ASSESMENT 3

**QUESTION 1**

**Data step programming**

The data step is used to meet most data manipulation requirements. It may be used for tasks ranging from importing data through to executing complex programs without a data set.

**Self assessment question**

Using *Step-by-step programming with Base SAS 9.4*, describe how the data step works, covering what happens in the main phases of operation.

**ANSWER 1**

**Defining the SAS Data Step;**

The SAS Data Step is a set of statements i.e. codes, that serve as the primary mechanism for manipulating and cleaning data in preparation for analysis.

In general, the syntax/for the SAS data step is;

DATA myfilename;

SAS code to input data;

SAS code to manipulate data;

SAS code to save data;

RUN;

**QUESTION 2**

##### CREDIT.DAT Formats

Below is the format used to convert the numeric values of the credit history variable in the CREDIT.DAT data set into text that the client will understand:

proc format;  
 value cr\_history  
 **0** = 'all paid'  
 **1** = 'bank paid'  
 **2** = 'paid before'  
 **3** = 'delay in paying'  
 **4** = 'not paid'  
 . = 'missing'  
 other = 'ERROR'  
 ;  
 run;

###### Self assessment question

1. What does the mapping: '*. = 'missing'*' do?
2. What does the mapping: '*other = 'ERROR'*' do?
3. Examine the observation for the customer with ID Number 7: can you explain why 'ERROR' appears in the output?

*Hint*: In the folder for this practical session assignment, you will find a SAS program, which imports the CREDIT.DAT data, as could have been produced following practical session assignment 2.

If you use this file for this practical session assignment, please alter the *filename* command for the alias *cwd* to use the directory where you placed your CREDIT.DAT file and run the program. If all goes well, you should see tables in the results window which help you answer this and the following questions.

*Suggestion*: You may find it instructive to examine the code in the file and to compare it with your own work.

**ANSWER 2**

1) Functions of the mapping ‘.’ = ‘missing’:

If raw data is incomplete because the value for a numeric variable is missing for an observation, the missing value is represented by a period (.) which serves as a placeholder.

**Note:** By adding a code for the missing value, an observation is added to the SAS data set and the SAS table retains its rectangular shape.

2) Functions of the mapping ‘other’=’ERROR’

If raw data variables is defined based on categorises, all the values that fall outside the cateogried defined for the observation of a variable are set to ‘ERROR’ in the SAS Data Set.

E.g: the defined categories for sex is ‘1’=‘M’

‘2’=’F’

If the value falls outside 1 or 2 in the raw data, then it’s an error.

3) Examining Customer ID 7:

The output for customer ID 7 is ‘ERROR’ because the mapping assigned to any value that’s not within the defined format is categorised as ‘other’ and the data set output is ‘ERROR’

**QUESTION 3**

1) List of Categorical variables;

* Cheques Account
* Credit History
* Purpose
* Savings accounts/bonds
* Present employment since
* Personal status and sex
* Other debtors/guarantors
* Personal status and sex
* Other debtors/guarantors
* Present residence since
* Property
* Age
* Other instalment plans
* Housing
* Job
* Telephone
* Foreign Worker

2) using

**proc** **freq** data = CUSTOMER;

/\* use table statement to choose particular variables\*/

table account

        history

        purpose

        amount

        savings

        employment

        married

        debtors

        resident

        property

        age

        plans

        housing

        job

        telephone

        foreign

        ;

**run**;

The proc freq shows that Credit History and Personal status and Sex contain ‘ERRORS’

3) options locale = English\_UnitedKingdom;

filename pwd 'C:\Users\Folashikemi\OneDrive - De Montfort University\P2586104 (vfiler1.lec-admin.dmu.ac.ukHome4)\Documents\IMAT5168 Analytical Programming Mark\Lab 2';

**proc** **format**;

value cr\_account

**1** = '< £0'

**2** = '£0 - £200'

**3** = '>= £200'

**4** = 'no account'

**.** = 'missing'

other = 'ERROR'

;

value cr\_history

**0** = 'all paid'

**1** = 'bank paid'

**2** = 'paid before'

**3** = 'delay in paying'

**4** = 'not paid'

**.** = 'missing'

other = 'ERROR'

;

value $cr\_purpose

'0' = 'car (new)'

'1' = 'car (used)'

'2' = 'furniture etc'

'3' = 'radio/tv'

'4' = 'appliances'

'5' = 'repairs'

'6' = 'education'

'7' = 'vacation'

'8' = 'retraining'

'9' = 'business'

'X' = 'others'

'' = 'missing'

other = 'ERROR'

