RECEIVED
JAN 3 4 44 PM '02
POSTAL RATE COMMISSION
OFFICE OF THE SECRETARY

Docket No. R2001-1

USPS-LR-J-191 (Errata)

Carrier Costs for First-Class Single Piece Letters By Indicia, Provided in Response to MMA/USPS-T43-13A

(Category 4 Library Reference)

Table of Contents

Introduction

This library reference provides the supporting analyses used to estimate carrier costs for First-Class Single Piece letters by shape and indicia. This is a Category 4 library reference provided in response to interrogatory MMA/USPS-T43-13A.

The methodology used to develop the carrier costs by shape and indicia is very similar to that used in USPS-LR-J-117. Other testimony and library references used in this analysis include:

- USPS-LR-J-10 for the IOCS data set
- USPS-LR-J-57 for CRA worksheets
- USPS-LR-J-112 for First-Class Single Piece letter volumes by shape and indicia
- USPS-LR-J-117 for carrier costs and piggyback factors (base year and test year)

Organization

The final carrier base year and test year costs by shape and indicia for First-Class Single Piece letters are provided in the Excel workbook MMAT43_1o_revised.xls. The city carrier in-office costs (cost segments 6.1 and 6.2) are calculated using a FORTRAN version of the LIOCATT process similar to the one described in USPS-LR-J-117, Appendix A. The FORTRAN programs used for this library reference are described in Appendix A of this document.

The costs by indicia for other carrier cost segments (7.1-7.3, and 10) are derived by distributing the costs from the Excel workbook LR-J-117.xls in USPS-LR-J-117 to indicia using a volume distribution key from USPS-LR-J-112, table 10. Costs for cost segment 7.4 are distributed to indicia by using the distributed costs for the other city carrier cost segments (6.1,6.2, and 7.1-7.3) as a distribution key.

Appendix A: Program Documentation

I. 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 ending with an ".f" file extension are FORTRAN programs and programs ending in an ".sm" file extension are SORT/MERGE programs. The remaining data processing is performed in Excel workbooks (.xls file extension) on PCs running the Microsoft Windows NT 4 and Windows 2000 operating systems and Microsoft Office.

II. Preparation of IOCS Data

The following programs are used to extract, code and process the 2000 IOCS data in preparation for LIOCATT distribution of city carrier in-office costs (CRA Cost Segment 6.1)

Program: **encode_mtr.f** - Extracts the necessary data from the IOCS tally data set, encodes specific tally fields into indexes, and writes the indexes to be used by LIOCATT

Input: **FY00 IOCS tally data** (USPS LR-J-10)
iocs2000.h – Declaration of IOCS tally fields
fincag.98 – Tally finance number and CAG combinations
activity00.ecr.all – List of activity codes

Output: **encdata** – Encoded IOCS tallies

Program: **encdata.sm** - Sorts the encoded IOCS data for the LIOCATT process

Input: **encdata**

Output: **encdata.s**

III. LIOCATT Based Cost Distribution Process

The LIOCATT distribution process is run through a main FORTRAN program, which uses subroutines to distribute mixed-mail costs, which are also FORTRAN programs. The declaration file 'liocatt.h' is included in each program and subroutine. This file contains common variables used by all programs and subroutines.

Program: **liocatt.f** - This program controls the LIOCATT process by running various FORTRAN programs, which replicates the LIOCATT process for mixed-mail cost distribution

Subroutines: **fillmixmap.f** – Produces a map for distributing the mixed mail codes to appropriate direct activity codes

Input: **activity00.ecr.all** – List of activity codes
mmcodes.intl – List of mixed-mail activity codes
mxmail.all.ecr – Maps class specific mixed-mail activity codes to corresponding direct activity codes

loaddata.f - Loads the encoded IOCS data

Input: **encdata.s** – Encoded IOCS data

fungroup.f - Forms function groups for operations

Input: **operrtemap** – Maps operation to function group

sortcost.f - Sort records for level 1 cost distribution

level1.f - Level 1 distribution of mixed-mail/not-handling tallies

level2.f - Level 2 distribution of mixed-mail/not-handling tallies

sortlev2a.f – Sort records for level 3 cost distribution

level3.f - Level 3 distribution of mixed-mail/not-handling tallies

report.f - Write results to file

Output: **level1b** – Level 1 distributed direct costs
level2b – Level 2 distributed direct costs
level3a – Level 3 direct costs
level3b – Level 3 distributed direct costs

IV. First-Class Single Piece Letters Cost Estimates

Program: **rpt_ind.f** - Summarizes LIOCATT cost distribution estimations by indicia for city carrier in-office costs, First-Class Single Piece letters only

Input: **activity00.ecr.all** - List of activity codes and corresponding subclass codes
classes_ecr.old – List of CRA subclasses
level1b – Level 1 distributed direct costs
level2b - Level 2 distributed direct costs
level3a – Level 3 direct costs
level3b – Level 3 distributed direct costs

Output: **car001SP_ind.csv** - Estimated city carrier in-office First-Class Single Piece letter costs by indicia

V. Final Summary Spreadsheets

Workbook: **MMAT43-1o_revised.xls** – Calculates the First-Class Single Piece letter costs by indicia for carriers by cost segment

Input: **car001SP_ind.csv** - Estimated city carrier in-office First-Class Single Piece letter costs by indicia
BY00 CRA Costs and Piggyback Factors – LR-J-117.xls, worksheet 'BY Summary' (USPS-LR-J-117)
TY03 CRA Costs and Piggyback Factors – LR-J-117.xls, worksheet 'TY Summary' (USPS-LR-J-117)
BY00 RPW Volumes – First-Class Single Piece letter volumes by indicia (USPS-LR-J-112, Table 10)

