Programming Exercises

29. *Hint*: Carefully inspect the variable names and labels for each data set. Consider tracking the data set source in order to create the country name variable.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

DATA australia;

SET sasdata.australia;

Rank+1;

RUN;

DATA brazil;

SET sasdata.brazil (RENAME = (Menina = Girl

Menino = Boy));

Rank+1;

RUN;

DATA france;

SET sasdata.france (RENAME = (Fille = Girl

Garcon = Boy));

Rank+1;

RUN;

DATA india;

SET sasdata.india (RENAME = (Laraki = Girl

Laraka = Boy));

Rank+1;

RUN;

DATA russia;

SET sasdata.russia (RENAME = (Devushka = Girl

Malchik = Boy));

Rank+1;

RUN;

DATA unitedstates;

SET sasdata.unitedstates;

Rank+1;

RUN;

\*\* Part b);

DATA stack;

SET australia brazil france india russia unitedstates;

RUN;

PROC SORT DATA = stack;

BY Rank;

RUN;

\*\* Part c);

DATA interleave;

SET australia (IN = aus) brazil (IN = bra)

france (IN = fra) india (IN = ind)

russia (IN = rus) unitedstates (IN = usa);

BY Rank;

\*\* Part d);

IF aus = 1 then Country = 'Australia';

ELSE IF bra = 1 then Country = 'Brazil';

ELSE IF fra = 1 then Country = 'France';

ELSE IF ind = 1 then Country = 'India';

ELSE IF rus = 1 then Country = 'Russia';

ELSE IF usa = 1 then Country = 'USA';

RUN;

\*\* Part e);

\*\* There are 60 observations and four variables in the

final data set;

(sections 6.1, 6.2, 6.3, 6.12)

30. *Hint*: Think about which variables are common to both data sets even though they may not have the same variable names.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

PROC SORT DATA = sasdata.teachers OUT = teachersort;

BY TeacherScore CurriculumGrd;

RUN;

DATA rate;

MERGE sasdata.district (RENAME = (TS = TeacherScore

CG = CurriculumGrd))

teachersort (IN = Tdat);

BY TeacherScore CurriculumGrd;

\*\* Part b);

IF Tdat = 1;

RUN;

\*\* Part c);

PROC SORT DATA = rate;

BY Teacher;

RUN;

\*\* Part d);

\*\* There are 10 observations and four variables in the

final data set;

(sections 6.4, 6.11, 6.12)

31. *Hint*: For part b), consider which variables you want to keep from the old data set, which variable will define the new variable names, and which variable contains the data values. Be sure to sort the data before using a BY statement in a subsequent DATA or PROC step.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

PROC SORT DATA = sasdata.aveprices OUT = sortprices;

BY Commodity Year;

RUN;

PROC MEANS DATA = sortprices NOPRINT;

VAR Price;

BY Commodity Year;

OUTPUT OUT = avgs MEAN(Price) = Average;

RUN;

\*\* Part b);

PROC TRANSPOSE DATA = sortprices OUT = transprices;

VAR Price;

ID Month;

BY Commodity Year;

RUN;

\*\* Part c);

DATA combineprices;

MERGE avgs (KEEP = Commodity Year Average)

transprices;

BY Commodity Year;

RUN;

\*\* Part d);

DATA transegg transgas transmilk;

\*\* Part e);

SET combineprices (DROP = \_NAME\_ \_LABEL\_

RENAME = (\_1 = January \_2 = February

\_3 = March \_4 = April \_5 = May

\_6 = June \_7 = July \_8 = August

\_9 = September \_10 = October

\_11 = November \_12 = December));

IF Commodity = 'Egg' THEN OUTPUT transegg;

ELSE IF Commodity = 'Gas' THEN OUTPUT transgas;

ELSE IF Commodity = 'Milk' THEN OUTPUT transmilk;

RUN;

\*\* Part f);

\*\* There are 11 observations and 15 variables in each of

the three final data sets;

(sections 6.6, 6.9, 6.11, 6.14)

32. *Hint*: Think about procedures that can calculate medians and how you could combine those results with a SAS data set.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

PROC SORT DATA = sasdata.visits OUT = sortvisit;

BY ID;

RUN;

\*\* Part b);

PROC SORT DATA = sasdata.txgroup OUT = sorttxgroup

NODUPKEY DUPOUT = duplicates;

BY ID;

RUN;

PROC PRINT DATA = duplicates;

TITLE 'Duplicate Records';

RUN;

DATA identify;

MERGE sortvisit sorttxgroup;

BY ID;

RUN;

\*\* Part c);

