Docket No. R2001-1

USPS-LR-J-59

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. This is a Category 2 library reference sponsored by witness Schenk (USPS-T-43). This library reference updates a previous study (USPS-LR-I-96) sponsored by witness Daniel (USPS-T-28) in Docket No. R2000-1.

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

- USPS-T-11 for BY00 CRA costs
- USPS-LR-J-10 for the IOCS Data Set
- USPS-LR-J-112 for volumes by shape and weight increment
- USPS-LR-J-68 for nontransportation unit cost avoidance
- USPS-LR-J-52 for base year and test year cost factors

Witnesses Hope (USPS-T-31) uses the Standard Mail ECR and NECR mail processing saturation savings developed in this library reference in developing rates for Standard ECR and NECR mail.

II. Organization

The main results are presented in the Excel workbook 'LR-J-59.xls' in the spreadsheet 'Table 1.' Table 1, which is presented below, is analogous to Table 6 in USPS-LR-I-96/R2000-1. Table 1 contains the major results referenced by other Postal Service witnesses in this docket. The methodology used to develop the estimates in Table 1 is the same as that used in USPS-LR-I-96, as described

in USPS-T-28/R2000-1. To update the model for this library reference, the following data were used: base year and test year cost data (from USPS-LR-J-52), 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-68). Data sources are referenced in each spreadsheet in LR-J-59.xls. The underlying mail processing volume-variable costs for clerks and mailhandlers were estimated in a similar manner to that described in Section II of USPS LR-I-101/R2000-1. The current methodology uses FY2000 IOCS data and the Postal Service's proposed 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)

ECR Rate Category	ECR (in cents)
Auto Basic Letters	1.615
Basic Letters	2.766
High Density Letters	0.669
Saturation Letters	0.669
Basic Flats	3.075
Basic Parcels	232.094
Total Basic Nonletters	3.331
High Density/Saturation Flats	1.133
High Density/Saturation Parcels	74.818
Total High Density/Saturation Nonletters	1.152

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 proposed Postal Service mail processing volume-variable cost distribution for clerks and mailhandler mail processing costs.

Program: **cadoc00by_rep.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

Output: **mods12_mp00by_new.dat** – IOCS mail processing tallies for MODS 1&2 offices
mods12_aw00by_new.dat – IOCS administrative and window service tallies for MODS 1&2 offices
bmcs_mp00by_new.dat – IOCS mail processing tallies for BMCs
bmcs_aw00by_new.dat – IOCS administrative and window service tallies for BMCs
nonmods_mp00by_new.dat – IOCS mail processing tallies for Non-MODS offices
nonmods_aw00by_new.dat – IOCS administrative and window service tallies for Non-MODS offices
nonmods_op88_new.dat – IOCS expedited delivery tallies for Non-MODS offices

C. USPS Method Volume-Variable Cost Estimates – Clerks and Mailhandlers, Mail Processing

The volume-variable cost distribution FORTRAN programs replicate the function of witness Van-Ty-Smith's mail processing cost distribution SAS programs documented in USPS LR-J-55. The FORTRAN programs described below divide the cost estimates by Standard ECR rate category, cost pool, and shape.

Program: **modsproc00_wgt.f** – Estimates mail processing volume-variable costs for MODS 1&2 offices by activity code, cost pool, and weight increment

Input: **mods12_mp00by_new.dat** – IOCS mail processing tallies for MODS 1&2 offices
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra – 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.00.619 - List of MODS 1&2 cost pool dollars and corresponding variability factors used in the cost distribution for MODS 1&2 offices (USPS LR-J-55)
windk_wgt_ecr.00.619 – BY00 CRA Cost Segment 3.2 costs by activity code and weight increment for 'function 4 support' cost pool distribution (USPS-LR-J-58, Appendix A, Section C)

Output: **mods002by.data** – Estimated 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 modsproc00_wgt.f from activity code to Standard Mail ECR and NECR rate category by cost pool and shape

Input: **mods002by.data** – Estimated mail processing volume-variable costs by cost pool, activity code, and weight increment for MODS 1&2 offices
costpools.00.619 - List of cost pools for MODS 1&2 offices (USPS-LR-J-55)
activity00.ecr.cra – 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_ecr.csv** – Estimated costs by Standard ECR rate category, cost pool, and shape

Workbook: **mod00 all New.xls** – Summarizes the 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_ecr.csv** – Estimated mail processing volume-variable costs by Standard ECR rate category, cost pool, and shape

Program: **bmcproc00_wgt.f** – Estimates mail processing volume-variable costs for BMCs by activity code, cost pool, and weight increment

Input: **bmcs_mp00by_new.dat** – IOCS mail processing tallies for BMCs
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra – 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 BMC cost pool dollars and corresponding variability factors used in the cost distribution for BMCs (USPS-LR-J-55)

Output: **bmc002by_wgt.data** – Estimated 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 bmcproc00_wgt.f from activity code to Standard Mail ECR and NECR rate categories by cost pool and shape

Input: **bmc002by_wgt.data** – Estimated 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-55)
activity00.ecr.cra – 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_ecr.csv** – Estimated mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Workbook: **bmc00 all New.xls** – Summarizes the 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_ecr.csv** – Estimated mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Program: **nmodproc00_wgt.f** – Estimates mail processing volume-variable costs for Non-MODS offices by activity code, cost pool, and weight increment

Input: **nonmods_mp00by_new.dat** – IOCS mail processing tallies for Non-MODS offices
iocs2000.h – Declaration of IOCS tally fields
activity00.ecr.cra – 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 Non-MODS office cost pool dollars and corresponding variability factors used in the cost distribution for Non-MODS offices (USPS-LR-J-55)

Output: **nmod00by_wgt.data** – Estimated mail processing volume-variable costs by cost pool, activity code, and weight increment

Program: **sumclass_nmod_ecr.f** - Rolls up the output from nmodproc00_wgt.f from activity code to Standard Mail ECR and NECR rate categories by cost pool and shape

Input: **nmod00by_wgt.data** – Estimated 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-55)
activity00.ecr.cra – 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_ecr.csv** – Estimated mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Workbook: **nmod00 all New.xls** – Summarizes the 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_ecr.csv** – Estimated mail processing volume-variable costs by Standard Mail ECR and NECR rate category, cost pool, and shape

Workbook: **LR59Output.xls** – Summarizes the Standard Mail ECR and NECR mail processing volume-variable cost estimates 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.

Input: **mod00 all New.xls**
bmc00 all New.xls
nmod00 all New.xls

Workbook: **Volumes by Wgt GFY00 update.xls** – Volumes by shape and ounce increment controlled to RPW GFY00.

Input: **USPS-LR-J-112, Tables 17-18, 20-21.**

Workbook: **LR-J-59.xls** – Development of test year ECR and NECR mail processing saturation savings and dropship adjusted savings.

Input: **LR59Output.xls**
Volumes by Wgt GFY00 update.xls
USPS-LR-J-52 (base year and test year Standard ECR mail processing costs, piggyback factors, and reconciliation factors)
USPS-LR-J-112 – Standard ECR volumes and weights
USPS-LR-J-68 – Nontransportation unit cost avoidance per pound per entry point