Appendix B: Program Lists

```
      PROGRAM encode_mtr
C
C       PURPOSE: Encode tallies with indexes of arrays instead of
C                actual data. Delete leave tallies. Split old group 26
C                into new sub groups, change nonmod groups

C       Modified:  Break out First-Class Single Piece into Stamped, Metered, and Other
C                   for MMA Interrogatory MMA/USPS-T43-1, part O


      IMPLICIT NONE

      integer*4   numcf, numact, numor, nw

      parameter   (numcf = 11)  ! number of cag - finnance number combs.
      parameter   (numor = 17)  ! number of operation/route codes
      parameter   (numact = 501) ! number of activity codes
      parameter   (nw = 3)      ! number of categories

      include 'iocs2000.h'

      integer*4   desigind      ! function to assign pay category
      integer*4   orind         ! function to assign operation/route code
      integer*4   bfind         ! function to assign basic function index
      integer*4   indicia       ! function to assign indicia
      integer*4   i2, i3, i4, i5, i6, i8, i7, i9
      integer*4   i, searchc, z, totall, ier, countlv
      integer*4   counto, countg, countsp, countk, countf

      real*8    tvalue, cost_clk, cost_mp

      character*7    cagfins(numcf), fincag
      character*4    acodes(numact), activity

      countlv = 0
      counto = 0
      countg = 0
      countsp = 0
      countk = 0
      countf = 0
      totall = 0
      ier = 0
      i2 = 0
      i3 = 0
      i4 = 0
      i5 = 0
      i6 = 0
      i7 = 0
      i8 = 0
      i9 = 0

c     Map of Fin CAG combinations (used for strata)
      open(14,file='fincag.98',iostat=ier)
      if (ier.ne.0) then
         print *,' error opening cagfin.s = ',ier
         stop
      end if

15    format(a7)
      do i = 1,numcf
         read(14,15) cagfins(i)
      end do
      print *,'finished reading cags  '

c     Map of activity codes
      open(16,file='activity00.ecr.all',iostat=ier )
      if (ier.ne.0) then
         print *,' error opening activity code files = ',ier
         stop
      end if

17    format(a4)
      do i = 1,numact
         read(16,17) acodes(i)
      end do
      close (14)
      close (16)
      print *,'finished reading activity codes  '

      open(25,file='iocsdata.2000.new',recl=1200) !  FY00 IOCS Data Set (based on full data set)
```

```
      open(30,file='encdata')
21    format(a)
31    format(i2,i1,i1,i3,i2,i3,i2,f11.2)

      ier = 0
      z=0
      cost_clk = 0.0
      cost_mp = 0.0

      do while (ier.eq.0)

         read(25,21,iostat=ier,end=100) rec

         z=z+1
         fincag = f263//f264    !  Strata

         i2 = searchc(cagfins,numcf,fincag) !  Find fincag (strata)
         i3 = desigind(f257)    ! craft
         i4 = orind(f260)       ! operation or route code
         if (i3.eq.1) i4 = 1    ! supervisors have no route or oper code
         i5 = bfind(f261)       ! basic function
         if (i3.eq.1) i5 = 1    ! supervisors have bf set to 0

         activity = f9806       ! activity code

         if (activity(1:1).ge.'1'.and.activity(1:1).le.'4'.and. ! map x091 to x080
     &      activity(2:4).eq.'091') then
            activity(2:4) = '080'
         end if

         if (i3.eq.2) then
            if (i4.eq.13) i4 = 11 !  Dump Op 88 into other for clk/mh
         end if

         i6 = searchc(acodes,numact,activity)

         if (i6.eq.0) then
            print*, 'Activity code not found ', activity
         end if

         i9 = 1

c     Set code for First-Class Single Piece Indicia
         if ((activity.eq.'1060').or.(activity.eq.'2060').or.(activity.eq.'3060').or.
     &      (activity.eq.'4060')) then
            i7 = indicia(f136)
         else
            i7 = nw             !  Non First-Class Single Piece are given an arbitrary indicia code
         end if

         read(f9250,'(f10.2)') tvalue !  F9250

         if ((i2.gt.0).and.(i3.gt.0).and.
     +      (i4.gt.0).and.(i5.gt.0).and.(i6.gt.0).and.(i7.gt.0)) then
            write (30,31) i2, i3, i5, i6, i9, i7, i4, tvalue
            countg = countg + 1
            if (i3.eq.2) then   ! clk/mh
               cost_clk = cost_clk + tvalue
               if ((i4.le.12).and.(i4.ne.10).and.(i4.ne.11)) then
                  cost_mp = cost_mp + tvalue
               end if
            end if
         else
            if (f9806(1:1).eq.'9') then ! first digit of activity code f262
               countlv = countlv + 1
            else if (f257(2:2).eq.'4') then ! second digit of da code f257
               countsp = countsp + 1
            else if (f264.eq.'K') then ! cag k tally  f264
               countk = countk + 1
            else if (i2.eq.0) then
               print *,' invalid cagfin = ', fincag, ' fin = ', f2,
     &            ' roster ', f257, ' op code ', f260
               countf = countf + 1
            else
               print *,'    indexes = ',i2,i3,i4,i5,i6,i8
               print*, ' f260 = ', f260, ' f261 = ', f261
               counto = counto + 1
            end if
         end if
      end do
```

```
100   print *,' read exit code = ',ier
      print *,' number of records written to encdata = ',countg
      print *,' number of leave records excluded     = ',countlv
      print *,' number of spec. delivery excluded    = ',countsp
      print *,' number of CAG K records excluded     = ',countk
      print *,' number of invalid finance numbers    = ',countf
      print *,' number of other records excluded     = ',counto

      print*, 'Total valid clk/mh costs = ', cost_clk
      print*, 'Total valid clk/mh mail proc costs = ', cost_mp

      end


C ------------------------------------------------------------------------------
C       desigind
C
C       assigns index of 1-6 based on roster designation

        function    desigind(char)

        integer*4       desigind
        character*2     char

        if ((char.eq.' 9').or.
     +      (char.eq.'09').or.
     +      (char.eq.'19')) then
           desigind = 1
        else if (char.eq.'11') then
           desigind = 2
        else if ((char.eq.'31').or.
     +           (char.eq.'41').or.
     +           (char.eq.'61').or.
     &           (char.eq.'81')) then
           desigind = 2
        else if ((char.eq.'12').or.
     +           (char.eq.'32').or.
     +           (char.eq.'42').or.
     +           (char.eq.'62').or.
     &           (char.eq.'82')) then
           desigind = 2
        else if (char.eq.'13') then
           desigind = 3
        else if ((char.eq.'33').or.
     +           (char.eq.'43').or.
     +           (char.eq.'63').or.
     &           (char.eq.'83')) then
           desigind = 3
        else
           desigind = -1
        end if

        return
        end
C ------------------------------------------------------------------------------
C     bfind
C
C     returns index for basic function

        function    bfind(char)

        integer*4       bfind
        character*1     char

        if (char.eq.'1') then
           bfind = 1
        else if (char.eq.'2') then
           bfind = 2
        else if (char.eq.'3') then
           bfind = 3
        else if (char.eq.'5') then
           bfind = 4
        else
           bfind = -1
        end if

        return
        end
C ------------------------------------------------------------------------
C  orind
```

```
c
c  returns index value for operation or route code

      function    orind(char)

      integer*4      orind
      character*2    char

      if (char.eq.'00') then
         orind = 1
      else if (char.eq.'01') then
         orind = 2
