Lesson 6 Homework

1. Refer to HOCKEY data. Write a SAS program which calculates the number of games won, lost, and tied up to and including the current observation. Print the dataset with an appropriate format for the date. Don’t forget to change the score of the final game to Boston College 5, Ohio State 2 (do this in your code, don’t change the original file). The first few lines of output should be similar to this:

DATE TEAM CITY STATE OSU OPP W L T

10/10/97 Toronto Columbus Ohio 5 0 1 0 0

10/18/97 Miami Oxford Ohio 0 3 1 1 0

10/24/97 Merrimack Columbus Ohio 2 7 1 2 0

10/26/97 Merrimack Columbus Ohio 5 3 2 2 0

10/31/97 Clarkson Potsdam New York 1 1 2 2 1

**CODE-**

**data** hockey;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 4\Homework\hockey.csv' dlm=',' dsd missover firstobs=**2**;

INPUT Game\_Date : MMDDYY10.

Team : $27.

City : $17.

State : $13.

OSU

OPP;

FORMAT Game\_Date mmddyy10.;

if team = 'Boston College' then opp=**5**;

if team = 'Boston College' then osu=**2**;

**RUN**;

**data** record;

set hockey;

retain W L T;

if osu>opp then w+**1**;

else if osu<opp then L+**1**;

else if osu=opp then T+**1**;

array change \_numeric\_;

do over change;

if change=**.** then change=**0**;

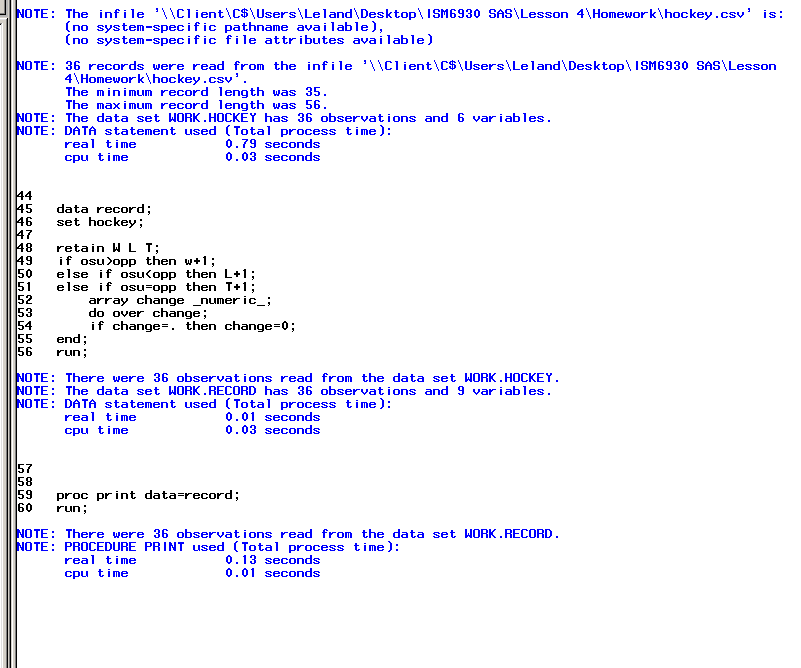
end;

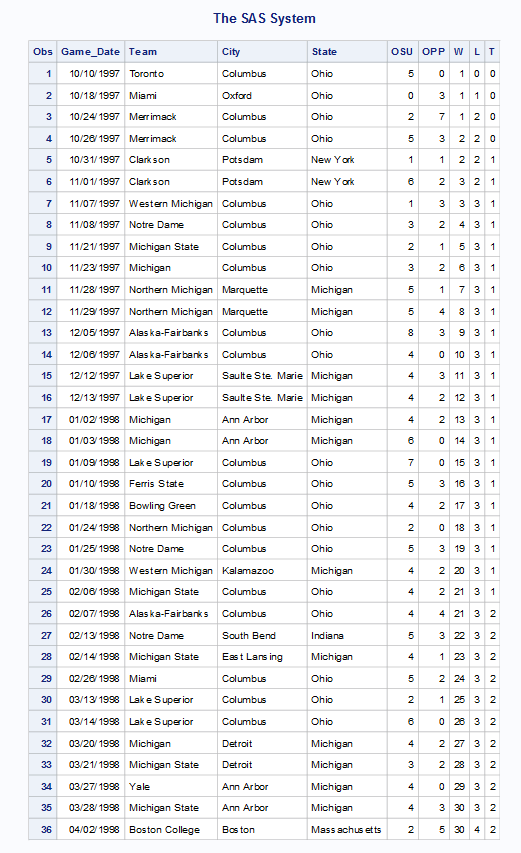
**run**;

**proc** **print** data=record;

**run**;

**LOG**

 **OUTPUT**



1. Suppose that your 5th grader is learning how to write Roman numerals, and you want to help her or him by preparing a study guide. Write a SAS program which uses DO loop to print the Arabic numbers 1, 2, 3, …,49, 50 AND their Roman equivalents. The ROMAN7. format in SAS will be helpful.