Appendix B: Program Lists

Section I: Preparation of IOCS Data

(Program: cadoc00by_rep.f)

```
      Program cadoc00by_rep

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.

      IMPLICIT NONE

      integer*4    nfin, npool, nmod, nmod2, 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
      parameter    (nmod=861)  !  # of MODS codes
      parameter    (nmod2=862) !  # of MODS codes (including 000)

      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, ldc, bmcgrp2
      integer*4    modgrp, nmodgrp, if9805, if9806, if245
      integer*4    keep, hand, bmcgrp, ier, ct, i1
      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       rf9250, wgt, dlrs, mp_nmod, brk_bmc, brk_nmod
      real*8       mp_mod, mp_bmc, ct_reg_before
      real*8       cost_win, cost_adm, cost_inq, adm_bmc, adm_non
      real*8       win_bmc, win_non, cost_int1, 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


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_mp00by_new.dat',recl=1200) !  MODS 1&2 mail processing IOCS tallies
      open(35,file='mods12_aw00by_new.dat',recl=1200) !  MODS 1&2 admin/window service IOCS tallies
      open(40,file='bmcs_mp00by_new.dat',recl=1200) !  BMCs mail processing IOCS tallies
      open(45,file='bmcs_aw00by_new.dat',recl=1200) !  BMCs admin/window service IOCS tallies
      open(50,file='nonmods_mp00by_new.dat',recl=1200) !  Non-MODS mail processing IOCS tallies
      open(55,file='nonmods_aw00by_new.dat',recl=1200) !  Non-MODS admwin/window service IOCS tallies
      open(56,file='nonmods_op88_new.dat',recl=1200) !  Non-MODS expedited delivery tallies

c     Initialize counter variables

      ier=0
      ct=0
      i1=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
          if (modgrp.eq.32) then !  Intl
             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 (((f245.ge.'001').and.(f245.le.'030')).or.
     &       ((f246.ge.'001').and.(f246.le.'030')).or.
     &       ((if9806.ge.10).and.(if9806.le.300))) then
             if ((f245.eq.'006').or.(f246.eq.'006')) then
                actv = 60
             else if (f245.eq.'019') then
                if ((f9606.eq.'A').and.(f9606.eq.'B')) then
                   actv = 190
                else if ((f245.eq.'019').and.(f9805(2:4).eq.'510')) then
                   actv = 190
                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 = 190
                else
                   actv = if9805
                end if
             else if ((f245.eq.'030').and.((f9606.eq.'C').or.(f9632.eq.'1'))) then
                actv = 300
             else if ((f246.eq.'   ').and.(f247.eq.'   ').and.
     &                 (f248.eq.'   ').and.(f249.eq.'   ')) then
                actv = if9805
                if (f245.eq.'021') then
                   actv = 210
                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 = 90
                else 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
                   read(f245,'(i3)') if245
                   actv = if245*10
                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 = 10
                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 = 50
                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 = 20
                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
              read(f245,'(i3)') if245
              actv = if245*10
           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 = 10
           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 = 50
           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 = 20
           end if
        end if
     else
        actv = if9806
     end if

For international mail
     if ((f9805(1:2).eq.'54').or.(((f9805(2:2).ge.'6').and.
     &     (f9805(2:2).le.'8')).and.(if9805.le.4950))) then
     &     if (((f245.ge.'001').and.(f245.le.'030')).or.
     &        ((f246.ge.'001').and.(f246.le.'030'))) then
              if ((f245.eq.'006').and.(f246.eq.'006')) then
                 actv = 700
              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 (modgrp.eq.27) then !  Registry
     if (actv.eq.60) then
        ct_reg_after = ct_reg_after + 1
     end if
  end if

  if (((ldc.ge.11).and.(ldc.le.17)).or.
     ((ldc.ge.41).and.(ldc.le.44)).or.(ldc.eq.79)) then
     actv = if9805
  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.24).and.(actv.ne.90)) then !  BusReply
     actv = if9805
  end if

  if ((modgrp.eq.27).and.(actv.ne.60)) then !  Registry
     actv = if9805
```

```
      end if

      if ((modgrp.eq.41).and.(actv.lt.100)) then !  LD49
         actv = if9805
      end if

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

c     Special service operations in International
      if (if9806.eq.700) then !  Intl Sp Serv
         if ((f114.eq.'578').or.(f114.eq.'580').or.
     &       (f114.eq.'681').or.(f114.eq.'573').or.
     &       (f114.eq.'577')) then
            if ((f245.ge.'001').and.(f245.le.'030')) then
               actv = 700
            else
               actv = if9805
            end if
         else if ((ldc.eq.18).or.(ldc.eq.48)) then
            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').or.(f2.eq.'054520').or.
     &          (f2.eq.'054523').or.(f2.eq.'055509').or.
     &          (f2.eq.'055513').or.(f2.eq.'056790').or.
     &          (f2.eq.'115851').or.(f2.eq.'115852').or.
     &          (f2.eq.'160046').or.(f2.eq.'160047').or.
     &          (f2.eq.'252493').or.(f2.eq.'268363').or.
     &          (f2.eq.'333869').or.(f2.eq.'350185').or.
     &          (f2.eq.'351029').or.(f2.eq.'482267').or.
     &          (f2.eq.'482268').or.(f2.eq.'484313').or.
     &          (f2.eq.'512704').or.(f2.eq.'512705').or.
     &          (f2.eq.'547619')) then
               actv = 700
            else
               actv = if9805
            end if
         else
            actv = if9805
         end if
      end if

c     Special handling that will incur special serv costs in any pool
c     has to be associated with Std B or Std A Single Piece
      if (((if9806.eq.20).or.(f245.eq.'002').or.(f246.eq.'002').or.
     &    (f247.eq.'002').or.(f248.eq.'002').or.(f249.eq.'002')).and.
     &    ((if9805.ge.1000).and.(if9805.le.4950))) then
         if ((f9805(1:3).eq.'360').or.(f9805(2:2).eq.'4')) then
            actv = 20
         else
            actv = if9805
         end if
      end if

c     Detached forms that will incur special service costs in any pool
      if (f9635.eq.'C') then !  USPS form shape
         actv = actv
         if ((f9606.ge.'A').and.(f9606.le.'B')) then
            actv = 190
         end if
         if ((f9632.eq.'1').and.(f9606.eq.'C')) then
            actv = 300
         end if
         if (f9606.eq.'D') actv = if9805
         if (f9606.eq.'E') actv = 100
         if (f9606.eq.'F') actv = 60
         if (f9606.eq.'G') actv = if9805
         if (f9606.eq.'H') actv = 60
         if (f9606.eq.'I') actv = if9805
      end if

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

```
c       Adjustment to be consistent with what's included in RPW pieces
              if (actv.eq.60) then
                 if ((f9805(2:2).eq.'8').and.((if9805.ge.1000).and.
     &              (if9805.le.4950))) then
                    actv = if9805
                 else if ((f9805(2:4).eq.'510').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

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

c       Correction for business reply - incl BRMAS
              if ((f245.eq.'009').or.(f246.eq.'009')) then
                 if ((modgrp.eq.24).or.(f120.eq.'F')) then
                    actv = 90
                 else
                    actv = if9805
                 end if
              end if

              if ((modgrp.ge.97).or.(modgrp.eq.95)) then !  Admin/Window Service cost pools
                 actv = if9806
              end if

              if ((if9806.eq.110).or.(if9806.eq.120)) then
                 actv = if9806
                 print *,'Del conf tally; pool =',modgrp,' $=',wgt
              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 !  Registry
                 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
      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
         ctaw1=ctaw1+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
         ctmp1=ctmp1+1
         mp_bmc = mp_bmc + dlrs
         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
             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 encirclements

             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
                  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      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

c       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.'645').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 (Claims and Inquiry)
           if ((mod.ge.'551').and.(mod.le.'552')) then
              pool = 97
           end if

         end if

         return
         end
```

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

(Programs: modsproc00_wgt.f, sumclass_mod_ecr.f,
bmcproc00_wgt.f, sumclass_bmc_ecr.f,
nmodproc00_wgt.f, sumclass_nmod_ecr.f)

```
      program modsproc00_wgt

c     Purpose:  Computes distributed volume-variable costs (USPS 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, 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 = 255)     !  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)      !  Number of 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 code
      parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
      parameter (nact2 = 275)    ! 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     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)      ! 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, check, totj
      real*8    atot, btot, ctot, dtot, ftot, gtot, htot, jtot
      real*8    pooldols(nmod)
      real*8    variable(nmod), wincost(nact,nw)
      real*8    varcost(nw,nmod,nact)
      real*8    novarcst(nw,nmod,nact)
      real*8    distsum48, sum48, cost_cntr, cost_unid

      logical flag

      integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
      integer*4 ind, if114, ldc, k, l
      integer*4 cnt, class(nact2), class_code(nact2)
      integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw
      integer*4 ier, ct_cntr, ct_unid, ishp
      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 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

      do i = 1, nmod
         pooldols(i) = 0.0
         variable(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.cra')
21    format(i4,i6,i5)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(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
7     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.00.619')
24    format(2x,a16,i2,f10.0,f6.2)
      do i = 1,nmod
         read(20,24) modcodes(i), ldc1(i), pooldols(i), variable(i)
      end do

      close(20)

c     Window Service costs by activity code and weight category used in Funtion 4 support cost distribution
      open(20,file='windk_wgt_ecr.00.619')
28    format(7x,f16.5)

      do i = 1, nact
         do iw = 1, nw
            read(20,28) wincost(i,iw)
         end do
      end do
      print*, 'MODS Window Service costs read in '
      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.
               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
         jdols(imod) = 0.
      end do
      print*, 'Matrices initialized '

      open(25,file='mods12_mp00by_new.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(f9250,'(f10.0)') rf9250
         read(f166,'(i2)') if166
         read(f167,'(i2)') if167

         dlrs = rf9250/100000.