PROC MEANS DATA = identify NOPRINT;

VAR B\_Cholesterol;

OUTPUT OUT = med MEDIAN(B\_Cholesterol) = MedChol;

RUN;

DATA compare (DROP = MedChol);

IF \_N\_ = 1 THEN SET med (KEEP = MedChol);

SET identify;

IF B\_Cholesterol <= MedChol THEN B\_Group = '<=Median';

ELSE B\_Group = '>Median';

RUN;

\*\* Part d);

DATA followup (DROP = i);

SET compare;

\*\* To output the baseline visit;

OUTPUT;

DO i = 1 to 3;

VisitDt = VisitDt + (i \* 30);

Visit = i;

\*\* To output the next three visits;

OUTPUT;

END;

FORMAT VisitDt DATE9.;

RUN;

\*\* Part e);

\*\* There are 2363 observations and seven variables in

the final data set called COMPARE;

\*\* There are 9452 observations and seven variables in

the final data set called FOLLOWUP;

(sections 6.5, 6.7, 6.10, 6.11)

33. *Hint*: The FIRST. and LAST. variables are created when you use a BY statement in a DATA step, even in a merge. Use FIRST. or LAST. to reset the counter variable.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

\*\* There are different ways to examine the sort order

such as using PROC PRINT or viewing the data in your

SAS session;

\*\* The USERS data set is sorted by ascending UserID;

\*\* The PROJECTS data set is sorted by ascending

ProjectID and then by ascending UserID;

\*\* Part b);

PROC SORT DATA = sasdata.projects OUT = projectsort;

BY UserID;

RUN;

\*\* Part c);

DATA complete incomplete (DROP = Total)

none (DROP = Total);

MERGE sasdata.users (IN = User)

projectsort (IN = Prj);

BY UserID;

\*\* Part d);

\*\* Initialize a counter to 0 for the first

occurance of a UserID;

IF FIRST.UserID THEN Total = 0;

IF User = 1 AND Prj = 0 THEN OUTPUT none;

ELSE DO;

IF EndDate ~= . THEN DO;

Total + 1;

OUTPUT complete;

END;

ELSE OUTPUT incomplete;

END;

RUN;

\*\* Part e);

\*\* There are 4127 observations and 11 variables in

the final data set called COMPLETE;

\*\* There are 3139 observations and 10 variables in

the final data set called INCOMPLETE;

\*\* There are seven observations and 10 variables in

the final data set called NONE;

(sections 6.5, 6.9, 6.11, 6.12, 6.15)

34. *Hint*: You may need more than one DATA step to combine the data and various statistics.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

PROC FREQ DATA = sasdata.iluvthe80s NOPRINT;

TABLE Band / OUT = totals;

RUN;

\*\* Part b);

\*\* Creating the output data set with median length per

genre can be accomplished with a CLASS or BY

statement in PROC MEANS;

PROC MEANS DATA = sasdata.iluvthe80s NOPRINT;

CLASS Genre;

VAR Length;

OUTPUT OUT = meds MEDIAN(Length) = MedLength;

RUN;

\*\* Part c);

PROC SORT DATA = sasdata.iluvthe80s OUT = bandsort;

BY Band;

RUN;

DATA oneband;

MERGE totals (KEEP = Band Count) bandsort;

BY Band;

RUN;

PROC SORT DATA = oneband;

BY Genre;

RUN;

DATA all (DROP = \_TYPE\_ \_FREQ\_);

\*\* The values of \_TYPE\_ in the meds data set are

different depending on if the CLASS or BY statement

was used in the PROC MEANS from part b);

MERGE oneband meds (WHERE = (\_TYPE\_ = 1));

BY Genre;

RUN;

PROC SORT DATA = all;

BY Band;

RUN;

\*\* Part d);

\*\* There are 94 bands and 17 genres;

\*\* There are 100 observations in the original data set

and final data set called ALL;

(sections 6.5, 6.6, 6.11, 6.13, 6.15)

35. *Hint*: Consider which procedures you can use to count the number of older and younger siblings. Use FIRST. and LAST. to find the youngest and oldest siblings.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

\*\* There are different ways to examine the sort order

such as using PROC PRINT or viewing the data in your

SAS session;

\*\* The SCHOOLSURVEY data set is sorted by ascending

StudentID;

\*\* Part b);

\*\* To combine the data sets in part c) the data must

be sorted by FamilyID. Using a SORT with a WHERE

statement will sort and create a data set at the

same time;

PROC SORT DATA = sasdata.schoolsurvey OUT = grade6;

WHERE (Grade = '6th');

BY Family\_ID;