**CODE**

**data** numerals;

format Number comma7. Numerals roman7.;

do Number=**1** to **50**;

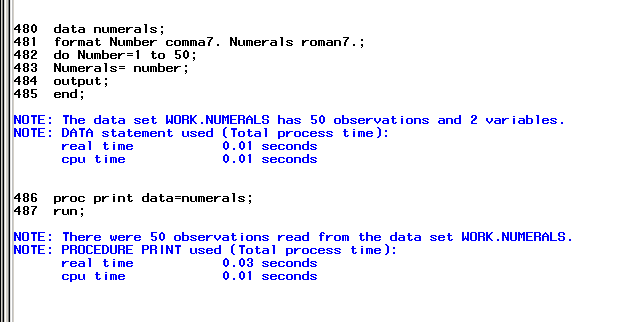
Numerals= number;

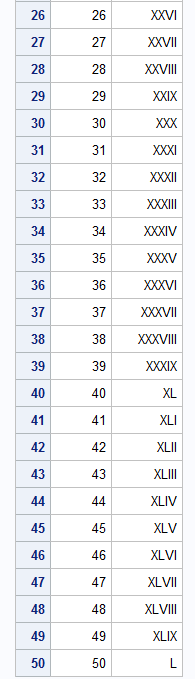
output;

end;

**proc** **print** data=numerals;

**run**; **LOG**

 **OUTPUT**

1. Refer to the DOGS3 data. Write a SAS program which creates a dataset using the INFILE statement. Then, create a new dataset which contains three variables: the name of the dog, the week of the measurement, and the eosinophil count in that week. There should be 75 observations in the new dataset. Print both datasets.

**CODE**

**data** dogs3;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 6\dogs3.txt' firstobs=**3** ;

input DogName :$8. Week0 Week2 Week4 ;

**run**;

**data** dogs4;

set dogs3;

Week=**0**; Eos=Week0; output;

Week=**2**; Eos=Week2; output;

Week=**4**; Eos=Week4; output;

keep DogName Week Eos;

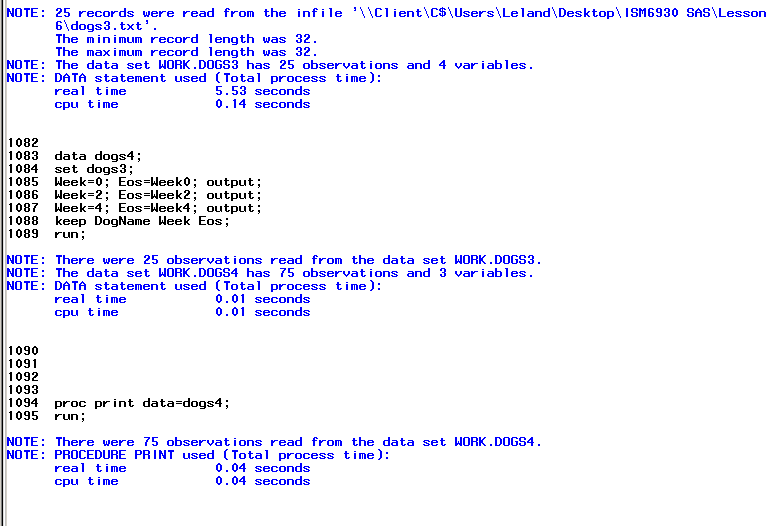
**run**;