         tot_dol = tot_dol + dlrs

         ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape

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     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 (non special service handling)
            else if ((f9635.ge.'A').and.(f9635.le.'K')) then
               hand = 1          ! direct (special services)
            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

         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     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 !  Direct single piece
                  atot = atot + dlrs
                  acnt = acnt + 1
                  tot_dol2 = tot_dol2 + dlrs
               else
                  print *,' bad MODS in matrix A ',f114, modgrp, dlrs
               end if
            else                !  Not handling mail
               print *,'Not-handling tally with direct code = ',actv,' cost pool = ',modgrp
               if ((modgrp.gt.0).and.(modgrp.ne.15)) then !  Exclude LDC 15
                  jdols(modgrp) = jdols(modgrp) + 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
            if (modgrp.ne.15) then !  Exclude LDC 15
               jdols(modgrp) = jdols(modgrp) + 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" 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
```

```
          else                ! Not handling mail
             print *,' imat 0 in modgrp = ',actv
             jdols(modgrp) = jdols(modgrp) + dlrs
             jtot = jtot + dlrs
             jcnt = jcnt + 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
                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
          else
             print *,' bad container or mods code ',f9219,',',modgrp
             if ((modgrp.gt.0).and.(modgrp.ne.15)) then !  LDC 15
                jdols(modgrp) = jdols(modgrp) + dlrs
                jtot = jtot + dlrs
                jcnt = jcnt + 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'
           if (modgrp.ne.15) then !  LDC 15
              jdols(modgrp) = jdols(modgrp) + dlrs
              jtot = jtot + dlrs
              jcnt = jcnt + 1
              tot_dol2 = tot_dol2 + dlrs
           end if
        end if

     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

C ***********************************End Read Loop*********************************

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 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 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

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, 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 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

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   'esidual 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 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 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 !  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 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 mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
c     over all activity codes and weight increments within cost pool and item type

      print *,' distributing D '
      do imod = 1, nmod
         do iitem = 1, nitem
            if (ddols(imod,iitem).gt.0.) then
               sum = 0.
               do iact = 1, nact2 !  Distribute over all activity code
                  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) over all activity codes, weight increments,
C     and cost pools within item type using direct item costs ("B" matrix)
      do iitem = 1, nitem
         do imod = 1, nmod
            if (ddols(imod,iitem).gt.0.) then
               print *,'residual distribution of item = ',iitem,' pool = ',imod
               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
      do iact = 1, nact2
         do iitem = 1, nitem
            do imod = 1, nmod
               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          !  Initial distribution within cost pools

         if (imod.ne.20) then    !  Excludes 1Platform

            do icsi = 1, ncsi
               if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
                  sum = 0.
                  distsum = 0.
                  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)
                     end do
                  end do
                  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) *
     &                              adols(iw,imod,iact,icsi) / sum
                              end do
                           end do
                        end if
                     end do
                  else          ! distribute over all cost pools, activity codes, and weight increments
                     do iact = 1, nact2 !  Distribute over all activity codes
                        do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                           do iw = 1 , nw !  Distribute over all weight increments
                              distsum = distsum + adols(iw,i,iact,icsi)
                           end do
                        end do
                     end do
                     if (distsum.gt.0) then
                        do iact = 1, nact2
                           do icon = 1, ncon
                              do i = 1, nmod2
                                 do iw = 1, nw
                                    gdist(iw,imod,icon,iact) =
     &                                 gdist(iw,imod,icon,iact) +
     &                                 gdols(imod,icon,icsi) *
     &                                 adols(iw,i,iact,icsi)/distsum
                                 end do
                              end do
                           end do
                        end do
```

```
            else       !  Undistributed costs included with uncounted/empty containers ("H" matrix)
               print *,'shape distribution empty: mod = ',imod,
                  ', shape = ',icsi
               do icon = 1, ncon
                  hdols(imod,icon) = hdols(imod,icon) +
                     gdols(imod,icon,icsi)
               end do
            end if
         end if
      else             ! Items distributed upon direct item costs ("B" matrix)
         iitem = icsi - nshp2
         sum = 0.
         distsum = 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 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) * bdols(iw,imod,iitem,iact) / sum
                     end do
                  end do
               end if
            end do
         else          ! distribute over all cost pools, activity codes, and weight increments
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
            if (distsum.gt.0) then
               do iact = 1, nact2
                  do icon = 1, ncon
                     do i = 1, nmod2
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) =
                              gdist(iw,imod,icon,iact) +
                              gdols(imod,icon,icsi) *
                              bdols(iw,i,iitem,iact)/distsum
                        end do
                     end do
                  end do
               end do
            else       !  Undistributed costs included with uncounted/empty containers ("H" matrix)
               print *,'shape distribution empty: mod = ',imod,
                  ', shape = ',icsi
               do icon = 1, ncon
                  hdols(imod,icon) = hdols(imod,icon) +
                     gdols(imod,icon,icsi)
               end do
            end if
         end if
      end if

   end do

else if (imod.eq.20) then ! Distribute Platform over all allied labor cost pools, activity codes, and weight increments

   do icsi = 1, ncsi
      if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
         sum = 0.
         do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  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)
                  end if
               end do
            end do
         end do
         if (sum.gt.0.) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
```

```
                do i = 1, nmod2
                   do iact = 1, nact2
                      do iw = 1, nw
                         if ((i.eq.10).or.((i.ge.16).and.(i.le.23))) then ! *CHECK this should be allied labor mod groups
                            gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
&                              gdols(imod,icon,icsi) * adols(iw,i,iact,icsi) / sum
                         end if
                      end do
                   end do
                end do
             end if
          end do
       else          ! distribute over all cost pools, activity codes, and weight increments
          print *,'platform level 2 shape = ',icsi
          do iact = 1, nact2 !  Distribute over all activity codes
             do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                do iw = 1 , nw !  Distribute over all weight increments
                   distsum = distsum + adols(iw,i,iact,icsi)
                end do
             end do
          end do
          if (distsum.gt.0) then
             do iact = 1, nact2
                do icon = 1, ncon
                   do i = 1, nmod2
                      do iw = 1, nw
                         gdist(iw,imod,icon,iact) =
&                           gdist(iw,imod,icon,iact) +
&                           gdols(imod,icon,icsi) *
&                           adols(iw,i,iact,icsi)/distsum
                      end do
                   end do
                end do
             end do
          else       !  Undistributed Platform costs included with uncounted/empty containers ("H" matrix)
             print *,'shape distribution empty: mod = ',imod,
&               ', shape = ',icsi
             do icon = 1, ncon
                hdols(imod,icon) = hdols(imod,icon) +
&                  gdols(imod,icon,icsi)
             end do
          end if
       end if
    else             ! Platform items distributed upon identical/top piece item costs ("B" matrix)
       iitem = icsi - nshp2
       sum = 0.
       do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
          do iact = 1, nact2 !  Distribute over all activity codes
             do iw = 1, nw !  Distribute over all weight increments
                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)
                end if
             end do
          end do
       end do
       if (sum.gt.0.) then
          do icon = 1, ncon
             if (gdols(imod,icon,icsi).gt.0.) then
                do i = 1, nmod2
                   do iact = 1, nact2
                      do iw = 1, nw
                         if ((i.eq.10).or.((i.ge.16).and.(i.le.23))) then ! *CHECK this should be allied labor mod groups
                            gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
&                              gdols(imod,icon,icsi) * bdols(iw,i,iitem,iact) / sum
                         end if
                      end do
                   end do
                end do
             end if
          end do
       else          ! distribute over all cost pools, activity codes, and weight increments
          print *,'platform level 2 item = ',iitem
          do iact = 1, nact2 !  Distribute over all activity codes
             do i = 1, nmod2 !  Distribute over all cost pools (exclude LDC 15 proxy (pool #43))
                do iw = 1 , nw !  Distribute over all weight increments
                   distsum = distsum + bdols(iw,i,iitem,iact)
                end do
             end do
          end do
          if (distsum.gt.0) then
```