;

value cr\_savings

**1** = '<£100'

**2** = '£100 - £500'

**3** = '£500 - £1000'

**4** = '>= £1000'

**5** = 'no account'

**.** = 'missing'

other = 'ERROR'

;

value cr\_employment

**1** = 'unemployed'

**2** = '< 1 year'

**3** = '1 - 4 years'

**4** = '4 - 7 years'

**5** = '>= 7 years'

**.** = 'missing'

other = 'ERROR'

;

value cr\_married

**1** = 'male :was married'

**2** = 'female :is or was married'

**3** = 'male :single'

**4** = 'male :is married'

**5** = 'female :single'

**.** = 'missing'

other = 'ERROR'

;

value cr\_debtors

**1** = 'none'

**2** = 'co-applicant'

**3** = 'guarantor'

**.** = 'missing'

other = 'ERROR'

;

value cr\_resident

**1** = '1 year'

**2** = '2 years'

**3** = '3 years'

**4** = '>=4 years'

**.** = 'missing'

other = 'ERROR'

;

value cr\_property

**1** = 'real estate'

**2** = 'if not 1: building society loan'

**3** = 'if not 1/2: car or other'

**4** = 'no property'

**.** = 'missing'

other = 'ERROR'

;

value cr\_plans

**1** = 'bank'

**2** = 'stores'

**3** = 'none'

**.** = 'missing'

other = 'ERROR'

;

value cr\_housing

**1** = 'rent'

**2** = 'own'

**3** = 'for free'

**.** = 'missing'

other = 'ERROR'

;

value cr\_job

**1** = 'unemployed'

**2** = 'unskilled'

**3** = 'skilled employee'

**4** = 'management'

**.** = 'missing'

other = 'ERROR'

;

value cr\_telephone

**1** = 'yes'

**2** = 'no'

**.** = 'missing'

other = 'ERROR'

;

value cr\_foreign

**1** = 'yes'

**.** = 'no'

other = 'ERROR'

;

**run**;

**data** CUSTOMER;

infile

pwd(credit.dat)

missover

;

input

account **5**

duration **7**-**8**

history **10**

purpose $ **12**-**13**

amount **14**-**18**

savings **20**

employment **22**

instalment **24**

married **26**

debtors **28**

resident **30**

property **32**

age **34**-**36**

plans **38**

housing **40**

credits **42**

job **44**

dependents **46**

telephone **48**

foreign **50**

;

label

customer = 'ID Number'

account = 'Chequing account [overdraft?]'

duration = 'Duration in months'

history = 'Credit history'

purpose = 'Purpose [of loan?]'

amount = 'Credit amount [requested?]'

savings = 'Savings accounts/bonds'

employment = 'Present employment since'

instalment = 'Instalment rate % income'

married = 'Personal status and sex'

debtors = 'Other debtors/guarantors'

resident = 'Present residence since'

property = 'Property [purchase method?]'

age = 'Age in years'

plans = 'Other instalment plans'

housing = 'Housing [ownership?]'

credits = 'Number of existing credits'

job = 'Job [type?]'

dependents = 'Number of dependents'

telephone = 'Telephone [line rental?]'

foreign = 'Foreign worker'

;

format

account cr\_account.

history cr\_history.

purpose $cr\_purpose.

amount nlmnlgbp8.0

savings cr\_savings.

employment cr\_employment.

married cr\_married.

debtors cr\_debtors.

resident cr\_resident.

property cr\_property.

plans cr\_plans.

housing cr\_housing.

job cr\_job.

telephone cr\_telephone.

foreign cr\_foreign.

;

**run**;

ods exclude enginehost;

**proc** **contents**

data=CUSTOMER

varnum

;

**run**;

ods select all;

**proc** **print** data=CUSTOMER;

format

account

history

purpose

amount

savings

employment

married

debtors

resident

property

plans

housing

job

telephone

foreign

;

**run**;

**proc** **print**

data = CUSTOMER

run;

**proc** **print**

data=CUSTOMER(obs=**10**)

label

noobs

;

**run**;

**proc** **freq** data = CUSTOMER;

/\* use table statement to choose particular variables\*/

table account

history

purpose

amount

savings

employment

married

debtors

resident

property

age

plans

housing

job

telephone

foreign

;

**run**;

**QUESTION 4**

##### Validation using: proc means

*proc means* can be used to determine the range of values for a continuous variable.

###### Self-assessment question

1. List the continuous variables in the CUSTOMER data set.
2. Use *proc means* to determine the number of observations (n), the number missing (nmiss), the minimum (min) and the maximum (max).
3. Are the ranges what you expect?
4. Please paste the new code in the box below.