**proc** **print** data=dogs3;

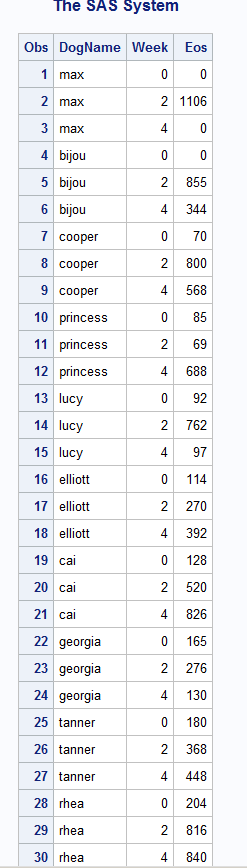
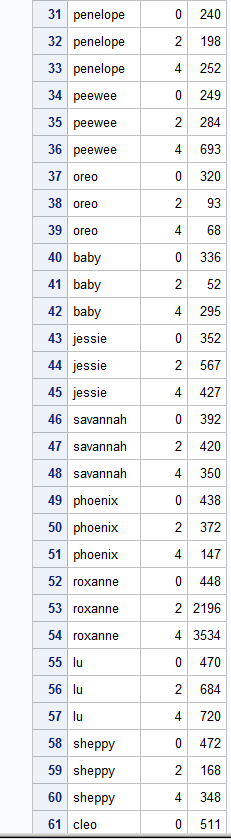
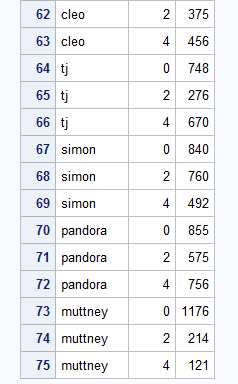
**run**;

**proc** **print** data=dogs4;

**run**; **LOG**

 **OUTPUT**



1. Refer to the RYAN data. Many of the ratings are missing. Suppose that a statistician wants to perform a procedure which does not allow missing values. He decides that one acceptable way to do the analysis is to replace each missing rating with the neutral rating (5). Write a SAS program which uses an array to replace all of the missing values for movie ratings with scores of five. Print the corrected dataset.

**CODE**

**data** ryan;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 4\Homework\ryan.txt' dlm='09'x firstobs=**2** missover;

input Title :$25. Roger NYTIMES USAT;

**run**;

\*6.2 slide 20 for example;

**data** missing;

set ryan;

array myvars{**3**} Roger NYTIMES USAT;

do i = **1** to **3**;

if myvars{i} = **.** then myvars{i} = **5**;

end;

drop i;

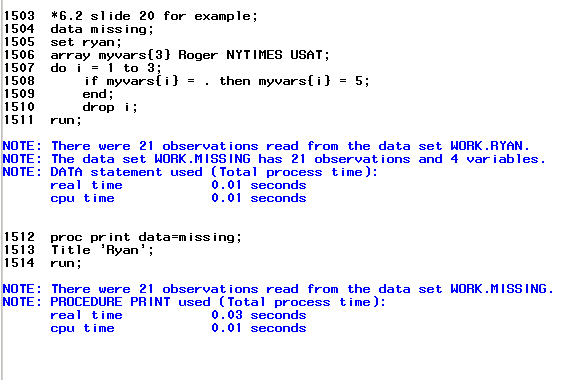
**run**;

**proc** **print** data=missing;

Title 'Ryan';

**run**;

**LOG**

 **OUTPUT**



1. Suppose that you work with many data files which contain observations collected over time, and you would find it convenient to have a program which would print the most recent observations in those datasets. Examples include stock market prices and batting averages in baseball. Write a SAS macro in which the user supplies the name of a SAS dataset which contains a date variable called DATE. The macro should then print out only the 10 most recently-occurring values. In other words, print the 10 observations that have the largest values of DATE. In the printout, use an appropriate format for the date. Demonstrate that your macro works by creating SAS datasets for the HOCKEY and CLINTON data, with DATE representing the date variable in each dataset, then apply the macro to those two datasets.

**CODE- %macro** ***TenDate***;

proc sort data=clinton;

by Descending Date;

run;

proc print data=clinton (obs = **10**);

format Date MMDDYY8.;

Title '10 Most Recent Dates';

run;

**%mend** TenDates;

**data** clinton;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 5\Homework\Clinton.txt' firstobs=**3**;

input @**7** Day **2.0**

Mon $ **10**-**12**

Year **14**-**17**

Approve **24**-**25**

Disapporve **32**-**33**

No\_opinion **40**-**41**;

Date\_cat = catt (day,Mon,Year);

Date= Input(Substr(strip(date\_cat),**1**,**9**),Date9.);

Put Date=Date9.;