```
                  do iact = 1, nact2
                     do icon = 1, ncon
                        do i = 1, nmod2
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) =
     &                           gdist(iw,imod,icon,iact) +
     &                           gdols(imod,icon,icsi) *
     &                           bdols(iw,i,iitem,iact)/distsum
                              check = check + gdols(imod,icon,icsi)*
     &                           bdols(iw,i,iitem,iact)/distsum
                           end do
                        end do
                     end do
                  end do
               else       !  Undistributed Platform costs included with uncounted/empty containers ("H" matrix)
                  print *,'item distribution empty: mod = ',imod,
     &                ', item = ',icsi
                  do icon = 1, ncon
                     hdols(imod,icon) = hdols(imod,icon) +
     &                   gdols(imod,icon,icsi)
                  end do
               end if
            end if
         end if

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

c  Sum distributed "identified" container costs into direct container costs ("F" matrix)
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = 1, nmod
            do iw = 1, nw
               fdols(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + gdist(iw,imod,icon,iact)
               gdist(iw,imod,icon,iact) = 0.
            end do
         end do
      end do
   end do

c  Distribute uncounted/empty containers ("H" matrix) using direct and distributed "identified"
C  container costs ("F" matrix) over all activity codes and weight increments within cost pool and
C  container type
   do imod = 1, 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 + fdols(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) =
     &               hdols(imod,icon) * fdols(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) over all activity codes, weight
C  increments, and cost pools within container type using direct/distributed "identified" container
C  costs ("F" matrix)

   do icon = 1, ncon
      do imod = 1, 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 = 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) + fdols(iw,j,icon,iact)
                  sum = sum + fdols(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)
                  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 = 1, nmod2
               do iw = 1, nw
                  result(iw,imod,iact) = result(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
      end do
c     Containers
      do iact = 1, nact2
         do icon = 1, ncon
            do imod = 1, nmod2
               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 do

C     Distribute not handling costs ("J" matrix) using all other costs ("results" matrix)
      do imod = 1, nmod2        ! All cost pools except LDC 15 proxy
         sum = 0.
         distsum = 0.
c     Exclude allied cost pools (including 1Sacks_M, excluding 1CancMPP), 1EEqmt, Support Fcn 1,
c     and Support Fcn 4
         if (((imod.le.15).and.(imod.ne.10)).or.(imod.eq.17).or.((imod.ge.24).and.(imod.le.28)).or.
     &       ((imod.ge.32).and.(imod.le.37)).or.((imod.ge.40).and.(imod.le.43))) then
            do iact = 1, nact2  !  Distribute over all activity codes
               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) * result(iw,imod,iact) / sum
                  end do
               end do
            else
```

```
            print *,' unable to distribute J dollars for ',imod
          end if

c     Allied not-handling (except cancellation) is distributed across LDCs 11-19, 79
c     except Registry, BusReply, & Support Fcn 1
        else if (((imod.ge.16).and.(imod.le.23)).and.(imod.ne.17).or.(imod.eq.10)) then !  Allied cost pools except cancellation
          do iact = 1, nact2  !  Distribute over all activity codes
            do i = 1, nmod2  !  Distribute over all LDC 11-14, 17 cost pools
              if ((((ldc1(i).ge.11).and.(ldc1(i).le.19)).or.(ldc1(i).eq.79))
     &            .and.((i.ne.27).and.(i.ne.24).and.(i.ne.30).and.(i.ne.31))) then
                do iw = 1, nw !  Distribute over all weight increments
                  distsum = distsum + result(iw,i,iact)
                end do
              end if
            end do
          end do
          if (distsum.gt.0) then
            do iact = 1, nact2
              do i = 1, nmod2
                if ((((ldc1(i).ge.11).and.(ldc1(i).le.19)).or.(ldc1(i).eq.79))
     &              .and.((i.ne.27).and.(i.ne.24).and.(i.ne.30).and.(i.ne.31))) then
                  do iw = 1, nw
                    work(iw,imod,iact) = work(iw,imod,iact) +
     &                jdols(imod) * result(iw,i,iact) / distsum
                  end do
                end if
              end do
            end do
          else
            print *,' unable to distribute J dollars for ',imod
          end if

c     1EEQMT is distributed across all MODS cost pools; distribution includes special services
C     exclude Support Fcn 1 & 4, Registry, BusReply, and LD48 SSV cost pools
        else if (imod.eq.29) then !  1EEqmt
          do iact = 1, nact2  !  Distribute over all activity codes
            do i = 1, nmod2  !  Distribute over all cost pools as noted above
              if ((i.ne.38).and.(i.ne.39).and.(i.ne.30).and.(i.ne.31)
     &            .and.(i.ne.27).and.(i.ne.24).and.(i.ne.40)) then
                do iw = 1, nw !  Distribute over all weight increments
                  distsum = distsum + result(iw,i,iact)
                end do
              end if
            end do
          end do
          if (distsum.gt.0) then
            do iact = 1, nact2
              do i = 1, nmod2
                if ((i.ne.38).and.(i.ne.39).and.(i.ne.30).and.(i.ne.31)
     &              .and.(i.ne.27).and.(i.ne.24).and.(i.ne.40)) then
                  do iw = 1, nw
                    work(iw,imod,iact) = work(iw,imod,iact) +
     &                jdols(imod) * result(iw,i,iact) / distsum
                  end do
                end if
              end do
            end do
          else
            print *,' unable to distribute J dollars for ',imod
          end if

c     1SUPPORT is distributed across all pools (including special service activity codes)
C     exclude Support Fcn 1 & 4, Registry, BusReply, and LD48 SSV cost pools

        else if (imod.eq.31) then !  Support Fcn 1 (not including Misc)
          do iact = 1, nact2  !  Distribute over all activity codes
            do i = 1, nmod2  !  Distribute over all cost pools except as noted above
              if ((i.ne.38).and.(i.ne.39).and.(i.ne.30).and.(i.ne.31)
     &            .and.(i.ne.27).and.(i.ne.24).and.(i.ne.40)) then
                do iw = 1, nw !  Distribute over all weight increments
                  distsum = distsum + result(iw,i,iact)
                end do
              end if
            end do
          end do
          if (distsum.gt.0) then
            do iact = 1, nact2
              do i = 1, nmod2
                if ((i.ne.38).and.(i.ne.39).and.(i.ne.30).and.(i.ne.31)
     &              .and.(i.ne.27).and.(i.ne.24).and.(i.ne.40)) then
```

```
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod) * result(iw,i,iact) / distsum
                  end do
               end if
            end do
         end do
      else
         print *,' unable to distribute J dollars for ',imod
      end if

c     Support Fcn 4 Admin/Other distributed over Function 4 cost pools, except LD48 Exp
      else if ((imod.eq.38).or.(imod.eq.39)) then !  LD48 Oth, LD48 Adm
         do iact = 1, nact2  !  Distribute over all activity codes
            do i = 1, nmod   !  Distribute over all Function 4 cost pools
               if ((ldc1(i).ge.41).and.(ldc1(i).le.48).and.(i.ne.37)) then ! Exclude LD48 Exp
                  do iw = 1, nw !  Distribute over all weight increments
                     distsum = distsum + result(iw,i,iact)
                  end do
               end if
            end do
         end do
         if (distsum.gt.0) then
            do iact = 1, nact2
               do i = 1, nmod
                  if ((ldc1(i).ge.41).and.(ldc1(i).le.48).and.(i.ne.37)) then
                     do iw = 1, nw
                        work(iw,imod,iact) = work(iw,imod,iact) +
     &                     jdols(imod) * result(iw,i,iact) / distsum
                     end do
                  end if
               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 I and 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)
               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 = 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
                     print*, 'Sum = ', sum, ' for actv ', acodes(nact+iact)
                     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   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,nmod
       if ((ldc1(imod).ge.11).and.(ldc1(imod).le.19).and.(imod.ne.31)) then
          distsum = distsum + pooldols(imod) !  Function 1 costs
       end if
       if ((ldc1(imod).ge.41).and.(ldc1(imod).le.49).and.(imod.ne.39)) then !
          distsum48 = distsum48 + pooldols(imod) !  Function 4 costs
       end if
       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     !  Distribute over all direct activity codes
             do iw = 1,nw      !  Distribute over all weight increments
                if ((imod.ne.31).and.(imod.ne.30).and.(imod.ne.38).and.(imod.ne.39)) then
                   varcost(iw,imod,iact) = varcost(iw,imod,iact) +
     &                pooldols(imod)*variable(imod)*result(iw,imod,iact)/sum !  All
                   if (((ldc1(imod).ge.11).and.(ldc1(imod).le.19)).or.(ldc1(imod).eq.79)) then
                      work(iw,31,iact) = work(iw,31,iact) + varcost(iw,imod,iact) !  Distrib key for Support Fcn 1
```