      else if (char.eq.'02') then
         orind = 3
      else if (char.eq.'03') then
         orind = 4
      else if (char.eq.'04') then
         orind = 5
      else if ((char.eq.'05').or.
     +           ((char.ge.'11').and.(char.le.'13')).or.
     +           (char.eq.'15').or.
     +           (char.eq.'16').or.
     +           ((char.ge.'19').and.(char.le.'21')).or.
     +           ((char.ge.'27').and.(char.le.'29'))) then
         orind = 6
      else if ((char.eq.'06').or.
     +           (char.eq.'18').or.
     +           (char.eq.'22').or.
     +           (char.eq.'23')) then
         orind = 7
      else if (char.eq.'07') then
         orind = 8
      else if (char.eq.'08') then
         orind = 9
      else if ((char.eq.'09').or.
     +           (char.eq.'24').or.
     +           (char.eq.'25').or.
     +           (char.eq.'26')) then
         orind = 10
      else if ((char.eq.'10').or.
     +           (char.eq.'17')) then
         orind = 11
      else if (char.eq.'14') then
         orind = 12
      else if (char.eq.'71') then
         orind = 1
      else if (char.eq.'73') then
         orind = 2
      else if (char.eq.'75') then
         orind = 3
      else if (char.eq.'77') then
         orind = 4
      else if (char.eq.'78') then
         orind = 5
      else if (char.eq.'80') then
         orind = 6
      else if (char.eq.'82') then
         orind = 7
      else if (char.eq.'83') then
         orind = 8
      else if (char.eq.'84') then
         orind = 9
      else if (char.eq.'85') then
         orind = 10
      else if (char.eq.'86') then
         orind = 11
      else if (char.eq.'87') then
         orind = 12
      else if (char.eq.'88') then
         orind = 13
      else if (char.eq.'89') then
         orind = 14
      else if (char.eq.'90') then
         orind = 15
      else if (char.eq.'98') then
         orind = 16
      else if (char.eq.'99') then
         orind = 17
      else
         orind = -1
```

```
      end if

      return
      end

c -------------------------------------------------------------------------------
c     indicia

c     Assigns First-Class Single Piece indicia

      function indicia(f136)

      character*1  f136
      integer*4    indicia

      indicia = 0

      if ((f136.ge.'A').and.(f136.le.'C')) then
         indicia = 1            !  Stamped
      else if ((f136.ge.'D').and.(f136.le.'E')) then
         indicia = 2            !  Metered
      else
         indicia = 3            !  Other
      end if

      return
      end
```

```
% Name: Encdata.sm
% Sort encoded IOCS tally info
i      file is 'encdata', recs are data sensitive upto 30 chars.
ou  ut file is 'encdata.s', recs are data sensitive  upto 30 chars.
sort.
end.
```

```
      PROGRAM liocatt

C     PURPOSE: Allocate costs to raw tallies, and performs LIOCATT mixed mail cost distribution

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   nlev1a, nlev1b, nfun
      integer*4   nlev2a, nlev2b, nlev3a, nlev3b

      character*3 runtype

      logical     dofun

      call  getarg(1,runtype)
      if (runtype.eq.'fun') then
         dofun = .true.
      else
         dofun = .false.
      end if

      call  fillmixmap          ! load mixed mail distribution map and activity codes
      call  loaddata            ! load encdata.s (from encode_wgt.f, encdata.sm)
      call  fungroup(nfun)      ! form function groups from operations
      call  sortcost(nfun)      ! sort records for level1
      call  level1(nfun,nlev1a,nlev1b) ! level 1 indirect cost allocation
      call  level2(nlev1a,nlev2a,nlev2b) ! level 2 indirect cost allocation
      call  sortlev2a(nlev2a)   ! sort records for level 3
      call  level3(nlev2a,nlev3a,nlev3b) ! level 3 indirect cost allocation
      call  report(nlev1b,nlev2b,nlev3a,nlev3b) ! write results to file
      print *,' ************************************************** '
      print *,' Fiscal year 2000 completed '
      print *,' Total number of records = ', nrec
      print *,' Number of records after function creation = ',nfun
      print *,' Number of level 1 records = ',nlev1a,', ',nlev1b
      print *,' Number of level 2 records = ',nlev2a,', ',nlev2b
      print *,' Number of level 3 records = ',nlev3a,', ',nlev3b
      print *,' ************************************************** '

      end

C -------------------------------------------------
```

```
      subroutine fillmixmap

C     PURPOSE: Read the mixed mail codes and produce map for distributing
C              mixed mail codes to appropriate direct activity codes

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   i, j, ind
      integer*4   ier
      integer*4   searchc

      character*4 mmcodes(nummix)
      character*4 codes(20)

      logical  flag

      if (debug) print *,' Enter subroutine fillmixmap '

      ier = 0

c     Map of activity codes
      open(16,file='activity00.ecr.all',iostat=ier )
17    format(a4)
      do i = 1,numact
         read(16,17) acodes(i)
      end do

C       initialize count array (number of direct keys indirect code is distributed accross)
      do i =1 , nummix
         count(i) = 0
      end do

c     Map of mixed mail activity codes
      open(18,file='mmcodes.intl')
1     format(a4)
      do i = 1,nummix
         read(18,19) mmcodes(i)
      end do

c     Maps mixed mail codes to direct activity codes
      open(20,file='mxmail.all.ecr')
21    format(20a4)

      do while (ier.eq.0)
         read (20,21,iostat=ier,end=100) codes
         i = searchc(mmcodes,nummix,codes(1))
         if (i.gt.0) then
            flag = .true.
            ind = 1
            do while ((flag).and.(ind.lt.20))
               ind = ind + 1
               if (codes(ind).ne.'   0') then
                  j = searchc(acodes,numact,codes(ind))
                  if (j.gt.0) then
                     count(i) = count(i) + 1
                     mixmap(count(i),i) = j
                  else
                     print *,' Direct mail code did not map ',codes(ind)
                  end if
               else
                  flag = .false.
               end if
            end do
         else
            print *,' Mixed mail code did not map ',codes(1)
         end if
      end do
100   print *,' read exit code = ',ier

C     Fill mix_to_act array

      do i = 1,nummix
         mix_to_act(i) = searchc(acodes,numact,mmcodes(i))
      end do

      if (debug) print *,' exiting fillmixmap '

      close (16)
```

```
close (18)
close (20)

return
end
```

```
      subroutine loaddata

c     Purpose:  To read in coded data set - encode_wgt.f, encdata.sm

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   j
      integer*2   ioff, iact, ibf, iw, ifun, ipig, iocc
      integer*4   ier/0/
      character*14   datum, ldatum
      real*8         cost, lcost, tcost


      open (20,file='encdata.s') !  Sorted data set
21    format(i2,i1,i1,i3,i2,i3,i2,f11.2)

      lcost = 0.0
      tcost = 0.0
      ldatum = ' '
      j = 0
      nrec = 0
      do while (ier.eq.0)
         read (20,21,iostat=ier,end=100) ioff,iocc,ibf,iact,ipig,iw,ifun, cost
         write (datum,'(7a2)') ioff,iocc,ibf,iact,iw,ipig,ifun
         if ((datum.ne.ldatum).and.(j.ne.0)) then
            nrec = nrec + 1
            if (nrec.le.maxcost) then
               write (costbuf2(nrec),'(a14,a8)') ldatum, tcost
            else
               print *,' maxcost exceeded in loaddata '
               stop
            end if
            ldatum = datum
            tcost = cost
         else
            if (j.eq.0) then
               j = 1
               ldatum = datum
            end if
