv codes
                  if ((mixclass(j).ge.46).and.(mixclass(j).le.50)) then !  Intl direct actv codes
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod,nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
               if (sum.gt.0.) then
                  do j = 1,nact2
                     do iw = 1,nw
                        do k = 1,3
                           mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                        actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                           mixkey(iw,imod,j,4) = mixkey(iw,imod,j,4) +
     &                        actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                        end do
                     end do
                  end do
               else
                  print*,'cant redistribute code=',acodes(iact),' pool=',imod
               end if
            else if (acodes(iact).eq.5340) then !  Std A Mixed
               sum = 0.
               do iw = 1,nw
                  do j = 1,nact2
                     do k = 1,3
                        actshr4(iw,j,k) = 0.
                     end do
                  end do
               end do
               do j = 1,nact2   !  Distribute over all actv codes
                  if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
                     do iw = 1,nw !  Distribute over all weight increments
                        do k = 1,3 !  Distribute over all PRC shape categories
                           do i = begmod,nmod !  Distribute over all cost pools
                              sum = sum + mixkey(iw,i,j,k)
                              actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,i,j,k)
                           end do
                        end do
                     end do
                  end if
               end do
               if (sum.gt.0.0) then
                  do j = 1,nact2
                     do iw = 1,nw
                        do k = 1,3
                           mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
     &                        actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                           mixkey(iw,imod,j,4) = mixkey(iw,imod,j,4) +
```

```
&                        actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                     end do
                  end do
               end do
            else
               print*, 'Unable to distribute mixkey ', imod, acodes(iact)
            end if
         end if
      end do
   end do

c     Distribute mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
      do imod = begmod, nmod
         do iitem = 1, nitem
            if (ddols(imod,iitem).gt.0.) then
               sum = 0.
               do iact = 1, nact2 !  Distribute over all activity codes
                  do iw = 1, nw !  Distribute over all weight increments
                     sum = sum + bdols(iw,imod,iitem,iact)
                  end do
               end do
               if (sum.gt.0) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
&                          ddols(imod,iitem) * bdols(iw,imod,iitem,iact) / sum
                     end do
                  end do
                  ddols(imod,iitem) = 0.
               end if
            end if
         end do
      end do

c     Distribute remaining mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix)
      do iitem = 1, nitem
         do imod = begmod, nmod
            if (ddols(imod,iitem).gt.0.) then
               sum = 0
               do iact = 1, nact2
                  do iw = 1, nw
                     actshr(iw,iact) = 0.
                  end do
               end do
               do iact = 1, nact2 !  Distribute over all activity codes
                  do j = 1, nmod !  Distribute over all cost pools
                     do iw = 1, nw !  Distribute over all weight increments
                        actshr(iw,iact) = actshr(iw,iact) + bdols(iw,j,iitem,iact)
                        sum = sum + bdols(iw,j,iitem,iact)
                     end do
                  end do
               end do
               if (sum.gt.0.) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        cdist(iw,imod,iitem,iact) = cdist(iw,imod,iitem,iact) +
&                          ddols(imod,iitem) * actshr(iw,iact) / sum
                     end do
                  end do
               else
                  print *,' unable to dist D dols for iitem = ',iitem,', ',ddols(imod,iitem)
               end if
            end if
         end do
      end do

C     Distribute "identified" container costs ("G" matrix)

      do imod = begmod, nmod

         do icsi = 1, ncsi
            sum = 0.
            distsum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  actshr(iw,iact) = 0.
               end do
            end do
            if (imod.ne.1) then !  Excludes Platform
               if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
```