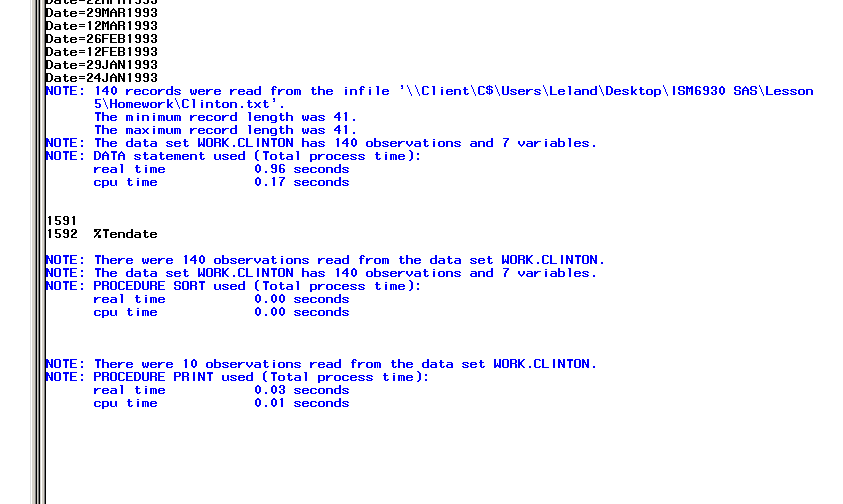
Format Date Date9.;

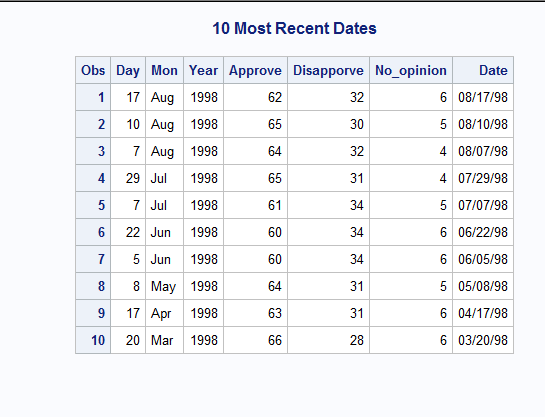
Drop Date\_cat;

**run**;

%***Tendate***

**LOG**

 **OUTPUT**



**CODE**

**%macro** ***TenDate***;

proc sort data=hockey;

by Descending Date;

run;

proc print data=hockey (obs = **10**);

format Date MMDDYY8.;

Title '10 Most Recent Dates';

run;

**%mend** TenDates;

**data** hockey;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 4\Homework\hockey.csv' dlm=',' dsd missover firstobs=**2**;

INPUT Date : MMDDYY10.

Team : $27.

City : $17.

State : $13.

OSU

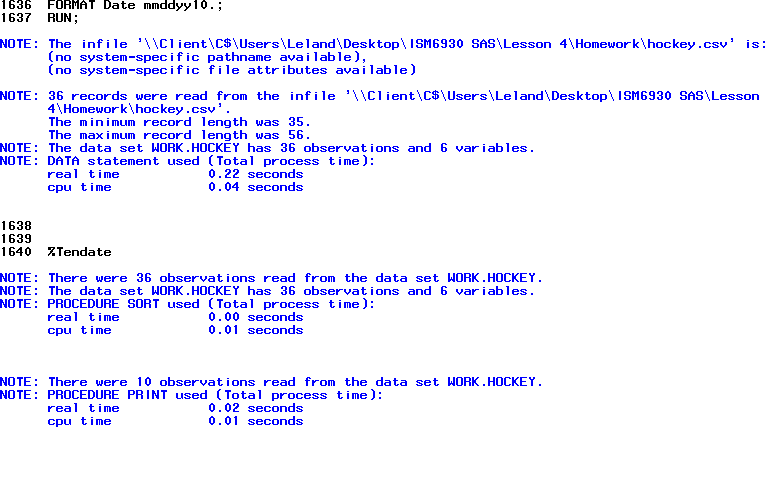
OPP;

FORMAT Date mmddyy10.;

**RUN**;

%***Tendate***

**LOG**

 **OUTPUT**



1. Refer to the GRADES data. Suppose that a total score of 60 is needed to earn a passing grade, and the instructor wants to know how many assignments each student turned in before achieving a total of at least 60 points. For example, a student who earned an 8 on each assignment would achieve at least 60 points in Week 8 (total=56 after 7 weeks, total=64 after 8 weeks). Print a listing of the students’ identification numbers and the numbers of weeks needed to achieve a passing score. The first few lines of output should look similar to this:

STUDENT WEEKS

1105 9

1294 9

2009 10

**CODE**

**data** grades;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Data\grades.txt' dlm='/';

input student **1**-**4** Week1 **6** Week2 **7** Week3 **8** Week4 **9** Week5 **10** Week6 **11** Week7 **12** Week8 **13** Week9 **14** Week10 **15** Week11 **16** Week12 **17** Week13 **18**;

array grades (**13**) Week1-Week13;

No\_Weeks= **0**;

sum= **0**;

do i=**1** to **13**;

if sum<**60** then do;

sum=grades(i)+sum;

no\_weeks=i;

end;

end;

drop i Week1-Week13 sum;