            tcost = tcost + cost
         end if
      end do
100   if (debug) print *,' Read exit of encdata = ',ier,', nrec = ',nrec
      close (20)
      return
      end
```

```
      subroutine fungroup(numout)

C     PURPOSE: Add clerk/mail handler records up into functional groups from operation codes.

      IMPLICIT NONE

      include  'liocatt.h'

      real*8       original_costs(numopr)
      real*8       fun_costs(numfun)
      real*8       cost

      integer*2    opr_to_fun(numfun,numopr)
      integer*2    cofg, cpay, copr, cbf, cact, cw, cpig
      integer*2    ofg, pay, opr, fun, bf, act, w, pig
      integer*4    i, j, numout
      integer*4    ier/0/
      integer*4    indb1, indb2

      logical      same

      if (debug) print *,' entering fungroup.f77 '

C     Fill opr_to_fun array

      open(20,file='operrtemap')
21    format(45i3)
      do i = 1,numfun
         read (20,21) (opr_to_fun(i,j),j=1,numopr)
      end do

C     open input and output file

C     Collect data for a office, pay category cell.
C     1.  Collapse over quarter
C     2.  If pay category is clerk/mailhandler create functional groups
C     3.  Output for regular processing in level1 - level3

31    format(7a2,a8)

C     Read first record of first group

      same = .true.
      indb1 = 1
      indb2 = 0
      do opr = 1, numopr
         original_costs(opr) = 0.0
      end do

      read (costbuf2(indb1),31) cofg, cpay, cbf, cact, cw, cpig, copr, cost
      original_costs(copr) = original_costs(copr) + cost

      do while (ier.eq.0)

C     Read rest of cost group
         do while (same)
            indb1 = indb1 + 1
            if (indb1.gt.nrec) then
               ier = -1
               goto 100
            end if
            read (costbuf2(indb1),31) ofg, pay, bf, act, w, pig, opr, cost
            if ((ofg.eq.cofg).and.(pay.eq.cpay).and.(bf.eq.cbf).and.
     +          (act.eq.cact).and.(w.eq.cw).and.(pig.eq.cpig)) then
               original_costs(opr) = original_costs(opr) + cost
            else
               copr = opr
               same = .false.
            end if
         end do
         if (ier.ne.0) print *,' Read exit code = ',ier

C     Sum operation codes into function groups if clerk/mailhandler

         if (cpay.eq.2) then
            do fun = 1,numfun
               fun_costs(fun) = 0.
               do i = 2,opr_to_fun(fun,1)
                  fun_costs(fun) = fun_costs(fun) +
     +                 original_costs(opr_to_fun(fun,i))
```

```
            end do
         end do
      end if

C     Output data from original costs if supervisor or carrier,
C     and from fun costs if clerk/mailhandler

      if ((cpay.eq.1).or.(cpay.eq.3)) then
         do opr = 1, numopr
            if (original_costs(opr).gt.0) then
               indb2 = indb2 + 1
               if (indb2.gt.maxcost) then
                  print *,' maxcost exceeded on write to costbuf1 in fungroup '
                  stop
               end if
               write (costbuf1(indb2),31) cofg, cpay, opr, cbf, cact, cw, cpig,
     +            original_costs(opr)
            end if
         end do
      else if (cpay.eq.2) then
         do fun = 1,numfun
            if (fun_costs(fun).gt.0) then
               indb2 = indb2 + 1
               if (indb2.gt.maxcost) then
                  print *,' maxcost exceeded on write to costbuf1 in fungroup '
                  stop
               end if
               write (costbuf1(indb2),31) cofg, cpay, fun, cbf, cact, cw, cpig,
     +            fun_costs(fun)
            end if
         end do
      end if

C     Set up next group from last record read

      same = .true.
      do opr = 1, numopr
         original_costs(opr) = 0.0
      end do
      cofg = ofg
      cpay = pay
      cbf = bf
      cact = act
      cw = w
      cpig = pig
      original_costs(copr) = original_costs(copr) + cost

      end do

      if (debug) print *,' number of records written to costbuf1 = ',indb2
      numout = indb2

      return
      end

C -------------------------------------------------------------------------
```

```
      subroutine sortcost(n)

c     Purpose:  Sorts records for level 1 cost distribution

      implicit none

      include 'liocatt.h'

      integer*4   i, j, l, n, ir
      character*22   rra

      if (debug) print *,' entering sortcost '

      l=n/2+1
      ir=n
10    continue

      if(l.gt.1)then
         l=l-1
         rra=costbuf1(l)
      else
         rra=costbuf1(ir)
         costbuf1(ir)=costbuf1(1)
         ir=ir-1
         if(ir.eq.1)then
            costbuf1(1)=rra
            if (debug) print *,' sortcost finished '
            return
         end if
      end if
      i=l
      j=l+l
20    if (j.le.ir) then
         if (j.lt.ir) then
            if (costbuf1(j)(1:14).lt.costbuf1(j+1)(1:14)) j=j+1
         end if
         if (rra(1:14).lt.costbuf1(j)(1:14)) then
            costbuf1(i)=costbuf1(j)
            i=j
            j=j+j
         else
            j=ir+1
         end if
         goto 20
      end if
      costbuf1(i)=rra
      goto 10

      end
```

```
      subroutine level1(numin,numouta,numoutb)

C     PURPOSE: Perform level one distribution of mixed mail costs to
C              direct mail codes.

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   maxgrp

      parameter   (maxgrp = 20000)

      integer*4   size1
      parameter   (size1 = numact*nummix)

      integer*2   group(4,maxgrp)
      integer*4   actptr(2,numact,numbf)
      integer*4   bfptr(2,numbf)

      real*8      original_costs(maxgrp)
      real*8      dist_mix_costs(npig,maxgrp)
      real*8      sum, cost, mixsum, chkmix

      integer*4   numin, numouta, numoutb
      integer*4   indin, inda, indb, indx
      integer*2   cofg, cpay, copr, cbf, cact, cw, cpig
      integer*2   ofg, pay, opr, bf, act, mixkey, w, ind, pig
      integer*4   i, j, k
      integer*4   ier/0/

      logical     same

      logical   debug1/.true./

      if (debug) print *,' Entering Level1.f77 '

31    format(7a2,a8)

C       Perform level one allocation
C
C       1. Collect matrix of costs for a office group, pay and cost group
C          category cell
C       2. Within each basic function cell :
C          a. For each mixed mail activity sum over direct mail costs it is
C             to be distributed to.
C          b. If sum is positive use shares to distribute mixed mail costs to
C             to direct mail costs.
C          c. If sum is zero add mixed costs to same cell for basic function 4
C          d. Output records for this bf cell.
C              1) all direct costs and all bf 4 costs to "a" file
C              2) all distributed direct cost to "b" file (bfs 1-3)

C       Set up matrices for first record

      do bf = 1, numbf          ! initialize actptr array
         do act = 1, numact
            actptr(1,act,bf) = 0
         end do
      end do
      do bf = 1, numbf
         bfptr(1,bf) = 0
      end do
      indin = 1
      inda = 0
      indb = 0

      same = .true.
      ind = 1

        Read first record of first cost group

      read (costbuf1(indin),31) cofg, cpay, copr, cbf, cact, cw, cpig, cost

      original_costs(ind) = cost
      group(1,ind) = cbf
      group(2,ind) = cact
      group(3,ind) = cw
      group(4,ind) = cpig
      actptr(1,cact,cbf) = ind
```

```
      actptr(2,cact,cbf) = 1
      bfptr(1,cbf) = ind
      bfptr(2,cbf) = 1
      ier = 0

      do while (ier.eq.0)

C          Read rest of cost group

         do while (same)
            indin = indin + 1
            if (indin.gt.numin) then
               ier = -1