```
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  sum = sum + adols(iw,imod,iact,icsi)
                  actshr(iw,iact) = actshr(iw,iact) + adols(iw,imod,iact,icsi)
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw ! Distribute over all weight increments
                  sum = sum + bdols(iw,imod,iitem,iact)
                  actshr(iw,iact) = actshr(iw,iact) + bdols(iw,imod,iitem,iact)
               end do
            end do
         end if
      else                ! Distribute Platform over all ops
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw ! Distribute over all weight increments
                  do i = 1,nmod !  Distribute over all cost pools
                     if (i.lt.7) then !  Exclude Registry & Misc cost pools
                        sum = sum + adols(iw,i,iact,icsi)
                        actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                     end if
                  end do
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = 1,nmod !  Distribute over all cost pools
                     if (i.lt.7) then !  Exclude Registry & Misc cost pools
                        sum = sum + bdols(iw,i,iitem,iact)
                        actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                     end if
                  end do
               end do
            end do
         end if
      end if
      if (sum.gt.0.) then
         do icon = 1, ncon
            if (gdols(imod,icon,icsi).gt.0.) then
               do iact = 1, nact2
                  do iw = 1, nw
                     gdist(iw,imod,icon,iact) =
     &                  gdist(iw,imod,icon,iact) +
     &                  gdols(imod,icon,icsi) *
     &                  actshr(iw,iact) / sum
                     tot_dol = tot_dol + gdols(imod,icon,icsi)*actshr(iw,iact)/sum
                  end do
               end do
            end if
         end do
      else                ! level 2 distribution over all pools
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = 1,nmod !  Distribute over all cost pools
                     distsum = distsum + adols(iw,i,iact,icsi)
                     actshr(iw,iact) = actshr(iw,iact) + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
         else             ! Items distributed upon direct item costs ("B" matrix)
            iitem = icsi - nshp2 !  Distribute over all item types
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  do i = 1,nmod !  Distribute over all cost pools
                     distsum = distsum + bdols(iw,i,iitem,iact)
                     actshr(iw,iact) = actshr(iw,iact) + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
         end if
         if (distsum.gt.0) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do iact = 1, nact2
```

```
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) =
     &                        gdist(iw,imod,icon,iact) +
     &                        gdols(imod,icon,icsi) *
     &                        actshr(iw,iact)/distsum
                           tot_dol = tot_dol + gdols(imod,icon,icsi)*actshr(iw,iact)/distsum
                        end do
                     end do
                  end if
               end do
            else
               do icon = 1, ncon
                  if (gdols(imod,icon,icsi).gt.0.) then
                     print *,'G key empty: mod = ',imod,
     &                  ', shape = ',icsi,gdols(imod,icon,icsi)
                  end if
               end do
            end if
         end if
      end do
   end do                    ! End of "identified" container ("G" matrix) distribution

c  Level 3 distribution of matrices D and G
   do iact = 1,nact2
      do iw = 1,nw
         do iitem = 1,nitem
            cdist(iw,4,iitem,iact) = cdist(iw,4,iitem,iact)*23135.8/(23135.8-218.8)
            cdist(iw,6,iitem,iact) = cdist(iw,6,iitem,iact)*9539.6/(9539.6-942.6)
            cdist(iw,7,iitem,iact) = cdist(iw,7,iitem,iact)*5823.7/(5823.7-199.7)
         end do
         do icon = 1,ncon
            gdist(iw,1,icon,iact) = gdist(iw,1,icon,iact)*52293.4/(52293.4-686.7)
            gdist(iw,4,icon,iact) = gdist(iw,4,icon,iact)*23135.8/(23135.8-218.8)
            gdist(iw,6,icon,iact) = gdist(iw,6,icon,iact)*9539.6/(9539.6-942.6)
            gdist(iw,7,icon,iact) = gdist(iw,7,icon,iact)*5823.7/(5823.7-199.7)
         end do
      end do
   end do

c  Sum direct container costs and distributed "identified container costs for uncounted container distribution
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = begmod, nmod
            do iw = 1, nw
               hkey(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + gdist(iw,imod,icon,iact)
            end do
         end do
      end do
   end do

c  Distribute uncounted/empty containers ("H" matrix) using direct and distributed "identified"
C  container costs
   do imod = begmod, nmod
      do icon = 1, ncon
         sum = 0.
         do iact = 1, nact2   !  Distribute over all activity codes
            do iw = 1, nw     !  Distribute over all weight increments
               sum = sum + hkey(iw,imod,icon,iact)
            end do
         end do
         if (sum.gt.0) then
            do iact = 1, nact2
               do iw = 1, nw
                  gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
     &                hdols(imod,icon) * hkey(iw,imod,icon,iact) / sum
               end do
            end do
            hdols(imod,icon) = 0.
         end if
      end do
   end do

   Distribute remaining uncounted/empty container costs ("H" matrix) using direct/distributed
C  "identified" container costs ("F" matrix)
   do icon = 1, ncon
      do imod = begmod, nmod
         if (hdols(imod,icon).gt.0.) then
            sum = 0.
            do iact = 1, nact2
               do iw = 1, nw
```