**run**;

**data** weeks;

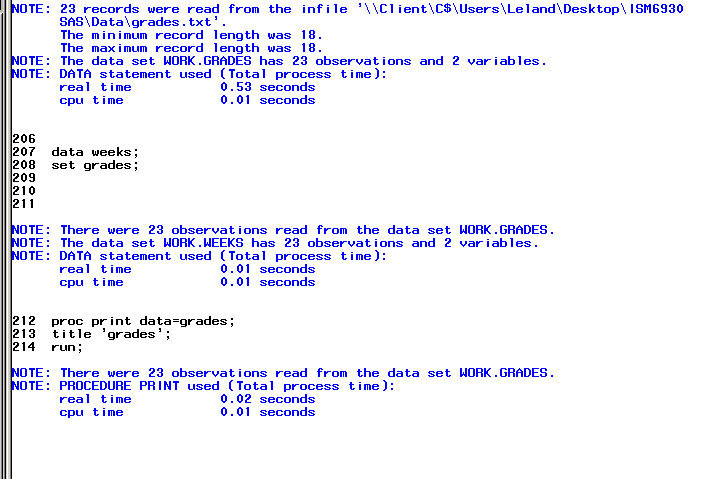
set grades;

**proc** **print** data=grades;

title 'grades';

**run**;

**LOG**

 **OUTPUT**



1. Refer to the CLINTON data. Write a SAS program which reads the data. Using only the polls taken in the year 1998, create a new variable which indicates whether the percentage of people approving of the President’s performance increased, decreased, or stayed the same from the time the last poll was taken. Also, create another variable which indicates the number of days elapsed from the previous poll to the current one. Print the dataset with the new variables.

Why would we want to do this? A statistician might suspect that the timing of the polls is dictated by current events and their effects on the President’s popularity. We might want to see if the polls are conducted more frequently in times of crisis (such as the Oklahoma City bombing or the release of the Lewinsky videotape) in which the President plays a major role. This problem shows one way to provide data to investigate this idea.

**CODE**

**data** clinton;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 5\Homework\Clinton.txt' firstobs=**3**;

input date date18. App Disapp No;

**run**;

**proc** **sort** data=clinton;

by date;

**run**;

**data** clinton98;

set clinton;

if date<**13880** then delete;

retain Approval Dat;

if (app-Approval)<**0** then change='Decrease';

else if (app-Approval)>**0** then change='Increase';

else if(app-Approval)=**0** then change='None';

else if app=**.** then change=' ';

elapsed=date-dat;

if elapsed=**.** then elapsed=**0**;

if approval=**.** then change=**.**;

Approval=app;

dat=date;

**run**;

**data** clinton98;

set clinton98;

drop Approval dat;

**run**;

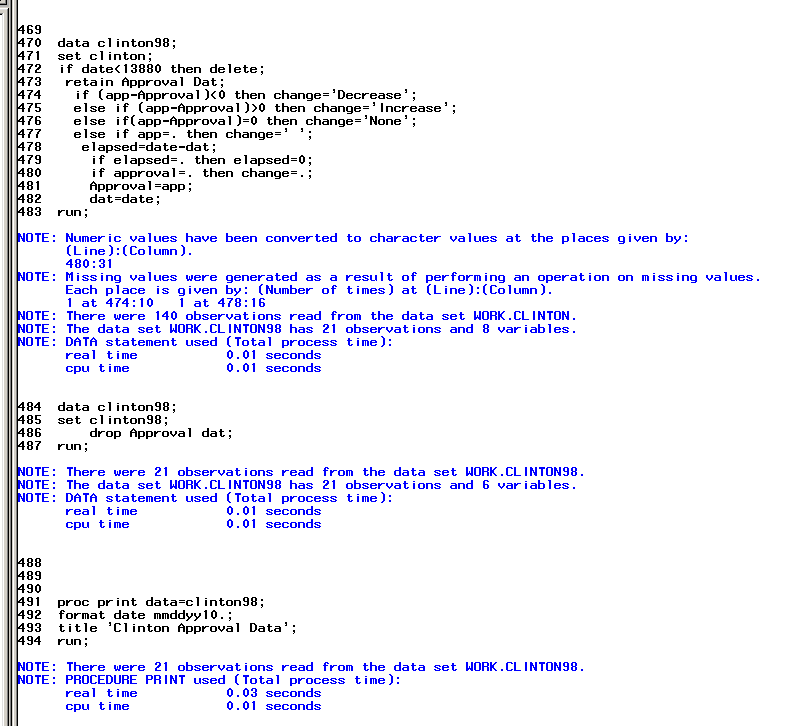
**proc** **print** data=clinton98;