               goto 100
            end if
            read (costbuf1(indin),31) ofg, pay, opr, bf, act, w, pig, cost
            if ((ofg.eq.cofg).and.(pay.eq.cpay).and.(opr.eq.copr)) then
               ind = ind + 1
               if (debug) then
                  if (ind.gt.maxgrp) then
                     print *,' maxgrp execeeded , ofg = ',ofg,' pay = ',pay,' opr = ',opr
                     ier = -999
                     goto 100
                  end if
               end if
               original_costs(ind) = cost
               group(1,ind) = bf
               group(2,ind) = act
               group(3,ind) = w
               group(4,ind) = pig
               if (actptr(1,act,bf).eq.0) then
                  actptr(1,act,bf) = ind
                  actptr(2,act,bf) = 1
               else
                  actptr(2,act,bf) = actptr(2,act,bf) + 1
               end if
               if (bfptr(1,bf).eq.0) then
                  bfptr(1,bf) = ind
                  bfptr(2,bf) = 1
               else
                  bfptr(2,bf) = bfptr(2,bf) + 1
               end if
            else
               same = .false.
               cbf = bf
               cact = act
               cw = w
               cpig = pig
            end if
         end do
100      if ((ier.ne.0).and.(debug))  print *,' Read exit code = ',ier
         do i = 1, ind
            do j = 1, npig
               dist_mix_costs(j,i) = 0.0
            end do
         end do
         if (debug1) then
            print *,' bfptr(1,1) = ',bfptr(1,1)
            print *,' bfptr(1,2) = ',bfptr(1,2)
            print *,' bfptr(1,3) = ',bfptr(1,3)
            print *,' bfptr(1,4) = ',bfptr(1,4)
         end if

C          Attempt to distribute mixed dollars into direct costs

         do bf = 1,3            ! do not attempt to distribute other at this level
            if (bfptr(1,bf).ne.0) then
               do i = 1,nummix  ! loop over mixed mail activities
                  if (actptr(1,mix_to_act(i),bf).gt.0) then
                     do indx = actptr(1,mix_to_act(i),bf),(actptr(1,mix_to_act(i),bf)+actptr(2,mix_to_act(i),bf)-1)
                        sum = 0
                        mixsum = original_costs(indx)
                        pig = group(4,indx)
                        do j = 1,count(i) ! sum over direct keys for mixed code
                           mixkey = mixmap(j,i)
                           if (actptr(1,mixkey,bf).ne.0) then
                              do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                                 sum = sum + original_costs(k)
                              end do
                           end if
```

```
                  end do
                  chkmix = 0
                  if (sum.gt.0) then ! distribute to direct codes
                     do j = 1,count(i)
                        mixkey = mixmap(j,i)
                        if (actptr(1,mixkey,bf).ne.0) then
                           do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                              dist_mix_costs(pig,k) = dist_mix_costs(pig,k) +
     +                           mixsum*(original_costs(k)/sum)
                              if (debug1) chkmix = chkmix +
     +                           mixsum*(original_costs(k)/sum)
                           end do
                        end if
                     end do
                     original_costs(indx) = 0.0
                     if (dabs(mixsum-chkmix).gt.1.0)
     &                  print *,' allocation failure, mixsum = ',mixsum,', chkmix = ',chkmix
                  end if
               end do
            end if
         end do
         do k = bfptr(1,bf), (bfptr(1,bf)+bfptr(2,bf)-1) ! Output records for this opr,bf  cell
            if (original_costs(k).gt.0.0) then
               inda = inda + 1
               write (costbuf2(inda),31) cofg, cpay, copr,
     +            group(1,k), group(2,k), group(3,k), group(4,k), original_costs(k)
            end if
            do pig = 1, npig
               if (dist_mix_costs(pig,k).gt.0.0) then
                  indb = indb + 1
                  if (indb.le.maxl1b) then
                     write (level1b(indb),31) cofg, cpay, copr, group(1,k),
     +                  group(2,k), group(3,k), pig, dist_mix_costs(pig,k)
                  else
                     print *,' maxl1b exceeded, inda = ', inda
                     stop
                  end if
               end if
            end do
         end do
      end if
   end do
   if (bfptr(1,4).gt.0) then
      do k = bfptr(1,4), ind ! Output all records for basic function "other"
         inda = inda + 1
         write (costbuf2(inda),31) cofg, cpay, copr, group(1,k),
     +      group(2,k), group(3,k), group(4,k), original_costs(k)
      end do
   end if

C     Set up next cost group using last record read

   if (ind.gt.(numbf*numact)) then
      do bf = 1, numbf    ! initialize actptr array
         do act = 1, numact
            actptr(1,act,bf) = 0
         end do
      end do
   else
      do k = 1, ind
         bf = group(1,k)
         act = group(2,k)
         actptr(1,act,bf) = 0
      end do
   end if
   do bf = 1, numbf
      bfptr(1,bf) = 0
   end do

   same = .true.
   ind = 1
   cofg = ofg
   copr = opr
   cpay = pay
   cpig = pig

   original_costs(ind) = cost
   group(1,ind) = cbf
   group(2,ind) = cact
   group(3,ind) = cw
```

```
        actptr(1,cact,cbf) = ind
        actptr(2,cact,cbf) = 1
        bfptr(1,cbf) = ind
        bfptr(2,cbf) = 1

      end do

      numouta = inda
      numoutb = indb

      if (debug) print *,' number of records written to costbuf2 = ',numouta
      if (debug) print *,' number of records written to level1b  = ',numoutb
      return
      end

C -------------------------------------------------------------------------
```

```
      subroutine level2(numin,numouta,numoutb)

C     PURPOSE: Perform level two distribution of mixed mail costs to
C              direct mail codes.

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   maxgrp

      parameter   (maxgrp = 20000)

      integer*4   size1
      parameter   (size1 = numact*nummix)

      integer*2   group(4,maxgrp)
      integer*4   actptr(2,numact,numbf)
      integer*4   bfptr(2,numbf)

      real*8      original_costs(maxgrp)
      real*8      dist_mix_costs(npig,maxgrp)
      real*8      sum, cost, mixsum, chkmix
      real*8      mixpig(npig)
      real*8      tdirect/0/  , tmixed/0/ , tmixeddist/0/ , tratio/0/

      integer*4   numin, numouta, numoutb
      integer*4   indin, inda, indb, indx
      integer*2   cofg, cpay, copr, cbf, cact, cw, cpig
      integer*2   ofg, pay, opr, bf, act, mixkey, w, ind, pig
      integer*2   bf4/4/
      integer*4   i, j, k
      integer*4   ier/0/

      logical     same , dist
      logical   debug1/.true./

      if (debug) print *,' Entering Level2.f77 '

C       open input and output file

31    format(7a2,a8)
45    format(7a2,a8)

C       Perform level two mixed cost allocation
C
C       1. Collect matrix of costs for a office group,pay category, and
C          operation/route code cell
C       2. Over all records in cell
C          a. For each mixed mail activity sum over basic functions to get
C             mixed mail costs.
C          c. For each mixed mail activity sum over direct mail costs it is
C             to be distributed to (over all basic functions).
C          b. If sum is positive use shares to distribute mixed mail costs to
C             to direct mail costs (all distributed mixed costs get basic
C             function 4.