```
                  actshr(iw,iact) = 0.
               end do
            end do
            do iact = 1, nact2 !  Distribute over all activity codes
               do j = begmod, nmod !  Distribute over all cost pools
                  do iw = 1, nw !  Distribute over all weight increments
                     actshr(iw,iact) = actshr(iw,iact) + hkey(iw,j,icon,iact)
                     sum = sum + hkey(iw,j,icon,iact)
                  end do
               end do
            end do
            if (sum.gt.0.) then
               do iact = 1, nact2
                  do iw = 1, nw
                     gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
     &                  actshr(iw,iact)/sum * hdols(imod,icon)
                  end do
               end do
            else
               print *,' unable to dist h dols for imod = ',imod,
     &              ' icon = ',icon
            end if
         end if
      end do
   end do

C     Distribute mixed allied matrix on directs from all pools
      do imod = begmod,nmod
         if ((imod.eq.1).or.(imod.eq.8)) then ! Allied and misc pool
            do ishp = 1,nshp3
               print*,imod,ishp,' mix $=',mixallied(imod,ishp)
               sum = 0.
               distsum = 0.
               do iact = 1, nact2
                  do iw = 1, nw
                     actshr(iw,iact) = 0.
                  end do
               end do
               do iact = 1, nact2 !  Distribute over all activity codes
                  do iw = 1, nw !  Distribute over all weight increments
                     do j = begmod,nmod !  Distribute over all cost pools
                        actshr(iw,iact) = actshr(iw,iact) + mixkey(iw,j,iact,ishp)
                        sum = sum + mixkey(iw,j,iact,ishp)
                     end do
                  end do
               end do
               if (sum.gt.0) then
                  do iact = 1, nact2
                     do iw = 1, nw
                        result(iw,imod,iact) = result(iw,imod,iact) +
     &                     mixallied(imod,ishp) * actshr(iw,iact) / sum
                        distsum = distsum +
     &                     mixallied(imod,ishp) * actshr(iw,iact) / sum
                     end do
                  end do
               else
                  print *,' unable to distribute J dollars for ',imod
               end if
               print*,imod,ishp,' dis $=',distsum
            end do
         end if
      end do

c     Sum up all costs (direct and redistributed) except not handling costs ("J" matrix)
c     Pieces
      do ishp = 1, nshp
         do iact = 1, nact2
            do imod = begmod, nmod
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + adols(iw,imod,iact,ishp)
                  resulta(iw,imod,iact) = resulta(iw,imod,iact) + adols(iw,imod,iact,ishp)
               end do
            end do
         end do
      end do
c     Allied and other
      do imod = begmod, nmod
         if ((imod.eq.1).or.(imod.eq.8)) then
c     Items
            do iact = 1, nact2
```

```
            do iitem = 1, nitem
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
                  resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact) + cdist(iw,imod,iitem,iact)
               end do
            end do
         end do
c     Containers
         do iact = 1, nact2
            do icon = 1, ncon
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
               end do
            end do
         end do
      else                    !  All other cost pools
c     Items
         do iact = 1, nact2
            do iitem = 1, nitem
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact) + cdist(iw,imod,iitem,iact)
                  resultb(iw,imod,iact) = resultb(iw,imod,iact) + bdols(iw,imod,iitem,iact)
     &               + cdols(iw,imod,iitem,iact) + cdist(iw,imod,iitem,iact)
               end do
            end do
         end do
c     Containers
         do iact = 1, nact2
            do icon = 1, ncon
               do iw = 1, nw
                  result(iw,imod,iact) = result(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
                  resultf(iw,imod,iact) = resultf(iw,imod,iact) + fdols(iw,imod,icon,iact) +
     &               gdist(iw,imod,icon,iact)
               end do
            end do
         end do
      end if
   end do

   do ishp = 1, nshp2
      do iact = 1, nact2
         do imod = 1, nmod
            do iw = 1, nw
               mixkey(iw,imod,iact,ishp) = 0.0
            end do
         end do
      end do
   end do