```
                  else if ((ldc1(imod).ge.41).and.(ldc1(imod).le.49)) then
                     work(iw,39,iact) = work(iw,39,iact) + varcost(iw,imod,iact) !  Distrib key for Support Fcn 4
                  end if
                  novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
                     pooldols(imod)*result(iw,imod,iact)/sum
               end if
            end do
         end do
      else
         print *,'unable to distribute $ = ',pooldols(imod),
     &      ' for mods pool ',modcodes(imod)
      end if
   end do

   distsum48 = distsum48 + 765750. !  Add in Window Service cost pool dollars

c  Uses windows cost pool $ as dist key (MODS % of total (MODS,Non-MODS,BMC)
C  MODS, Non-MODS, BMCS cost pool $ (denominator) to add in CRA window service costs (MODS portion)

   do iact = 1, nact
      do iw = 1, nw
         work(iw,39,iact) = work(iw,39,iact) + wincost(iact,iw)*
     &      765750./(765750.+1400378.+341.)
      end do
   end do

C  Function 1 support piggybacked on other Function 1 mail processing
c  Calculation of volume-variable costs for Support Fcn 1 & 4 cost pools
   sum = 0.
   sum48 = 0.
   do iact = 1,nact
      do iw = 1,nw
         sum = sum + work(iw,31,iact)
         sum48 = sum48 + work(iw,39,iact)
      end do
   end do
   if ((sum.gt.0.).and.(sum48.gt.0.)) then
      do iact = 1,nact
         do iw = 1,nw
            varcost(iw,31,iact) = varcost(iw,31,iact) +
     &         pooldols(31)*variable(31)*work(iw,31,iact)/sum
            novarcst(iw,31,iact) = novarcst(iw,31,iact) +
     &         pooldols(31)*work(iw,31,iact)/sum
            varcost(iw,30,iact) = varcost(iw,30,iact) +
     &         pooldols(30)*variable(30)*work(iw,31,iact)/sum
            novarcst(iw,30,iact) = novarcst(iw,30,iact) +
     &         pooldols(30)*work(iw,31,iact)/sum
            varcost(iw,39,iact) = varcost(iw,39,iact) +
     &         pooldols(39)*variable(39)*work(iw,39,iact)/sum48
            novarcst(iw,39,iact) = novarcst(iw,39,iact) +
     &         pooldols(39)*work(iw,39,iact)/sum48
            varcost(iw,38,iact) = varcost(iw,38,iact) +
     &         pooldols(38)*variable(38)*work(iw,39,iact)/sum48
            novarcst(iw,38,iact) = novarcst(iw,38,iact) +
     &         pooldols(38)*work(iw,39,iact)/sum48
         end do
      end do
   else if (sum.eq.0.) then
      print *,'unable to distribute $ = ',pooldols(31),
     &   ' for mods pool ',modcodes(31)
   else if (sum48.eq.0.) then
      print *,'unable to distribute $ = ',pooldols(39),
     &   ' for mods pool ',modcodes(39)
   end if

   sum=sum/distsum
   sum48=sum48/distsum48

   print*,'1Support variability = ',sum
   print*,'LD48 A/O variability = ',sum48

   open(80,file='mods002by.data')
   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,i6,f15.2)') 'J', jcnt, jtot

    end

C ------------------------------------------------------------------------------------------
c     Assigns 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     Assigns 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_mod_ecr

c     Purpose:  Sum distributed volume-variable mail processing costs for MODS 1&2 offices to subclass
c               Costs are calculated in the Fortran program modsproc00_wgt.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 = 255)     !  Number of activity codes
      parameter (ncl = 80)       !  Number of subclasses
      parameter (nshp = 3)       !  Number of shapes
      parameter (nmat = 1)       !  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*16 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.00.619')
      format(i2,a16,i2)

      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.cra')
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='mods002by.data')
41 format(10x,f18.9)

   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
   close(40)

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     ine pools to create Fcn 1 & Fcn 4 Support
      do j = 1, nmat
         do imod = 1, nmod
            do icl = 1, ncl
               do is = 1, nshp
                  if (imod.eq.31) then !  lSupport
                     cdols(j,30,icl,is) = cdols(j,30,icl,is) + cdols(j,imod,icl,is) !  lMisc
                  else if (imod.eq.39) then !  LD48 Adm
                     cdols(j,38,icl,is) = cdols(j,38,icl,is) + cdols(j,imod,icl,is) !  LD48 Oth
                  end if
               end do
            end do
         end do
      end do

C     Write out costs by subclass, cost pool and shape

      open(50,file='mod00cra_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'
      grp(30) = 'Support Fcn1'
      grp(38) = 'Support Fcn4'

      do imod = 1, nmod
         do icl = 1, ncl
            do is = 1, nshp
               if ((imod.ne.31).and.(imod.ne.39)) then
                  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 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 bmcproc00_wgt

c     Purpose:  Computes distributed volume-variable costs (USPS Method) for BMCs
c               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

      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 = 255)     !  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)      !  Number of 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 code
      parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
      parameter (nact2 = 275)    ! 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    work(nw,nmod,nact2) ! Array to hold distributed mixed class-specific
      real*8    jdols(nmod)      ! Not Handling
      real*8    counts(ncsi)
      real*8    actshr(nw,nact2), actshr3(nact), actwgt(nw2), actshr2(nw,nact2)
      real*8     dlrs, sum, distsum, rf9250, check
      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    variable(nmod)
      real*8    varcost(nw,nmod,nact)
      real*8    dlrsin, dlrsout, dlrs5340in, dlrs5340out
      real*8    novarcst(nw,nmod,nact)
      real*8    nowgt_key(nw,nmod,nact2)

      logical flag

      integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
      integer*4 ind, ldc, if9806, iact2, l
      integer*4 cnt,npl,npnl,counted
      integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw
      integer*4 ier, k, 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), 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
      npnl = 0
      counted =0

      do i = 1, nmod
         pooldols(i) = 0.0
         variable(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.cra')
21    format(i4,i6,i5)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(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')
24 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 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
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.
            nowgt_key(iw,imod,iact) = 0.
         end do
      end do
   end do
   do imod = 1, nmod
      jdols(imod) = 0.
   end do
   print*, 'Matrices initialized '

   open(25,file='bmcs_mp00by_new.dat',recl=1200) !  BMC mail proc IOCS data
31 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

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

      if ((modgrp.lt.begmod).or.(modgrp.gt.nmod)) then
         goto 99              !  Exclude break tallies
      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 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

      ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape
      iitem = searchc(codes,nitem,f9214) !  Assign item type
      icon = searchc(codes,ncon,f9219) !  Assign container type

      if (if9806.ge.1000) then
         if9806 = actv
      end if

      iact = searchi(acodes,nact2,actv) !  Activity codes
      iact2 = searchi(acodes,nact2,if9806) !  Activity codes

      dlrsin = dlrsin + dlrs
      modclass=1

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.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 (non special service handling)
          else if ((f9635.ge.'A').and.(f9635.le.'K')) then
             hand = 1          ! direct (special service 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

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     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     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                 !  Not handling mail
              print *,' bad activity in matrix A ',actv
              if (modgrp.gt.0) then
                 jdols(modgrp) = jdols(modgrp) + 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) = jdols(modgrp) + 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                  !  Not handling mail
               print *,' imat 0 in modgrp = ',actv
               jdols(modgrp) = jdols(modgrp) + 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) = jdols(modgrp) + 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     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 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 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   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 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*, 'Level 5 b distribution of act = ', acodes(iact)
                     do iw = 1, nw2
                        actwgt(iw) = 0.
                     end do
                     do k = begmail, nact2 !  Distribute over all 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 if
      end do
   end do
end do

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 !  Disbribute 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 !  Disbribute 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 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) + 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 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 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