C          d. Output records for this opr cell.
C               1) all direct costs costs to "a" file
C               2) all distributed direct cost to "b" file (bf 4)

C       Set up matrices for first record

      do bf = 1, numbf          ! initialize actptr array
         do act = 1, numact
            actptr(1,act,bf) = 0
         end do
      end do
      do bf = 1, numbf
         bfptr(1,bf) = 0
      end do
      indin = 1
      inda = 0
      indb = 0

      same = .true.
      ind = 1

C       Read first record of first cost group

      read (costbuf2(indin),31) cofg, cpay, copr, cbf, cact, cw, cpig, cost
```

```
      original_costs(ind) = cost
      group(1,ind) = cbf
      group(2,ind) = cact
      group(3,ind) = cw
      group(4,ind) = cpig
      actptr(1,cact,cbf) = ind
      actptr(2,cact,cbf) = 1
      bfptr(1,cbf) = ind
      bfptr(2,cbf) = 1
      ier = 0

      do while (ier.eq.0)

C          Read rest of cost group

         do while (same)
            indin = indin + 1
            if (indin.gt.numin) then
               ier = -1
               goto 100
            end if
            read (costbuf2(indin),31) ofg, pay, opr, bf, act, w, pig, cost
            if ((ofg.eq.cofg).and.(pay.eq.cpay).and.(opr.eq.copr)) then
               ind = ind + 1
               if (debug) then
                  if (ind.gt.maxgrp) then
                     print *,' maxgrp exeeeded , ofg = ',ofg,' pay = ',pay,' opr = ',opr
                     ier = -999
                     goto 100
                  end if
               end if
               original_costs(ind) = cost
               group(1,ind) = bf
               group(2,ind) = act
               group(3,ind) = w
               group(4,ind) = pig
               if (actptr(1,act,bf).eq.0) then
                  actptr(1,act,bf) = ind
                  actptr(2,act,bf) = 1
               else
                  actptr(2,act,bf) = actptr(2,act,bf) + 1
               end if
               if (bfptr(1,bf).eq.0) then
                  bfptr(1,bf) = ind
                  bfptr(2,bf) = 1
               else
                  bfptr(2,bf) = bfptr(2,bf) + 1
               end if
            else
               same = .false.
               cbf = bf
               cact = act
               cw = w
               cpig = pig
            end if
         end do
100      if ((ier.ne.0).and.(debug)) print *,' Read exit code = ',ier
         do i = 1, ind
            do j = 1, npig
               dist_mix_costs(j,i) = 0.0
            end do
         end do
         if (debug1) then
            print *,' bfptr(1,1) = ',bfptr(1,1)
            print *,' bfptr(1,2) = ',bfptr(1,2)
            print *,' bfptr(1,3) = ',bfptr(1,3)
            print *,' bfptr(1,4) = ',bfptr(1,4)
         end if

C          Attempt to distribute mixed dollars into direct costs

         do i = 1,nummix        ! loop over mixed mail activities
            do pig = 1, npig
               mixpig(pig) = 0.0
            end do
            do bf = 1, numbf
               if (actptr(1,mix_to_act(i),bf).gt.0) then
                  do indx = actptr(1,mix_to_act(i),bf),(actptr(1,mix_to_act(i),bf)+actptr(2,mix_to_act(i),bf)-1)
                     pig = group(4,indx)
```

```
                mixpig(pig) = mixpig(pig) + original_costs(indx)
              end do
            end if
          end do
          dist = .false.
          do pig = 1, npig
             if (mixpig(pig).gt.0.0) then
                sum = 0.0
                do bf = 1, numbf
                   do j = 1,count(i) ! sum over direct keys for mixed code
                      mixkey = mixmap(j,i)
                      if (actptr(1,mixkey,bf).ne.0) then
                         do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                            sum = sum + original_costs(k)
                         end do
                      end if
                   end do
                end do
                chkmix = 0
                if (sum.gt.0) then ! distribute to direct codes
                   do bf = 1, numbf
                      do j = 1,count(i)
                         mixkey = mixmap(j,i)
                         if (actptr(1,mixkey,bf).ne.0) then
                            do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                               dist_mix_costs(pig,k) = dist_mix_costs(pig,k) +
     +                            mixpig(pig)*(original_costs(k)/sum)
                               if (debug1) chkmix = chkmix +
     +                            mixpig(pig)*(original_costs(k)/sum)
                            end do
                         end if
                      end do
                   end do
                   dist = .true.
                   if (dabs(mixpig(pig)-chkmix).gt.1.0)
     &                 print *,' level2 allocation failure, mixsum = ',mixsum,', chkmix = ',chkmix
                end if
             end if
          end do
          if (dist) then
             do bf = 1, numbf
                if (actptr(1,mix_to_act(i),bf).gt.0) then
                   do indx = actptr(1,mix_to_act(i),bf),(actptr(1,mix_to_act(i),bf)+actptr(2,mix_to_act(i),bf)-1)
                      original_costs(indx) = 0.0
                   end do
                end if
             end do
          end if
       end do
       do k = 1, ind
          if (original_costs(k).gt.0.0) then
             inda = inda + 1
             write (costbuf1(inda),45) cofg, cpay, copr, group(1,k),
     +          group(2,k), group(3,k), group(4,k), original_costs(k)
          end if
          do pig = 1, npig
             if (dist_mix_costs(pig,k).gt.0.0) then
                indb = indb + 1
                if (indb.le.maxl2b) then
                   write (level2b(indb),45) cofg, cpay, copr, bf4, group(2,k),
     +                group(3,k), pig, dist_mix_costs(pig,k)
                else
                   print *,' maxl2b exceeded, inda = ',inda
                   stop ' fatal error '
                end if
             end if
          end do
       end do

C      Set up next cost group using last record read

       if (ind.gt.(numbf*numact)) then
          do bf = 1, numbf    ! initialize actptr array
             do act = 1, numact
                actptr(1,act,bf) = 0
             end do
          end do
       else
          do k = 1, ind
             bf = group(1,k)
```

```
            act = group(2,k)
            actptr(1,act,bf) = 0
         end do
      end if
      do bf = 1, numbf
         bfptr(1,bf) = 0
      end do

      same = .true.
      ind = 1
      cofg = ofg
      copr = opr
      cpay = pay

      original_costs(ind) = cost
      group(1,ind) = cbf
      group(2,ind) = cact
      group(3,ind) = cw
      group(4,ind) = cpig
      actptr(1,cact,cbf) = ind
      actptr(2,cact,cbf) = 1
      bfptr(1,cbf) = ind
      bfptr(2,cbf) = 1

   end do

   numouta = inda
   numoutb = indb

   if (debug) print *,' number of records written to costbuf1 = ',numouta
   if (debug) print *,' number of records written to level2b  = ',numoutb
   return
   end

C ------------------------------------------------------------------------------
```

```
      subroutine sortlev2a(n)

C     Purpose:  To sort records for level 3 cost distribution

      implicit none

      include 'liocatt.h'

      integer*4   i, j, l, n, ir
      character*22   rra

      if (debug) print *,' entering sortlev2a.f77 '

      l=n/2+1
      ir=n
10    continue

      if(l.gt.1)then
         l=l-1
         rra=costbuf1(l)
      else
         rra=costbuf1(ir)
         costbuf1(ir)=costbuf1(1)
         ir=ir-1
         if(ir.eq.1)then