c  Generate allied not handling keys
   do imod = begmod,nmod
      do iw = 1,nw
         do iact = 1,nact2
            if ((acodes(iact).ge.1000).and.(acodes(iact).lt.2000)) then
               mixkey(iw,imod,iact,1) = mixkey(iw,imod,iact,1) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).ge.2000).and.(acodes(iact).lt.3000)) then
               mixkey(iw,imod,iact,2) = mixkey(iw,imod,iact,2) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).ge.3000).and.(acodes(iact).lt.5000)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5434).or.(acodes(iact).eq.5444).or.
     &           (acodes(iact).eq.5454).or.(acodes(iact).eq.5464)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5433).or.(acodes(iact).eq.5443).or.
     &           (acodes(iact).eq.5453).or.(acodes(iact).eq.5463)) then
               mixkey(iw,imod,iact,3) = mixkey(iw,imod,iact,3) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5432).or.(acodes(iact).eq.5442).or.
     &           (acodes(iact).eq.5452).or.(acodes(iact).eq.5462)) then
               mixkey(iw,imod,iact,2) = mixkey(iw,imod,iact,2) + result(iw,imod,iact)
               mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
            else if ((acodes(iact).eq.5431).or.(acodes(iact).eq.5441).or.
```

```
&               (acodes(iact).eq.5451).or.(acodes(iact).eq.5461)) then
                mixkey(iw,imod,iact,1) = mixkey(iw,imod,iact,1) + result(iw,imod,iact)
                mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
             else if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
&               (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then
                mixkey(iw,imod,iact,5) = mixkey(iw,imod,iact,5) + result(iw,imod,iact)
                mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
             else if (acodes(iact).eq.5340) then
                mixkey(iw,imod,iact,4) = mixkey(iw,imod,iact,4) + result(iw,imod,iact)
                mixkey(iw,imod,iact,5) = mixkey(iw,imod,iact,5) + result(iw,imod,iact)
             end if
          end do
       end do
    end do

c   Redistribution of 53XX and 54XX codes in mixed key for PRC method
    do imod = begmod,nmod
       do iact = 1, nact2

          if ((acodes(iact).eq.5430).or.(acodes(iact).eq.5440).or.(acodes(iact).eq.5450).or.
&            (acodes(iact).eq.5460).or.(acodes(iact).eq.5470).or.(acodes(iact).eq.5480)) then
             sum = 0.
             do iw = 1,nw
                do j = 1,nact2
                   do k = 1,3
                      actshr4(iw,j,k) = 0.
                   end do
                end do
             end do
             do j = 1,nact2   !  Distribute over all actv codes
                if ((mixclass(j).ge.46).and.(mixclass(j).le.50)) then !  Intl direct actv codes
                   do iw = 1,nw !  Distribute over all weight increments
                      do k = 1,3 !  Distribute over all PRC shape categories
                         sum = sum + mixkey(iw,imod,j,k)
                         actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,imod,j,k)
                      end do
                   end do
                end if
             end do
             do j = 1,nact2
                do iw = 1,nw
                   do k = 1,3
                      mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
&                        actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                   end do
                end do
             end do
          else if (acodes(iact).eq.5340) then !  Std A Mixed
             sum = 0.
             do iw = 1,nw
                do j = 1,nact2
                   do k = 1,3
                      actshr4(iw,j,k) = 0.
                   end do
                end do
             end do
             do j = 1,nact2   !  Distribute over all actv codes
                if ((mixclass(j).ge.10).and.(mixclass(j).le.11)) then !  Std A direct actv codes
                   do iw = 1,nw !  Distribute over all weight increments
                      do k = 1,3 !  Distribute over all PRC shape categories
                         sum = sum + mixkey(iw,imod,j,k)
                         actshr4(iw,j,k) = actshr4(iw,j,k) + mixkey(iw,imod,j,k)
                      end do
                   end do
                end if
             end do
             if (sum.gt.0.) then
                do j = 1,nact2
                   do iw = 1,nw
                      do k = 1,3
                         mixkey(iw,imod,j,k) = mixkey(iw,imod,j,k) +
&                           actshr4(iw,j,k)*mixkey(iw,imod,iact,5)/sum
                      end do
                   end do
                end do
             else
                print *,'unable to dist mixkey pool=',imod,' act=',acodes(iact)
             end if
          end if
       end do
```