RUN;

PROC SORT DATA = sasdata.schoolsurvey OUT = sortsurvey;

BY Family\_ID;

RUN;

\*\* Part c);

DATA all;

MERGE grade6 (KEEP = Family\_ID DOB Grade

RENAME = (DOB = DOB\_Sixth))

sortsurvey;

BY Family\_ID;

AgeDiff = YRDIF(DOB,DOB\_Sixth,'ACTUAL');

RUN;

\*\* Using DOB is easier and more accurate than using

Grade to compare the age of siblings;

\*\* Part d);

PROC FREQ DATA = all NOPRINT;

WHERE DOB > DOB\_Sixth;

TABLES Family\_ID / OUT = youngersibs;

RUN;

PROC FREQ DATA = all NOPRINT;

WHERE DOB < DOB\_Sixth;

TABLES Family\_ID / OUT = oldersibs;

RUN;

DATA numsibs;

MERGE grade6

youngersibs (KEEP = Family\_ID Count

RENAME = (Count = NumYoungerSibs))

oldersibs (KEEP = Family\_ID Count

RENAME = (Count = NumOlderSibs));

BY Family\_ID;

IF NumYoungerSibs = . THEN NumYoungerSibs = 0;

IF NumOlderSibs = . THEN NumOlderSibs = 0;

LABEL NumYoungerSibs = 'Number of Younger Siblings'

NumOlderSibs = 'Number of Older Siblings';

RUN;

\*\* Part e);

PROC FORMAT;

VALUE $sch 'RC' = 'Rachael Carson'

'GV' = 'Green Valley'

'RG' = 'Redwood Grove';

RUN;

\*\* The number of sixth graders is found in the NObs

column. The number of siblings is found in the Sum

column;

PROC MEANS DATA = numsibs SUM MAXDEC = 0;

CLASS School;

VAR NumYoungerSibs NumOlderSibs;

FORMAT School $sch.;

LABEL School = 'School';

TITLE 'Number of Sixth Graders and Siblings';

RUN;

\*\* Part f);

PROC SORT DATA = all;

BY Family\_ID AgeDiff;

RUN;

DATA youngest oldest;

SET all;

BY Family\_ID;

IF FIRST.Family\_ID AND Grade NE '6th' THEN

OUTPUT youngest;

ELSE IF LAST.Family\_ID AND Grade NE '6th' THEN

OUTPUT oldest;

RUN;

DATA sibages;

MERGE numsibs

youngest (KEEP = Family\_ID AgeDiff

RENAME = (AgeDiff = AgeDiffYoungest))

oldest (KEEP = Family\_ID AgeDiff

RENAME = (AgeDiff = AgeDiffOldest));

BY Family\_ID;

RUN;

\*\* Part g);

PROC MEANS DATA = sibages MEAN MIN MAX;

CLASS School;

VAR AgeDiffYoungest AgeDiffOldest;

FORMAT School $sch.;

LABEL School = 'School';

TITLE1 'Average Age Difference';

TITLE2 'Between Sixth Grader and Siblings';

RUN;

(sections 6.1, 6.5, 6.6, 6.9, 6.11, 6.13, 6.15)

36. *Hint*: Think about the best way to combine the two original data sets without creating a lot of missing data values. Back up the original FRIENDS data set using system commands.

*Answer*:

LIBNAME sasdata 'c:\MySASLib';

\*\* Part a);

\*\* There are different ways to examine the sort order

such as using PROC PRINT or viewing the data in your

SAS session;

\*\* The FRIENDS data set is sorted by ascending ID and

contains seven variables. All of these variables are

also in the NEWINFO data set;

\*\* The NEWINFO data set is not sorted and contains nine

variables. Of these, two variables (Donation and

Campaign) are unique to this data set;

\*\* Part b);

PROC SORT DATA = sasdata.newinfo OUT = newsort;

BY ID;

RUN;

DATA sasdata.friends;

UPDATE sasdata.friends

newsort (DROP = Donation Campaign);

BY ID;

RUN;

\*\* Part c);

\*\* This could also be accomplished with a CLASS

statement;

PROC MEANS DATA = newsort NOPRINT;

BY ID;

VAR Donation;

OUTPUT OUT = donortot SUM(Donation) = TotalDonations;

RUN;

DATA donations;

MERGE sasdata.friends donortot;

BY ID;

RUN;

PROC PRINT DATA = donations;

VAR ID FirstName LastName TotalDonations;

TITLE1 'Wildlife Rescue Center';

TITLE2 'Total Donations by Friend';

RUN;

(sections 6.6, 6.8, 6.11)