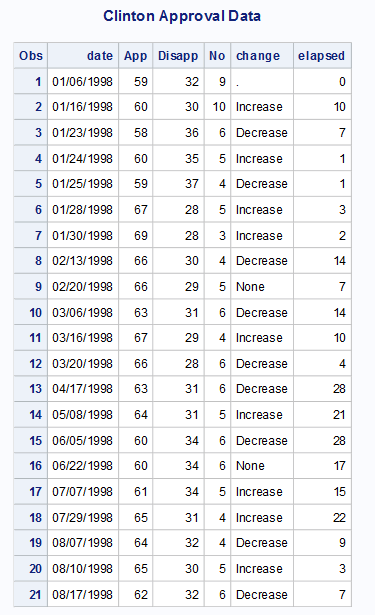
format date mmddyy10.;

title 'Clinton Approval Data';

**run**;

**LOG**

 **OUTPUT**



1. Refer to the AIRPORT data and create a SAS dataset from the CSV file. Then, create a new dataset with two observations and 21 variables. The two observations should correspond to the two years 1985 and 1995. One variable should indicate the year. The other 20 variables should be named CITY1 through CITY20 to represent the passenger totals at the 20 different airports represented in the data. Print both datasets.

**CODE**

**data** airport;

infile '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 3\Homework\airport.csv' dlm = ',' firstobs=**2** dsd ;

input city :$30. state $ abrev $ pass95 pass85;

**run**;

**proc** **transpose** data=airport

out=stuff(rename=(col1-col20=CITY1-CITY20 \_name\_=Year));

\*how do you rename \_name\_;

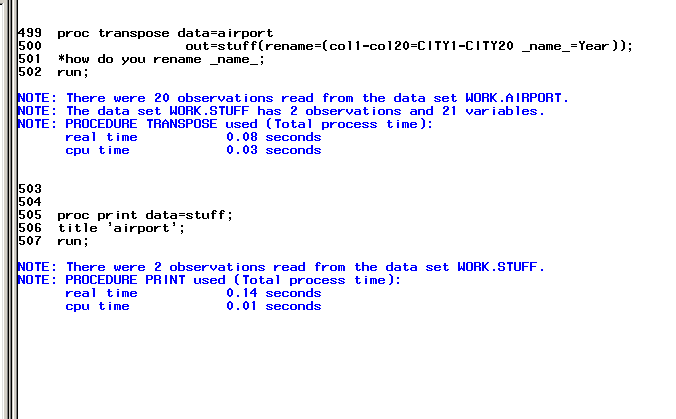
**run**;

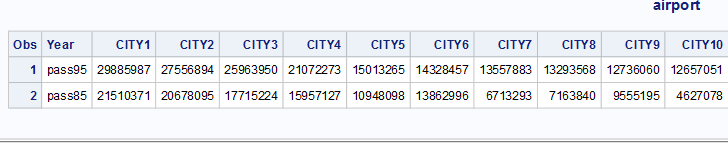
**proc** **print** data=stuff;

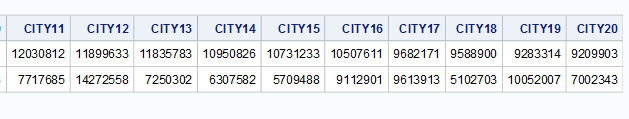
title 'airport';

**run**;

**LOG**

 **OUTPUT**





1. Merge the Purchase and Inventory data sets without sorting either one and omitting the BY Model statement. Check the SAS log and list the observations in Purchase, Inventory, and the merged data set. Now, run this same program with the system option MERGONBY set to WARN. Check the SAS log and display a snippet in your homework submission. Finally, set MERGEONBY to ERROR and run this program a third time. Check the SAS log.

**Note:** You might want to give the merged data set a different name for each of the three runs so that it is clear which programs run and which programs do not run.

You have an error in the merge with no BY statement- I didn’t understand this comment because I thought that is what the question is asking for.

**CODE**

libname learn '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Lesson 6';

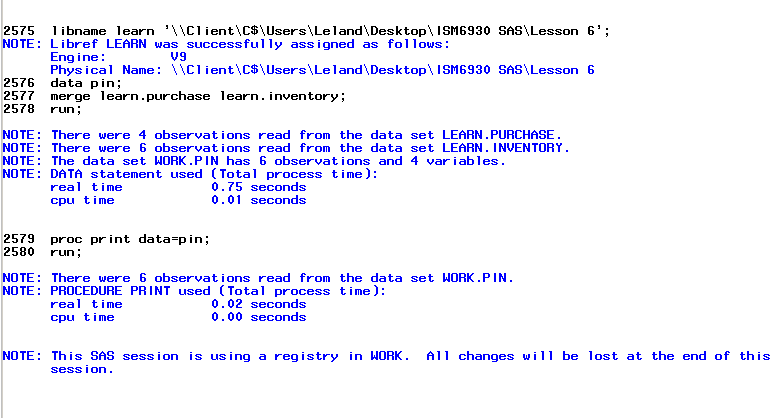
**data** pin;