            costbuf1(1)=rra
            if (debug) print *,' exiting sortlev2a '
            return
         end if
      end if
      i=l
      j=l+l
20    if (j.le.ir) then
         if (j.lt.ir) then
            if (costbuf1(j)(3:12).lt.costbuf1(j+1)(3:12)) j=j+1
         end if
         if (rra(3:12).lt.costbuf1(j)(3:12)) then
            costbuf1(i)=costbuf1(j)
            i=j
            j=j+j
         else
            j=ir+1
         end if
         goto 20
      end if
      costbuf1(i)=rra
      goto 10
      end
```

```
      subroutine level3(numin,numouta,numoutb)

C     PURPOSE: Perform level three distribution of mixed mail costs to direct mail codes.

      IMPLICIT NONE

      include 'liocatt.h'

      integer*4   maxgrp

      parameter   (maxgrp = 200000)

      integer*4   size1
      parameter   (size1 = numact*nummix)

      integer*2   group(4,maxgrp)
      integer*4   actptr(2,numact,numbf)
      integer*4   bfptr(2,numbf)

      real*8      original_costs(maxgrp)
      real*8      dist_mix_costs(npig,maxgrp)
      real*8      sum, cost, mixsum, chkmix
      real*8      mixed(npig)

      integer*4   numin, numouta, numoutb
      integer*4   indin, inda, indb, indx
      integer*2   cpay, copr, cbf, cact, cw, cpig
      integer*2   pay, opr, bf, act, mixkey, w, ind, pig
      integer*2   bf4/4/
      integer*4   i, j, k
      integer*4   ier/0/

      logical     same , dist
      logical   debug1/.true./

      if (debug) print *,' entering level3.f77 '

C       open input and output file

31    format(2x,6a2,a8)
45    format(6a2,a8)

C       Perform level three mixed cost allocation
C
C       1. Collect matrix of costs for a pay category,operation/route code cell
C       2. Over all records in cell
C          a. For each mixed mail activity sum over basic functions to get
C             mixed mail costs.
C          c. For each mixed mail activity sum over direct mail costs it is
C             to be distributed to (over all basic functions).
C          b. If sum is positive use shares to distribute mixed mail costs to
C             to direct mail costs (all distributed mixed costs get basic
C             function 4.
C          d. Output records for this opr cell.
C              1) all direct costs costs to "a" file
C              2) all distributed direct cost to "b" file (bf 4)

C       Set up matrices for first record

      do bf = 1, numbf          ! initialize actptr array
         do act = 1, numact
            actptr(1,act,bf) = 0
         end do
      end do
      do bf = 1, numbf
         bfptr(1,bf) = 0
      end do
      indin = 1
      inda = 0
      indb = 0

      same = .true.
      ind = 1

C       Read first record of first cost group

      read (costbuf1(indin),31) cpay, copr, cbf, cact, cw, cpig, cost

      original_costs(ind) = cost
      group(1,ind) = cbf
```

```
      group(2,ind) = cact
      group(3,ind) = cw
      group(4,ind) = cpig
      actptr(1,cact,cbf) = ind
      actptr(2,cact,cbf) = 1
      bfptr(1,cbf) = ind
      bfptr(2,cbf) = 1
      ier = 0

      do while (ier.eq.0)

C          Read rest of cost group

         do while (same)
            indin = indin + 1
            if (indin.gt.numin) then
               ier = -1
               goto 100
            end if
            read (costbuf1(indin),31) pay, opr, bf, act, w, pig, cost
            if ((pay.eq.cpay).and.(opr.eq.copr)) then
               if ((bf.eq.group(1,ind)).and.
     +             (act.eq.group(2,ind)).and.
     +             (w.eq.group(3,ind)).and.
     +             (pig.eq.group(4,ind))) then
                  original_costs(ind) = original_costs(ind) + cost
               else
                  ind = ind + 1
                  if (debug) then
                     if (ind.gt.maxgrp)
     +                  print *,' maxgroup exceeded, pay = ',pay,', opr = ',opr
                  end if
                  original_costs(ind) = cost
                  group(1,ind) = bf
                  group(2,ind) = act
                  group(3,ind) = w
                  group(4,ind) = pig
                  if (actptr(1,act,bf).eq.0) then
                     actptr(1,act,bf) = ind
                     actptr(2,act,bf) = 1
                  else
                     actptr(2,act,bf) = actptr(2,act,bf) + 1
                  end if
                  if (bfptr(1,bf).eq.0) then
                     bfptr(1,bf) = ind
                     bfptr(2,bf) = 1
                  else
                     bfptr(2,bf) = bfptr(2,bf) + 1
                  end if
               end if
            else
               same = .false.
               cbf = bf
               cact = act
               cw = w
               cpig = pig
            end if
         end do
100      if ((ier.ne.0).and.debug) print *,' Read exit code = ',ier
         do i = 1, ind
            do pig = 1, npig
               dist_mix_costs(pig,i) = 0.0
            end do
         end do
         if (debug1) then
            print *,' bfptr(1,1) = ',bfptr(1,1)
            print *,' bfptr(1,2) = ',bfptr(1,2)
            print *,' bfptr(1,3) = ',bfptr(1,3)
            print *,' bfptr(1,4) = ',bfptr(1,4)
         end if

C          Attempt to distribute mixed dollars into direct costs

         do i = 1,nummix          ! loop over mixed mail activities
            do pig = 1, npig
               mixed(pig) = 0.0
            end do
            do bf = 1, numbf
               if (actptr(1,mix_to_act(i),bf).gt.0) then
                  do indx = actptr(1,mix_to_act(i),bf),(actptr(1,mix_to_act(i),bf)+actptr(2,mix_to_act(i),bf)-1)
```

```
                  pig = group(4,indx)
                  mixed(pig) = mixed(pig) + original_costs(indx)
               end do
            end if
         end do
         dist = .false.
         do pig = 1, npig
            if (mixed(pig).gt.0.0) then
               sum = 0.0
               do bf = 1, numbf
                  do j = 1,count(i) ! sum over direct keys for mixed code
                     mixkey = mixmap(j,i)
                     if (actptr(1,mixkey,bf).ne.0) then
                        do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                           sum = sum + original_costs(k)
                        end do
                     end if
                  end do
               end do
               chkmix = 0
               if (sum.gt.0) then ! distribute to direct codes
                  do bf = 1, numbf
                     do j = 1,count(i)
                        mixkey = mixmap(j,i)
                        if (actptr(1,mixkey,bf).ne.0) then
                           do k = actptr(1,mixkey,bf),(actptr(1,mixkey,bf)+actptr(2,mixkey,bf)-1)
                              dist_mix_costs(pig,k) = dist_mix_costs(pig,k) +
     +                           mixed(pig)*(original_costs(k)/sum)
                              if (debug1) chkmix = chkmix +
     +                           mixed(pig)*(original_costs(k)/sum)
                           end do
                        end if
                     end do
                     if (actptr(1,mix_to_act(i),bf).ne.0)
     +                  original_costs(actptr(1,mix_to_act(i),bf)) = 0.0
                  end do
                  dist = .true.
               end if
               if (dabs(mixed(pig)-chkmix).gt.1.0)
     &            print *,' level 3 allocation failure, mixsum = ',mixsum,', chkmix = ',chkmix
            end if
         end do
         if (dist) then
            do bf = 1, numbf
               if (actptr(1,mix_to_act(i),bf).gt.0) then
                  do indx = actptr(1,mix_to_act(i),bf),(actptr(1,mix_to_act(i),bf)+actptr(2,mix_to_act(i),bf)-1)