      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 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 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 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 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 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
 end do

c    Distribute mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
c    over all activity codes and weight increments within cost pool and item type

    print *,' distributing D '
    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 = 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 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 = 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) + 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
      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    ! Initial distribution within cost pools

      if (imod.ne.1) then    !  Excludes Platform

         do icsi = 1, ncsi
            if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
               sum = 0.
               distsum = 0.
               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)
                  end do
```

```
      end do
      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) *
&                       adols(iw,imod,iact,icsi) / sum
                  end do
               end do
            end if
         end do
      else          ! distribute over all cost pools, activity codes, and weight increments
         do iact = 1, nact2 !  Distribute over all activity codes
            do i = begmod, nmod !  Distribute over all cost pools
               do iw = 1 , nw !  Distribute over all weight increments
                  distsum = distsum + adols(iw,i,iact,icsi)
               end do
            end do
         end do
         if (distsum.gt.0) then
            do iact = 1, nact2
               do icon = 1, ncon
                  do i = begmod, nmod
                     do iw = 1, nw
                        gdist(iw,imod,icon,iact) =
&                          gdist(iw,imod,icon,iact) +
&                          gdols(imod,icon,icsi) *
&                          adols(iw,i,iact,icsi)/distsum
                     end do
                  end do
               end do
            end do
         else       !  Undistributed costs included with uncounted/empty containers ("H" matrix)
            print *,'shape distribution empty: mod = ',imod,
&              ', shape = ',icsi
            do icon = 1, ncon
               hdols(imod,icon) = hdols(imod,icon) +
&                 gdols(imod,icon,icsi)
            end do
         end if
      end if
   else             ! Items distributed upon direct item costs ("B" matrix)
      iitem = icsi - nshp2
      print *,' G dist',imod,' ',iitem
      if (iitem.eq.9) then !  Exclude green sacks
         do icon = 1,ncon
            if (gdols(imod,icon,icsi).gt.0.) then
               print *,'mxd green sack in container, group =',imod,' $ =',
&                 gdols(imod,icon,icsi)
            end if
         end do
      else
         sum = 0.
         distsum = 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 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) * bdols(iw,imod,iitem,iact) / sum
                        ddist(iw,imod,iact) = ddist(iw,imod,iact) +
&                          gdols(imod,icon,icsi) * bdols(iw,imod,iitem,iact) / sum
                     end do
                  end do
               end if
            end do
         else       ! distribute over all cost pools, activity codes, and weight increment
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = begmod, nmod !  Distribute over all cost pools
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + bdols(iw,i,iitem,iact)
```

```
                        end do
                     end do
                  end do
                  if (distsum.gt.0) then
                     do iact = 1, nact2
                        do icon = 1, ncon
                           do i = begmod, nmod
                              do iw = 1, nw
                                 gdist(iw,imod,icon,iact) =
&                                   gdist(iw,imod,icon,iact) +
&                                   gdols(imod,icon,icsi) *
&                                   bdols(iw,i,iitem,iact)/distsum
                                 ddist(iw,imod,iact) =
&                                   ddist(iw,imod,iact) +
&                                   gdols(imod,icon,icsi) *
&                                   bdols(iw,i,iitem,iact)/distsum
                              end do
                           end do
                        end do
                     end do
                  else    !  Undistributed costs included with uncounted/empty containers ("H" matrix)
                     print *,'shape distribution empty: mod = ',imod,
&                       ', shape = ',icsi
                     do icon = 1, ncon
                        hdols(imod,icon) = hdols(imod,icon) +
&                          gdols(imod,icon,icsi)
                     end do
                  end if
               end if
            end if
         end if

      end do

   else if (imod.eq.1) then ! Distribute Platform over all allied labor cost pools, activity codes, and weight increments

      do icsi = 1, ncsi
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            sum = 0.
            do i = begmod, nmod !  Distribute over all cost pools
               do iact = 1, nact2 !  Distribute over all activity codes
                  do iw = 1, nw !  Distribute over all weight increments
                     sum = sum + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
            if (sum.gt.0.) then
               do icon = 1, ncon
                  if (gdols(imod,icon,icsi).gt.0.) then
                     do i = begmod, nmod
                        do iact = 1, nact2
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
&                                gdols(imod,icon,icsi) * adols(iw,i,iact,icsi) / sum
                           end do
                        end do
                     end do
                  end if
               end do
            else          ! Platform items distributed upon identical/top piece item costs ("B" matrix)
               print *,'platform level 2 shape = ',icsi
               do iact = 1, nact2 !  Distribute over all activity codes
                  do i = begmod, nmod !  Distribute over all cost pools
                     do iw = 1 , nw !  Distribute over all weight increments
                        distsum = distsum + adols(iw,i,iact,icsi)
                     end do
                  end do
               end do
               if (distsum.gt.0) then
                  do iact = 1, nact2
                     do icon = 1, ncon
                        do i = begmod, nmod
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) =
&                                gdist(iw,imod,icon,iact) +
&                                gdols(imod,icon,icsi) *
&                                adols(iw,i,iact,icsi)/distsum
                           end do
                        end do
                     end do
                  end do
```

```
            end do
         else       !  Undistributed Platform costs included with uncounted/empty containers ("H" matrix)
            print *,'shape distribution empty: mod = ',imod,
               ', shape = ',icsi
            do icon = 1, ncon
               hdols(imod,icon) = hdols(imod,icon) +
     &            gdols(imod,icon,icsi)
            end do
         end if
      end if
   else              ! Platform items distributed upon identical/top piece item costs ("B" matrix)
      iitem = icsi - nshp2
      sum = 0.
      if ((iitem.eq.2).or.(iitem.eq.9)) then !  Exclude con-con and green sacks
         do icon = 1,ncon
            if (gdols(imod,icon,icsi).gt.0.) then
               print *,'mxd item=',iitem,' in container, group =',imod,' $ =',
     &            gdols(imod,icon,icsi)
c               print *,'concon in platform container'
            end if
         end do
      else
         do i = begmod, nmod !  Distribute over all cost pools
            do iact = 1, nact2 !  Distribute over all activity codes
               do iw = 1, nw !  Distribute over all weight increments
                  sum = sum + bdols(iw,i,iitem,iact)
               end do
            end do
         end do
         if (sum.gt.0.) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do i = begmod, nmod
                     do iact = 1, nact2
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
     &                        gdols(imod,icon,icsi) * bdols(iw,i,iitem,iact) / sum
     &                        ddist(iw,imod,iact) = ddist(iw,imod,iact) +
     &                        gdols(imod,icon,icsi) * bdols(iw,i,iitem,iact) / sum
                        end do
                     end do
                  end do
               end if
            end do
         else
            print *,'platform level 2 item = ',iitem
            distsum = 0.
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = begmod, nmod !  Distribute over all cost pools
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
            if (distsum.gt.0) then
               do iact = 1, nact2
                  do icon = 1, ncon
                     do i = begmod, nmod
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) =
     &                        gdist(iw,imod,icon,iact) +
     &                        gdols(imod,icon,icsi) *
     &                        bdols(iw,i,iitem,iact)/distsum
                           ddist(iw,imod,iact) =
     &                        ddist(iw,imod,iact) +
     &                        gdols(imod,icon,icsi) *
     &                        bdols(iw,i,iitem,iact)/distsum
                        end do
                     end do
                  end do
               end do
            else    !  Undistributed Platform costs included with uncounted/empty containers ("H" matrix)
               print *,'item distribution empty: mod = ',imod,
     &            ', item = ',icsi
               do icon = 1, ncon
                  hdols(imod,icon) = hdols(imod,icon) +
     &               gdols(imod,icon,icsi)
               end do
            end if
         end if
```

```
                end if
              end if

          end do

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

c     Distribution 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) !  Platform
               cdist(iw,2,iitem,iact) = cdist(iw,2,iitem,iact)*37640.0/(37640.0-393.4-49.5) !  Allied
            end do
            do icon = 1,ncon
               gdist(iw,1,icon,iact) = gdist(iw,1,icon,iact)*36326.0/(36326.0-75.1) !  Platform
               gdist(iw,2,icon,iact) = gdist(iw,2,icon,iact)*37640.0/(37640.0-393.4-49.5) !  Allied
            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 over all activity codes and weight increment within cost pool and
C     container type
      do imod = begmod, nmod
         do icon = 1, ncon
            sum = 0.
            do iact = 1, nact2
               do iw = 1, nw
                  sum = sum + hkey(iw,imod,icon,iact)
               end do
            end do
            if (sum.gt.0) then
               do iact = 1, nact2 !  Distribute over all activity codes
                  do iw = 1, nw !  Distribute over all weight increments
                     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) over all activity codes, weight
C     increment, and cost pools within container type using direct/distributed "identified" container
C     costs