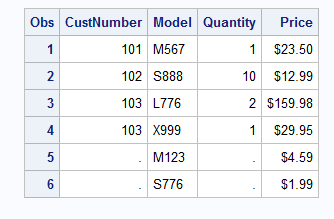
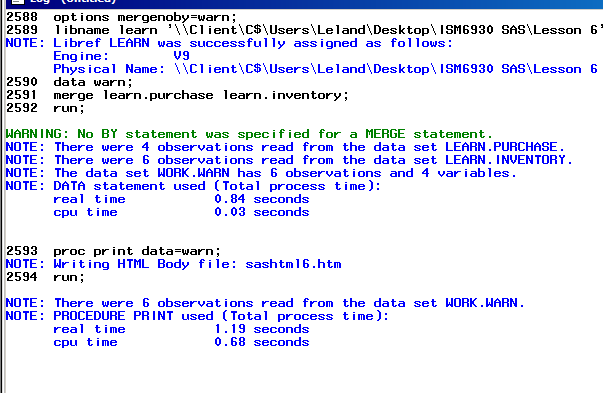
merge learn.purchase learn.inventory;

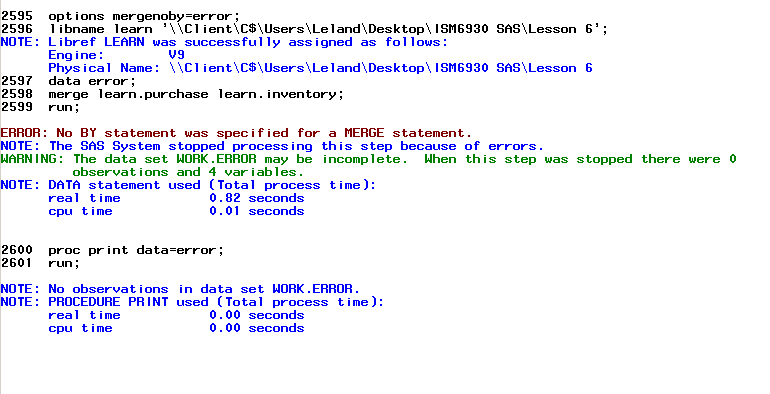
**run**;

**proc** **print** data=pin;

**run**; **LOG**

 **OUTPUT**

v

1. You want to merge two SAS data sets, Demographic and Survey2, based on an identifier. In Demographic, this identifier is called ID and it is character; in Survey2, the identifier is also called ID, but it is numeric.

**Hint:** If you choose to convert the numeric identifier to a character variable, use a Z3.format so that the leading 0s are present in the character variable.

ID = put(NumID,z3.);

**CODE**

libname learn '\\Client\C$\Users\Leland\Desktop\ISM6930 SAS\Data';

**proc** **sort** data=learn.demographic out=demographic;

by ID;

**run**;

**data** survey2;

set learn.survey2(rename=(ID = NumID));

ID= put(NumID,z3.);

drop numID;

**run**;

**proc** **sort** data=survey2;

by id;

**run**;

**data** combine;

merge demographic survey2;

by ID;

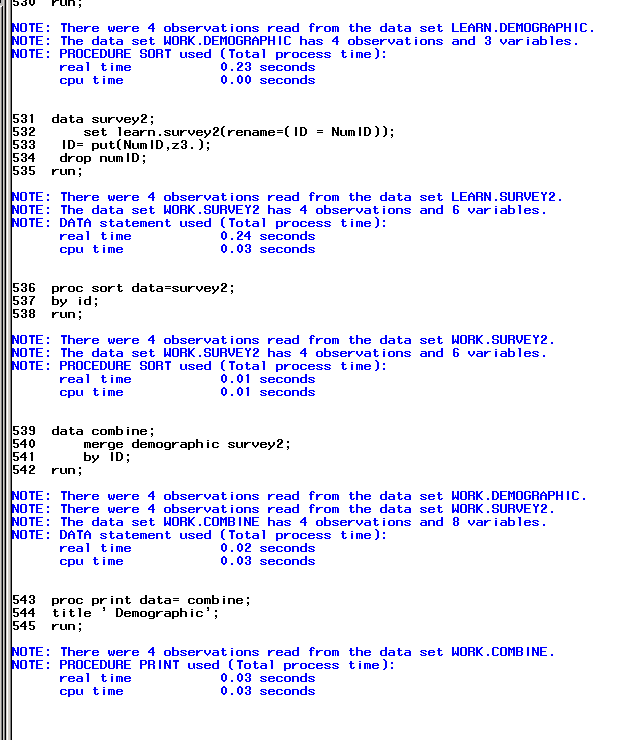
**run**;