```
      end do

C     Distribute not handling costs ("J" matrix) using all other costs ("results" matrix)
      do imod = begmod, nmod
         do ishp = 1,nshp3
            sum = 0.
            distsum = 0.
            do iact = 1, nact2  !  Distribute over all cost pools
               do iw = 1, nw    !  Distribute over all weight increments
                  sum = sum + result(iw,imod,iact)
               end do
            end do
            if (sum.gt.0) then
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod,ishp) * result(iw,imod,iact) / sum
                     distsum = distsum +
     &                  jdols(imod,ishp) * result(iw,imod,iact) / sum

                  end do
               end do
               print *,imod,ishp,distsum
            else
               print *,' unable to distribute J dollars for ',imod
            end if
         end do
      end do

c     Sum distributed not handling costs ("J" matrix) into handling costs ("results" matrix)
      do iact = 1, nact2
         do imod = begmod, nmod
            do iw = 1, nw
               result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
               resultj(iw,imod,iact) = work(iw,imod,iact)
               work(iw,imod,iact) = 0.
            end do
         end do
      end do

c     Redistribute class-specific mixed mail costs over appropriate class-specific direct activity codes,
C     weight increment, and within cost pools
      print *,' distributing mixed-mail items with class-specific codes'
      do imod = 1,nmod
         do iact = 1,nmixcl
            do iw = 1, nw
               if (result(iw,imod,nact+iact).gt.0.0) then
                  sum = 0.
                  do i = 1,nact
                     actshr3(i) = 0.
                  end do
                  do i = 1,nact !  Distribute over all direct activity codes
                     do j = 1,nw !  Distribute over all weight increments
                        if (mixmap(i,iact).gt.0) then
                           sum = sum + result(j,imod,mixmap(i,iact))
                           actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
     &                        + result(j,imod,mixmap(i,iact))
                        end if
                     end do
                  end do
                  if (sum.gt.0.) then
                     do i = 1,nact
                        if (mixmap(i,iact).gt.0) then
                           work(iw,imod,mixmap(i,iact)) =
     &                        work(iw,imod,mixmap(i,iact)) +
     &                        (result(iw,imod,nact+iact)*
     &                        actshr3(mixmap(i,iact))/sum)
                        end if
                     end do
                     result(iw,imod,nact+iact) = 0.
                  else
                     sum = 0.
                     do i = 1,nact
                        actshr3(i) = 0.
                     end do
                     do i = 1, nact !  Distribute over all direct activity codes
                        do j = 1,nw !  Distribute over all weight increments
                           do k = 1, nmod !  Distribute over all cost pools
                              if (mixmap(i,iact).gt.0) then
                                 sum = sum + result(j,k,mixmap(i,iact))
```

```
                              actshr3(mixmap(i,iact)) = actshr3(mixmap(i,iact))
     &                           + result(j,k,mixmap(i,iact))
                            end if
                         end do
                      end do
                   end do
                   if (sum.gt.0.) then
                      do i = 1, nact
                         if (mixmap(i,iact).gt.0) then
                            work(iw,imod,mixmap(i,iact)) =
     &                         work(iw,imod,mixmap(i,iact)) +
     &                         (result(iw,imod,nact+iact)*
     &                         actshr3(mixmap(i,iact))/sum)
                         end if
                      end do
                      result(iw,imod,nact+iact) = 0.
                   else
                      print*, 'Mix actv code not distributed ', acodes(nact+iact),
     &                    ' cost = ', result(iw,imod,nact+iact), ' pool ', modcodes(imod)
                   end if
                end if
             end if
          end do
       end do
    end do

c   Sum distributed class-specific mixed-mail costs into all other costs
    do iact = 1, nact
       do imod = begmod, nmod
          do iw = 1, nw
             result(iw,imod,iact) = result(iw,imod,iact) + work(iw,imod,iact)
             work(iw,imod,iact) = 0.
          end do
       end do
    end do

c   Compute volume-variable costs for all cost pools except Support Fcn 1 & 4
    distsum = 0.
    do imod = begmod, nmod
       sum = 0.
       do iact = 1,nact
          do iw = 1,nw
             sum = sum + result(iw,imod,iact)
          end do
       end do
       if (sum.gt.0.) then
          do iact = 1,nact
             do iw = 1,nw
                varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &             result(iw,imod,iact)*ovhfact
                novarcst(iw,imod,iact) = result(iw,imod,iact)
             end do
          end do
       else
          print *,'unable to distribute $ = ',pooldols(imod),
     &       ' for mods pool ',modcodes(imod)
       end if
    end do