      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     Sum up all costs (direct and redistributed) except not handling costs ("J" matrix)
c     Direct costs
      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
                  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)
               end do
            end do
         end do
      end do
c     Containers
      do iact = 1, nact2
         do icon = 1, ncon
            do imod = begmod, nmod
               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
            end do
         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(2) = jdols(2) + 273603.0 - 231878.0 !  Allied Oth
      jdols(1) = jdols(1) + 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.3) then    !  Exclude allied and platform cost pools
            sum = 0.
            distsum = 0.
            do iact = 1, nact2  !  Distribute over all activity codes
               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) * result(iw,imod,iact) / sum
               end do
            end do
         else
            print *,' unable to distribute J dollars for ',imod
         end if
      else                    !  Distribute allied other and platform over all cost pools
         sum = 0.
         distsum = 0.
         do iact = 1, nact2   !  Distribute over all activity codes
            do iw = 1, nw     !  Distribute over all weight increments
               do i = begmod, nmod !  Distribute over all cost pools
                  sum = sum + result(iw,i,iact)
               end do
            end do
         end do
         if (sum.gt.0) then
            do i = begmod, nmod
               do iact = 1, nact2
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod) * result(iw,i,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)
   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.
                  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
               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

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)*variable(imod)*result(iw,imod,iact)/sum
                  novarcst(iw,imod,iact) = novarcst(iw,imod,iact) +
     &                pooldols(imod)*variable(imod)*result2(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='bmc002by_wgt.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 *,'IOCS $ in  = ',dlrsin
      print *,'IOCS $ out = ',dlrsout
      print *,'5340 $ = ',dlrs5340in,' ',dlrs5340out

      end

C -------------------------------------------------------------------------------------------
c     Assigns 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     Assigns 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 distributed volume-variable mail processing costs for BMCs to subclass
c               Costs are calculated in the Fortran program bmcproc00_wgt.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 = 255)     !  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

c     Map of cost pools
      open(30,file='costpools.00.bmc.619')
      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.cra')
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='bmc002by_wgt.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 costs by subclass, cost pool and shape

    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_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(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 nmodproc00_wgt

Purpose:  Computes distributed volume-variable costs (USPS Method) for Non-MODS 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

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 = 255)     !  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)      !  Number of 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 code
parameter (nmixcl = 20)    ! Number of class-specific mixed-mail codes
parameter (nact2 = 275)    ! 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     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)      ! 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, check
real*8    bmix(nmod,nact2)
real*8    atot, btot, ctot, dtot, ftot, gtot, htot, jtot
real*8    pooldols(nmod)
real*8    variable(nmod)
real*8    varcost(nw,nmod,nact)
real*8    novarcst(nw,nmod,nact)
real*8    gfy,ovhfact,ovh6522,wgt,wgt6521,wgtall

logical flag

integer*4 acnt, bcnt, ccnt, dcnt, fcnt, gcnt, hcnt, jcnt
integer*4 ind, ldc, l
integer*4 cnt,npl,npnl,counted, class(nact2)
integer*4 i, j, imat, imod, icon, iact, icsi, iitem, shapeind, iw
integer*4 ier, k, class_code(nact2), poolcode(nmod), pool
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 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.

      do i = 1, nmod
         pooldols(i) = 0.0
         variable(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.cra')
21    format(i4,i6,i5)
      do i=1,nact2
         read (20,21) acodes(i), class(i), class_code(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')
24 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    ! a matrix
            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.
         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
   jdols(imod) = 0.
end do
print*, 'Matrices initialized '

open(25,file='nonmods_mp00by_new.dat',recl=1200) !  Non-MODS offices mail proc IOCS data
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)

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

c     Calculation of overhead factors to convert tally dollar weights to cost pool dollars
      gfy = 4833549./4402680.
      ovhfact = 2553568./2328659.
      ovh6522 = 4402680./4340556.
      wgt = rf9250/100000.

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

      dlrs = wgt*gfy*ovhfact*ovh6522

      if ((modgrp.lt.begmod).or.(modgrp.gt.nmod)) then !  Exclude break tallies
         goto 99
      end if

      tot_dol = tot_dol + dlrs

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     deals with new international mixed codes

      ishp = shapeind(actv,f9635,f9805) !  Subroutine assigns shape

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 (non special services)
         else if ((f9635.ge.'A').and.(f9635.le.'K')) then
            hand = 1         ! direct (special services)
         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

      if ((hand.eq.1).and.(iact.eq.0)) then
         print *,'missing direct activity code = ',actv,' modgrp = ',modgrp
      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                  !  Not handling mail
            print *,'Not-handling tally with direct code = ',actv,' cost pool = ',modgrp
            if (modgrp.gt.0) then
               jdols(modgrp) = jdols(modgrp) + 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) = jdols(modgrp) + 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

      "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
          else                  !  Not handling mail
            print *,' imat 0 in modgrp = ',actv
            jdols(modgrp) = jdols(modgrp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 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
          else                  !  Not handling
            print *,' bad container or mods code ',f9219,',',modgrp
            jdols(modgrp) = jdols(modgrp) + dlrs
            jtot = jtot + dlrs
            jcnt = jcnt + 1
            tot_dol2 = tot_dol2 + dlrs
          end if