                     original_costs(indx) = 0.0
                  end do
               end if
            end do
         end if
      end do
      do k = 1, ind
         if (original_costs(k).gt.0.0) then
            inda = inda + 1
            write (costbuf2(inda),45) cpay, copr, group(1,k),
     +         group(2,k), group(3,k), group(4,k), original_costs(k)
         end if
         do pig = 1, npig
            if (dist_mix_costs(pig,k).gt.0.0) then
               indb = indb + 1
               if (indb.le.maxl3b) then
                  write (level3b(indb),45) cpay, copr, bf4, group(2,k),
     +               group(3,k), pig, dist_mix_costs(pig,k)
               else
                  print *,' maxl3b exceeded inda = ',inda
                  stop
               end if
            end if
         end do
      end do

C     Set up next cost group using last record read

      if (ind.gt.(numbf*numact)) then
         do bf = 1, numbf    ! initialize actptr array
            do act = 1, numact
               actptr(1,act,bf) = 0
            end do
         end do
```

```
      else
         do k = 1, ind
            bf = group(1,k)
            act = group(2,k)
            actptr(1,act,bf) = 0
         end do
      end if
      do bf = 1, numbf
         bfptr(1,bf) = 0
      end do

      same = .true.
      ind = 1
      cpay = pay
      copr = opr

      original_costs(ind) = cost
      group(1,ind) = cbf
      group(2,ind) = cact
      group(3,ind) = cw
      group(4,ind) = cpig
      actptr(1,cact,cbf) = ind
      actptr(2,cact,cbf) = 1
      bfptr(1,cbf) = ind
      bfptr(2,cbf) = 1

   end do

   numouta = inda
   numoutb = indb

   if (debug) print *,' number of records written to level3a = ',numouta
   if (debug) print *,' number of records written to level3b = ',numoutb
   return
   end
C  ------------------------------------------------------------------------
```

```
      subroutine report(nlev1b,nlev2b,nlev3a,nlev3b)

C     PURPOSE: Produce report on results of Liocatt by activity for city carriers by
C              activity code and operation/route code

      IMPLICIT NONE

      include 'liocatt.h'

      real*8       data(numw,numact)
      real*8       indata

      integer*4    nlev1b, nlev2b, nlev3a, nlev3b, irun
      integer*2    ofg, opr, act, pay, bf, w, pig
      integer*4    i, j
      integer*4    ier/0/

      character*3 buffer

      logical flag/.false./

      if (debug) then
         print *,' in subroutine report '
         print *, ' nlev1b = ',nlev1b,' nlev2b = ',nlev2b
         print *, ' nlev3a = ',nlev3a,' nlev3b = ',nlev3b
      end if
      do i = 1, numact
         do j = 1, numw
            data(j,i) = 0.0
         end do
      end do

C       Open input files

25    format(7a2,a8)
35    format(6a2,a8)

      open(20,file='level1b')
      open(21,file='level2b')
      open(22,file='level3a')
      open(23,file='level3b')
45    format(i2.2,i1,i2.2,i1,i3.3,i3.3,i2,f13.1)
46    format(i1,i2.2,i1,i3.3,i3.3,i2,f13.1)

C       Assemble data for report

      do i = 1, nlev1b
         read (level1b(i),25) ofg,pay,opr,bf,act,w,pig,indata
         write (20,45) ofg,pay,opr,bf,act,w,pig,indata
      end do
      do i = 1, nlev2b
         read (level2b(i),25) ofg,pay,opr,bf,act,w,pig,indata
         write (21,45) ofg,pay,opr,bf,act,w,pig,indata
      end do
      do i = 1, nlev3a
         read (costbuf2(i),35) pay,opr,bf,act,w,pig,indata
         write (22,46) pay,opr,bf,act,w,pig,indata
      end do
      do i = 1, nlev3b
         read (level3b(i),35) pay,opr,bf,act,w,pig,indata
         write (23,46) pay,opr,bf,act,w,pig,indata
      end do

      return
      end

C -------------------------------------------------------------------------------
```

```
      PROGRAM rpt_ind

C     Purpose:  To summarize city carrier in-office costs by indicia and shape for
C               First-Class Single Piece

      IMPLICIT NONE

      integer*4   numfun, numact, nopr, nshp, ncl

      parameter   (numfun = 3)   ! number of indicia
      parameter   (numact = 501) ! number of activity codes
      parameter   (nopr = 12)    !  number of operations
      parameter   (ncl = 243)    !  number of activity codes
      parameter   (nshp = 4)     !  number of shapes

      integer*4   is, shape, ishp, shp(numact), icl
      integer*4   pay, opr, bf, act, w
      integer*4   unit, i, ier
      integer*4   class(numact)

      real*8      carrier(ncl,nshp,numfun)
      real*8      indata

      character*4 acodes(numact)
      character*9 classes(ncl)
      character*4 shapetype(nshp) /'1Ltr ','2Flt ','3Pcl ','4None'/

      ier = 0

c     Map of activity codes and codes to corresponding subclass
      open(16,file='activity00.ecr.all')
17    format(a4,i6)

      do i = 1,numact
         read(16,17) acodes(i), class(i)
         is = shape(acodes(i))
         shp(i) = is
      end do
      close(16)

c     Map of subclasses
      open(16,file='classes_ecr.old')
18    format(a9)
      do i = 1, ncl
         read(16,18) classes(i)
      end do

      close(16)

c     Initialize matrices
      do icl = 1, ncl
         do ishp = 1, nshp
            do i = 1, numfun
               carrier(icl,ishp,i) = 0.0
            end do
         end do
      end do

c     Open files of LIOCATT results
      open(20,file='level1b')
      open(21,file='level2b')
25    format(2x,i1,i2.2,i1,2i3.3,2x,f13.1)
      open(30,file='level3a')
      open(31,file='level3b')
35    format(i1,i2.2,i1,2i3.3,2x,f13.1)

      do unit = 20,21
         do while (ier.eq.0)
            read (unit,25,iostat=ier,end=100) pay,opr,bf,act,w,indata
            icl = class(act)
            ishp = shp(act)
            if (icl.eq.2) icl = 1 !  Combine First-Class Single Piece
            if (pay.eq.3) then
               if (icl.gt.0) then
                  if (ishp.gt.0) then
                     carrier(icl,ishp,w) = carrier(icl,ishp,w) + indata/1000.
                  else
                     print*, 'Invalid shape ', acodes(act), shp(act)
                  end if
               else
```

```
               print*, 'Invalid class assignment ', acodes(act), class(act)
            end if
         end if
      end do
100   print *,' Read exit of unit ',unit,' = ',ier
      ier = 0
   end do

   ier = 0

   do unit = 30,31
      do while (ier.eq.0)
         read (unit,35,iostat=ier,end=101) pay,opr,bf,act,w,indata
         icl = class(act)
         ishp = shp(act)
         if (icl.eq.2) icl = 1 !  Combine First-Class Single Piece
         if (pay.eq.3) then
            if (icl.gt.0) then
               if (ishp.gt.0) then
                  carrier(icl,ishp,w) = carrier(icl,ishp,w) + indata/1000.
               else
                  print*, 'Invalid shape ', acodes(act), shp(act)
               end if
            else
               print*, 'Invalid class assignment ', acodes(act), class(act)
            end if
         end if
      end do
101   print *,' Read exit of unit ',unit,' = ',ier
      ier = 0
   end do

   open(45,file='car001SP_ind.csv')
41 format(i3,',',a9,',',i2,',',a4,',',3f12.0)
   do icl = 1, 1              !  First-Class Single Piece
      do ishp = 1, 1          !  Letters only
         write (45,41) icl, classes(icl), ishp, shapetype(ishp),
     *      (carrier(icl,ishp,i), i = 1, numfun)
      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') then
      shape = 3               !  IPPs
   else if (act(1:1).eq.'4') then
      shape = 3               !  Parcels
   else
      shape = 4               !  Other (special services)
   end if

   return
   end
```