C   Write out result to a file

    open(80,file='nmod00prc.data')
81  format(i3,i4,i3,8f18.9)

    do imod = begmod, nmod
       do iact = 1, nact
          do iw = 1, nw
             write (80,81) ldcl(imod), iact, iw, varcost(iw,imod,iact),
     &          novarcst(iw,imod,iact), result(iw,imod,iact),
     &          resulta(iw,imod,iact), resultb(iw,imod,iact),
     &          resultf(iw,imod,iact), resultj(iw,imod,iact),work(iw,imod,iact)
          end do
       end do
    end do

    Print *, ' Total Count and Dollars by Matrix '
    write (*,'(2x,a1,i6,f15.2)') 'A', acnt, atot
    write (*,'(2x,a1,i6,f15.2)') 'B', bcnt, btot
    write (*,'(2x,a1,i6,f15.2)') 'C', ccnt, ctot
    write (*,'(2x,a1,i6,f15.2)') 'D', dcnt, dtot
```

```
      write (*,'(2x,a1,i6,f15.2)') 'F', fcnt, ftot
      write (*,'(2x,a1,i6,f15.2)') 'G', gcnt, gtot
      write (*,'(2x,a1,i6,f15.2)') 'H', hcnt, htot
      write (*,'(2x,a1,i6,f15.2)') 'J', jcnt, jtot

      print *,'total wgt w/o 6521 = ',wgt6521
      print *,'total wgt inc 6521 = ',wgtall

      end

C -------------------------------------------------------------------------------------------
c     Assign PRC shape

      function shapeprc(actv)

      integer*4 shapeprc,actv

      shapeprc = 0

      if ((actv.lt.1000).or.(actv.ge.5610)) then
         shapeprc = 0      ! special service and mixed-mail
      else if ((actv.ge.1000).and.(actv.lt.2000)) then
         shapeprc = 1      ! letter
      else if ((actv.ge.2000).and.(actv.lt.3000)) then
         shapeprc = 2      ! flat
      else if ((actv.ge.3000).and.(actv.lt.5000)) then
         shapeprc = 3      ! parcel
      else
         shapeprc = 0      ! other?
      end if

      return
      end

C -------------------------------------------------------------------------------------------
c     Assign shape

      function shapeind(actv,f9635,f9805)

      integer*4 shapeind, actv
      character*1  f9635
      character*4 f9805

      if (((actv.ge.1000).and.(actv.lt.2000)).or.(actv.eq.5431).or.(actv.eq.5441)
     &     .or.(actv.eq.5451).or.(actv.eq.5461)) then
         if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
            shapeind = 1         ! cards
         else
            shapeind = 2         ! letters
         end if
      else if (((actv.ge.2000).and.(actv.lt.3000)).or.(actv.eq.5432).or.(actv.eq.5442)
     &     .or.(actv.eq.5452).or.(actv.eq.5462)) then
         shapeind = 3            ! flats
      else if (((actv.ge.3000).and.(actv.lt.4000)).or.(actv.eq.5433).or.(actv.eq.5443)
     &     .or.(actv.eq.5453).or.(actv.eq.5463)) then
         shapeind = 4            ! IPPs
      else if (((actv.ge.4000).and.(actv.lt.5000)).or.(actv.eq.5434).or.(actv.eq.5444)
     &     .or.(actv.eq.5454).or.(actv.eq.5464)) then
         shapeind = 5            ! parcels
      else
         shapeind = 6            ! other?
      end if

      if (actv.eq.5340) then
         shapeind = 6  !  other
         if (((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K')) then
            shapeind = 1         ! cards
         end if
         if (f9635.eq.'A') then
            shapeind = 2   !  letters
         end if
         if ((f9635.eq.'D').or.(f9635.eq.'E')) then
            shapeind = 3  !  flats
         end if
         if ((f9635.eq.'F').or.(f9635.eq.'G').or.(f9635.eq.'J')) then
            shapeind = 4  !  IPPs
         end if
         if ((f9635.eq.'H').or.(f9635.eq.'I')) then
            shapeind = 5  !  parcels
         end if
```