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

```
            jdols(modgrp) = jdols(modgrp) + 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

C ***********************************End Read Loop*********************************
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 = begmail, 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 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 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 = begmail, 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 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*, 'Level 5 b distribution of act = ', acodes(iact)
                      do iw = 1, nw2
                         actwgt(iw) = 0.
                      end do
                      do k = begmail, nact2 !  Distribute over all 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 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 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 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,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 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 mixed/empty item costs ("D" matrix) using direct item costs ("B" matrix) as a distribution key
c     over all activity codes and weight increments within cost pool and item type

      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 code
                  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) over all activity codes, weight increments,
      and cost pools within item type using direct item costs ("B" matrix)
      do iitem = 1, nitem
         do imod = begmod, nmod
            if (ddols(imod,iitem).gt.0.) then
               print *,'residual distribution of item = ',iitem,' pool = ',imod
               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) + 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    !  Initial distribution within cost pools

         if (imod.ne.1) then    !  Exclude Allied pool

            do icsi = 1, ncsi
               if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
                  sum = 0.
                  distsum = 0.
                  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)
                     end do
                  end do
                  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) *
     &                              adols(iw,imod,iact,icsi) / sum
                              end do
                           end do
                        end if
                     end do
                  else          ! distribute over all cost pools, activity codes, and weight increments
                     do iact = 1, nact2 !  Distribute over all activity codes
                        do i = begmod, nmod !  Distribute over all cost pools
                           do iw = 1 , nw !  Distribute over all weight increments
                              distsum = distsum + adols(iw,i,iact,icsi)
                           end do
                        end do
                     end do
                     if (distsum.gt.0) then
                        do iact = 1, nact2
                           do icon = 1, ncon
                              do i = begmod, nmod
                                 do iw = 1, nw
                                    gdist(iw,imod,icon,iact) =
     &                                 gdist(iw,imod,icon,iact) +
     &                                 gdols(imod,icon,icsi) *
     &                                 adols(iw,i,iact,icsi)/distsum
```

```
                        end do
                     end do
                  end do
               end do
            else
               print *,'shape distribution empty: mod = ',imod,
&                 ', shape = ',icsi
            end if
         end if
      else              ! Items distributed upon direct item costs ("B" matrix)
         iitem = icsi - nshp2
         sum = 0.
         distsum = 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 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) * bdols(iw,imod,iitem,iact) / sum
                     end do
                  end do
               end if
            end do
         else           ! distribute over all cost pools, activity codes, and weight increments
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = begmod, nmod !  Distribute over all cost pools
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
            if (distsum.gt.0) then
               do iact = 1, nact2
                  do icon = 1, ncon
                     do i = begmod, nmod
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) =
&                             gdist(iw,imod,icon,iact) +
&                             gdols(imod,icon,icsi) *
&                             bdols(iw,i,iitem,iact)/distsum
                        end do
                     end do
                  end do
               end do
            else
               check = 0.
               do icon = 1,ncon
                  check = check + gdols(imod,icon,icsi)
               end do
               if (check.gt.0.) then
                  print *,'shape distribution empty: mod = ',imod,
&                    ', shape = ',icsi
               end if
            end if
         end if
      end do

   else if (imod.eq.1) then ! Distribute allied over all cost pools, activity codes, and weight increments

      do icsi = 1, ncsi
         if (icsi.le.5) then ! Loose shapes distributed based upon direct piece costs ("A" matrix)
            sum = 0.
            distsum = 0.
            do i = begmod, nmod !  Distribute over all cost pools
               if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc
                  do iact = 1, nact2 !  Distribute over all activity codes
                     do iw = 1, nw !  Distribute over all weight increments
                        sum = sum + adols(iw,i,iact,icsi)
                     end do
                  end do
               end if
            end do
            if (sum.gt.0.) then
```

```
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do i = begmod, nmod
                     if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc
                        do iact = 1, nact2
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
&                                gdols(imod,icon,icsi) * adols(iw,i,iact,icsi) / sum
                           end do
                        end do
                     end if
                  end do
               end if
            end do
         else          ! distribute over all cost pools, activity codes, and weight increments
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = begmod, nmod !  Distribute over all cost pools
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + adols(iw,i,iact,icsi)
                  end do
               end do
            end do
            if (distsum.gt.0) then
               do iact = 1, nact2
                  do icon = 1, ncon
                     do i = begmod, nmod
                        do iw = 1, nw
                           gdist(iw,imod,icon,iact) =
&                             gdist(iw,imod,icon,iact) +
&                             gdols(imod,icon,icsi) *
&                             adols(iw,i,iact,icsi)/distsum
                        end do
                     end do
                  end do
               end do
            else
               print *,'shape distribution empty: mod = ',imod,
&                 ', shape = ',icsi
            end if
         end if
      else              ! Allied items distributed upon identical/top piece item costs ("B" matrix)
         iitem = icsi - nshp2
         sum = 0.
         distsum = 0.
         do i = begmod, nmod !  Distribute over all cost pools
            if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc
               do iact = 1, nact2 !  Distribute over all activity codes
                  do iw = 1, nw !  Distribute over all weight increments
                     sum = sum + bdols(iw,i,iitem,iact)
                  end do
               end do
            end if
         end do
         if (sum.gt.0.) then
            do icon = 1, ncon
               if (gdols(imod,icon,icsi).gt.0.) then
                  do i = begmod, nmod
                     if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc
                        do iact = 1, nact2
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) = gdist(iw,imod,icon,iact) +
&                                gdols(imod,icon,icsi) * bdols(iw,i,iitem,iact) / sum
                           end do
                        end do
                     end if
                  end do
               end if
            end do
         else          ! distribute over all cost pools, activity codes, and weight increments
            print *,'Allied level 2 item = ',iitem
            do iact = 1, nact2 !  Distribute over all activity codes
               do i = begmod, nmod !  Distribute over all cost pools
                  do iw = 1 , nw !  Distribute over all weight increments
                     distsum = distsum + bdols(iw,i,iitem,iact)
                  end do
               end do
            end do
            if (distsum.gt.0) then
               do iact = 1, nact2
                  do icon = 1, ncon
```

```
                        do i = begmod, nmod
                           do iw = 1, nw
                              gdist(iw,imod,icon,iact) =
                                 gdist(iw,imod,icon,iact) +
     &                           gdols(imod,icon,icsi) *
     &                           bdols(iw,i,iitem,iact)/distsum
                           end do
                        end do
                     end do
                  end do
               else
                  print *,'item distribution empty: mod = ',imod,
     &               ', item = ',icsi
               end if
            end if
         end if

      end do

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

c  Additional item and container distributions
   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) !  Manual Flats
            cdist(iw,6,iitem,iact) = cdist(iw,6,iitem,iact)*9539.6/(9539.6-942.6) !  Manual Parcels
            cdist(iw,8,iitem,iact) = cdist(iw,8,iitem,iact)*5823.7/(5823.7-199.7) !  Misc
         end do
         do icon = 1,ncon
            gdist(iw,1,icon,iact) = gdist(iw,1,icon,iact)*52293.4/(52293.4-686.7) !  Allied
            gdist(iw,4,icon,iact) = gdist(iw,4,icon,iact)*23135.8/(23135.8-218.8) !  Manual Flats
            gdist(iw,6,icon,iact) = gdist(iw,6,icon,iact)*9539.6/(9539.6-942.6) !  Manual Parcels
            gdist(iw,8,icon,iact) = gdist(iw,8,icon,iact)*5823.7/(5823.7-199.7) !  Misc
         end do
      end do
   end do

c  Sum distributed "identified" container costs into direct container costs ("F" matrix)
   do iact = 1, nact2
      do icon = 1, ncon
         do imod = begmod, nmod
            do iw = 1, nw
               fdols(iw,imod,icon,iact) = fdols(iw,imod,icon,iact) + gdist(iw,imod,icon,iact)
               gdist(iw,imod,icon,iact) = 0.
            end do
         end do
      end do
   end do

c  Distribute uncounted/empty containers ("H" matrix) using direct and distributed "identified"
C  container costs ("F" matrix) over all activity codes and weight increment categories within cost pool and
C  container type

   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 + fdols(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) =
     &               hdols(imod,icon) * fdols(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) over all activity codes, weight
C  increments, and cost pools within container type using direct/distributed "identified" container
C  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) + fdols(iw,j,icon,iact)
                     sum = sum + fdols(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 = 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
                  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
      end do
c     Containers
      do iact = 1, nact2
         do icon = 1, ncon
            do imod = begmod, nmod
               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 do

      Distribute not handling costs ("J" matrix) using all other costs ("results" matrix)
      do imod = begmod, nmod
         sum = 0.
         distsum = 0.
         if ((imod.ne.1).and.(imod.ne.8)) then !  Exclude allied and miscellaneous cost pools
            do iact = 1, nact2  !  Distribute over all activity codes
               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) * result(iw,imod,iact) / sum

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

      else if (imod.eq.1) then !  Allied cost pool
         do iact = 1, nact2  !  Distribute over all activity codes
            do i = begmod, nmod !  Distribute over all cost pools
               if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc cost pools
                  do iw = 1, nw !  Distribute over all weight increments
                     distsum = distsum + result(iw,i,iact)
                  end do
               end if
            end do
         end do
         if (distsum.gt.0) then
            do iact = 1, nact2
               do i = begmod, nmod
                  if ((i.ne.7).and.(i.ne.8)) then !  Exclude Registry and Misc
                     do iw = 1, nw
                        work(iw,imod,iact) = work(iw,imod,iact) +
     &                     jdols(imod) * result(iw,i,iact) / distsum
                     end do
                  end if
               end do
            end do
         else
            print *,' unable to distribute J dollars for ',imod
         end if

      else if (imod.eq.8) then !  Misc cost pool
         do iact = 1, nact2  !  Distribute over all activity codes
            do i = begmod, nmod !  Distribute over all cost pools
               do iw = 1, nw !  Distribute over all weight increments
                  distsum = distsum + result(iw,i,iact)
               end do
            end do
         end do
         if (distsum.gt.0) then
            do iact = 1, nact2
               do i = begmod, nmod
                  do iw = 1, nw
                     work(iw,imod,iact) = work(iw,imod,iact) +
     &                  jdols(imod) * result(iw,i,iact) / distsum
                  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)
   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 increments, and within cost pools
   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
    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)*variable(imod)
                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 results to a file

      open(80,file='nmod00by_wgt.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 *,'total wgt w/o 6521 = ',wgt6521
      print *,'total wgt inc 6521 = ',wgtall

      end

C ------------------------------------------------------------------------------------------
c     Assigns 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     Assigns 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_nmod_ecr

c     Purpose:  Sum distributed volume-variable mail processing costs for Non-MODS offices to subclass
c               Costs are calculated in the Fortran program nmodproc00_wgt.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 = 255)     !  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

c     Map of cost pools
      open(30,file='costpools.00.nmod.619')
      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.cra')
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='nmod00by_wgt.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 costs by subclass, cost pool and shape

      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_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
```