**proc** **print** data= combine;

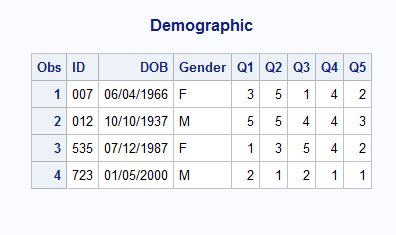
title ' Demographic';

**run**;

**LOG**



**OUTPUT**



1. Using SAS data set NINES, create a new temporary SAS data set (NONINES) where all values of 999 are replaced by SAS missing values. Do this without explicitly naming the numeric variables in data set Nines (use \_NUMERIC\_ when you define your array).

**CODE**

**data** nines;

set learn.nines;

**run**;

**data** Nonines;

set nines;

array nums{\*} \_numeric\_;

do i=**1** to dim(nums);

if nums{i}=**999** then call missing(nums{i});

end;

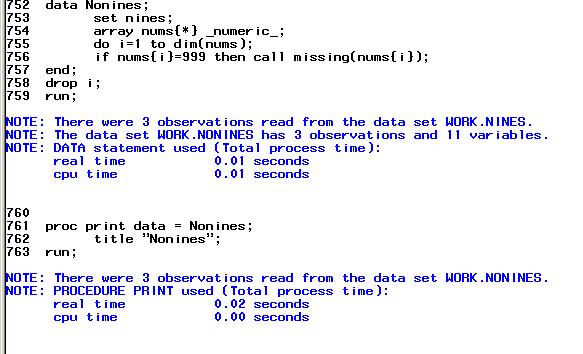
drop i;

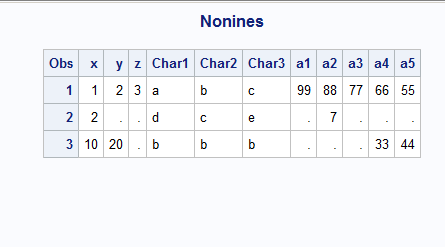
**run**;

**proc** **print** data = Nonines;

title "Nonines";

**run**; **LOG**

 **OUTPUT**



1. The passing score on each of five tests is 65, 70, 60, 62, and 68. Using the data here, use a temporary array to count the number of tests passed by each student.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test 1** | **Test 2** | **Test 3** | **Test 4** | **Test 5** |
| 001 | 90 | 88 | 92 | 95 | 90 |
| 002 | 64 | 64 | 77 | 72 | 71 |
| 003 | 68 | 69 | 80 | 75 | 70 |
|  |  |  |  |  |  |
| 004 | 88 | 77 | 66 | 77 | 67 |

CODE

**data** passing;

array pass\_score{**5**} \_temporary\_

(**65**,**70**,**60**,**62**,**68**);

array Score{**5**};

input ID : $3. Score1-Score5;

NumberPassed = **0**;

do Test = **1** to **5**;

NumberPassed + (Score{Test} ge pass\_score{Test});

\* for do equations you have to use letters for > < = ?;

end;

drop Test;

datalines;

001 90 88 92 95 90

002 64 64 77 72 71

003 68 69 80 75 70

004 88 77 66 77 67

;

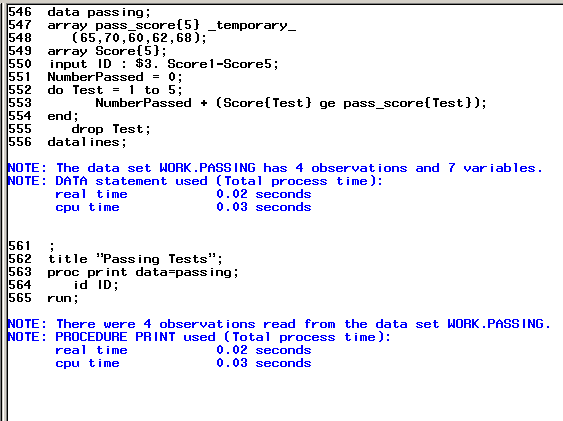
title "Passing Tests";

**proc** **print** data=passing;

id ID;

**run**;

LOG



OUTPUT