```
      end if

      if ((actv.ge.10).and.(actv.lt.1000)) then
         if ((f9805(1:1).eq.'1').and.(((f9635.ge.'B').and.(f9635.le.'C')).or.(f9635.eq.'K'))) then
            shapeind = 1  !  cards
         else if (f9805(1:1).eq.'1') then
            shapeind = 2  !  letters
         else if (f9805(1:1).eq.'2') then
            shapeind = 3  !  flats
         else if (f9805(1:1).eq.'3') then
            shapeind = 4  !  IPPs
         else if (f9805(1:1).eq.'4') then
            shapeind = 5  !  parcels
         else
            shapeind = 6  !  other
         end if
      end if


      return
      end

c -------------------------------------------------------------------------------------------
c     Assign weight increment

      function weight(f165,if166,if167,ct_nowgt,nw)

      character*1  f165
      integer*4    if166, if167, weight, ct_nowgt, nw

      weight = 0

      if (f165.eq.'A') then
         weight = 1              !  < 1/2 ounce
      else if (f165.eq.'B') then
         weight = 2              !  1 ounces
      else if (f165.eq.'C') then
         weight = 3              !  1 1/2 ounces
      else if (f165.eq.'D') then
         weight = 4              !  2 ounces
      else if (f165.eq.'E') then
         weight = 5              !  2 1/2 ounces
      else if (f165.eq.'F') then
         weight = 6              !  3 ounces
      else if (f165.eq.'G') then
         weight = 7              !  3 1/2 ounces
      else if (f165.eq.'H') then
         weight = 8              !  4 ounces
      else if (f165.eq.'I') then
         if (if166.eq.0) then    !  < 1 lb
            if (if167.gt.0) then
               weight = if167 + 4
            else
               weight = nw
               ct_nowgt = ct_nowgt + 1
            end if
         else if ((if166.eq.1).and.(if167.eq.0)) then ! 1 lb. 0 oz.
            weight = 20
         else if ((if166.gt.1).or.((if166.eq.1).and.(if167.gt.0))) then  ! >= 1 lb. 1 oz.
            weight = 21
         else
            weight = nw
            ct_nowgt = ct_nowgt + 1
         end if
      else
         weight = nw
         ct_nowgt = ct_nowgt + 1
      end if

      return
      end
```

```
      program sumclass_nmod_ecr

c     Purpose:  Sum PRC methodology distributed volume-variable mail processing costs
C               for Non-MOD offices to subclass
c               Costs are calculated in the Fortran program modsproc00prc_wgt2.f
c               Breaks out ECR costs by Basic, Automated, Walk Sequence Saturation, and
C               Walk Sequence High Density

      implicit none

      integer*4 nact, ncl, nmod, nshp, nmat, nshp2, nw

      parameter (nmod = 8)       !  Number of cost pools
      parameter (nact = 261)     !  Number of activity codes
      parameter (ncl = 80)       !  Number of subclasses
      parameter (nshp = 3)       !  Number of shapes
      parameter (nmat = 8)       !  Number of cost categories
      parameter (nshp2 = 5)      !  Number of shapes (class map)
      parameter (nw = 22)        !  Number of weight increments

      real*8    dollars(nmat,nw,nmod,nact)
      real*8   cdols(nmat,nmod,ncl,nshp)

      integer*4 imod, iact, icl, i, j, k, shape, is
      integer*4 ier, shp(nact), iw
      integer*4 clmap(nact), mod(nmod), ldc1(nmod)

      character*14  grp(nmod)
      character*9 class(ncl), clcode
      character*9 class2(ncl)
      character*10 class3(ncl)
      character*4 acodes(nact), temp,acin(nshp2)
      character*5 shapetype(nshp)/'1Ltr ','2Flt ','3Pcl '/


      ier = 0

      Map of cost pools
      open(30,file='costpools.00.nmod.619')
32    format(i4,a14,i5)

      do i = 1, nmod
         read(30,32) mod(i), grp(i), ldc1(i)
      end do
      print *, 'Mod groups read'
      close(30)

c     Map of activity codes
      open(20,file='activity00.ecr.cra2')
21    format(a4)

      do i = 1, nact
         read (20,21) acodes(i)
         is = shape(acodes(i))
         shp(i) = is
      end do
      print*, 'Read in activity codes '
      close(20)

c     Map of subclasses
      open(33,file='classes_ecr.old')
34    format(a9)
      do i = 1, ncl
         read(33,34) class(i)
         class2(i) = class(i)
      end do
      print*, 'Read in classes '
      close(33)

c     Maps activity codes to subclass
      open(35,file='classmap_ecr.old')
      format(a9,3x,a4,4(4x,a4))
      do i = 1, nact
         clmap(i) =  0
      end do
      do while (ier.eq.0)
         read(35,36,iostat=ier,end=101) clcode, acin
         do i = 1, nshp2
            j = 0
            if (acin(i).ne.'    ') then
```

```
            do iact = 1,nact
               if (acodes(iact).eq.acin(i)) then
                  j = iact
               end if
            end do
            if (j.gt.0) then
               temp = acin(i)
               if (((temp(2:2).eq.'6').or.(temp(2:2).eq.'7').or.
     &             (temp(2:2).eq.'8').or.(temp(1:2).eq.'54'))) then
                  clmap(j) = 37
               else
                  k = 0
                  do icl = 1,ncl
                     if (class2(icl).eq.clcode) then
                        k=icl
                     end if
                  end do
                  if (k.gt.0) then
                     clmap(j) = k
                  else
                     print *,' bad class code = ',clcode,' ',clcode
                  end if
               end if
            else
               print *,' activity code not found ',acin(i)
            end if
         end if
      end do
   end do
101 print *,' read exit of classmap = ',ier
   ier = 0
   close(35)

C  Initialize matrices

   do imod = 1, nmod
      do icl = 1, ncl
         do j = 1, nmat
            do is = 1, nshp
               cdols(j,imod,icl,is) = 0.
            end do
         end do
      end do
   end do

c     Read in distributed cost data
   open(40,file='nmod00prc.data')
41    format(10x,8f18.9)

   do imod = 1, nmod
      do iact = 1, nact
         do iw = 1, nw
            read (40,41) (dollars(j,iw,imod,iact),j=1,nmat)
         end do
      end do
   end do

C  Sum data to classes

   do j = 1, nmat
      do imod = 1, nmod
         do iact = 1, nact
            do iw = 1, nw
               icl = clmap(iact) !  Subclass for corresponding activity code
               is = shp(iact) !  Assign shape
               if (icl.eq.2) icl = 1 !  Combine 1SP
               if (icl.eq.7) icl = 6 !  Combine SP Cards
               if ((icl.eq.3).or.(icl.eq.4)) icl = 5 !  1st PreL
               if ((icl.eq.8).or.(icl.eq.9)) icl = 10 !  Pre Cds
               if (icl.eq.20) icl = 19 !  Std A ECR WSS/WSH
               if (icl.eq.22) icl = 23 !  Std A Non-ECR
               if (icl.eq.26) icl = 25 !  Std A NP ECR WSS/WSH
               if (icl.eq.28) icl = 29 !  Std A NP Non-ECR
               if (icl.eq.31) icl = 30 !  4th ZPP
               if (icl.gt.0) then
                  cdols(j,imod,icl,is) = cdols(j,imod,icl,is)
     &                + dollars(j,iw,imod,iact)
               else
                  print *,' activity ',acodes(iact),' not in class map ', iact
               end if
```

```
            end do
         end do
      end do
   end do

C     Write out to file for comparison with previous process

      do icl = 1, ncl
         class3(icl) = class(icl)
      end do

      class3(5) = 'PreL'
      class3(10) = 'PreC'
      class3(19) = 'ECR WSS/H'
      class3(23) = 'BRO'
      class3(25) = 'NECR WSS/H'
      class3(29) = 'NPO'

      open(50,file='nmod00cra_prc_ecr.csv')
51    format(i2,',',i2,',',a14,',',a10,',',i2,',',a5,',',f18.9) !

      do imod = 1, nmod
         do icl = 1, ncl
            do is = 1, nshp
               if ((icl.eq.18).or.(icl.eq.19).or.(icl.eq.21).or.(icl.eq.24).or.
     &             (icl.eq.25).or.(icl.eq.27)) then
                  write (50,51) imod, ldc1(imod), grp(imod), class3(icl), icl, shapetype(is),
     &                cdols(2,imod,icl,is)
               end if
            end do
         end do
      end do

      end


C-----------------------------------------------------------------------
      Assign shape

      function shape(act)

      integer*4    shape
      character*4  act

      if (act(1:1).eq.'1') then
         shape = 1  !  Letters
      else if (act(1:1).eq.'2') then
         shape = 2  !  Flats
      else if ((act(1:1).eq.'3').or.(act(1:1).eq.'4')) then
         shape = 3  !  IPPs/Parcels
      else
         shape = 3  !  Other (Special Service)
         if (act.gt.'1000') then
            print*, 'No shape for actv ', act
         end if
      end if

      return
      end
```