*Hint*: you should look up *proc means* in the [SAS Procedures Guide](https://documentation.sas.com/?docsetId=proc&docsetTarget=p0f0fjpjeuco4gn1ri963f683mi4.htm&docsetVersion=9.4&locale=en).

**ANSWER 4**

1) List of continuous variables:

* ID Number
* Duration in months
* Credit Amount
* Age
* Instalment rate % income
* Number of existing credits
* Number of dependents

2) **proc** **means** data = CUSTOMER n nmiss min max;

/\*var customer

duration

amount

age

instalment

credits dependents

;\*/

**run**;

3)No, the ranges are not what I expect.

From the raw data the mapping for missing data for variable age is ‘999’ the data for maximum value for variable age is ‘999’ which shows that the output is incorrect.

A solution for this error will be to use the if statement to assign ‘999’ as missing values for age in the SAS data set.

4) **proc** **means** data = CUSTOMER n nmiss min max;

var customer

duration

amount

age

instalment

credits dependents

;

**run**;

**QUESTION 5**

Missing values are recorded as '.' (a period/full-stop) for numeric variables and ' '' ' (a pair of single quotes) for character variables.

The purpose of identifying missing values in a SAS data set is to permit the analyst to make a decision about their exclusion from an analytic procedure. Automatic exclusion may not be appropriate because there may be a pattern in the missing values that affects the relationship under investigation.

**Self assessment question**

1. Using the credit data set description, list the variables where missing values should be identified
2. Should there be a missing value for: 'savings account/bonds'? (Justify your answer.)
3. Can you identify one further advantage of converting raw data missing values to the standard SAS ones?

**ANSWER 5**

1) List of variables where missing values should be identified:

* Credit history
* Personal status and sex
* Other debtors/guarantors
* Present residence since
* Age in years
* Other instalment plans
* Housing
* Job
* Telephone

2) No, there isn’t a missing value for savings accounts/bonds, however for further analysis of missing data we may decide to separate the grouping for 5=unknown/no savings account and group unknown as missing values.

3) Further advantage of converting raw data missing values to standard SAS ones;

If a missing value is converted to standard SAS data set, we can query the data to see what percentage of thee data is missing and how the meaning data affects the entire data set.

We would be able to determine the significance of the missing values.

**QUESTION 6**

**ecisions**

One of the most common data step programming requirements when importing data, is changing the many ways missing values may be represented in raw data to the standard SAS missing values.

The '*if*' statement can be used to decide whether to mark a value as missing. The data step execution phase ensures that the statement is applied to each observation.

**Self assessment question**

Using the list of observations that should have missing values and other appropriate sources:

1. Write *if* statements to alter the raw data value to a SAS missing value. (See the *hint* below as to how this should be done.)
2. Check that the if statements have done their job by adding appropriate procedure statements to determine if your changes have functioned correctly.
3. Please paste the new code in the box below.

*Hint*: You do not need to make changes to the existing code. In the data step illustrated below, the *set* command replaces the *infile*/*input*  commands, we used previously, and reads each observation from the SAS data set called CUSTOMER. It then writes to the CUSTOMER data set thereby over-writing the original with updated data.

data CUSTOMER;  
 set CUSTOMER;  
 /\* your code goes here \*/  
 run;

You should replace '*/\* your code goes here \*/*' with your if-statements.

**ANSWER 6**

options locale = English\_UnitedKingdom;

filename pwd 'C:\Users\Folashikemi\OneDrive - De Montfort University\P2586104 (vfiler1.lec-admin.dmu.ac.ukHome4)\Documents\IMAT5168 Analytical Programming Mark\Lab 2';

**proc** **format**;

value cr\_account

**1** = '< Â£0'

**2** = 'Â£0 - Â£200'

**3** = '>= Â£200'

**4** = 'no account'

**.** = 'missing'

other = 'ERROR'

;

value cr\_history

**0** = 'all paid'

**1** = 'bank paid'

**2** = 'paid before'

**3** = 'delay in paying'

**4** = 'not paid'

**.** = 'missing'

other = 'ERROR'

;

value $cr\_purpose

'0' = 'car (new)'

'1' = 'car (used)'

'2' = 'furniture etc'

'3' = 'radio/tv'

'4' = 'appliances'

'5' = 'repairs'

'6' = 'education'

'7' = 'vacation'

'8' = 'retraining'

'9' = 'business'

'X' = 'others'

'' = 'missing'

other = 'ERROR'

;

value cr\_savings

**1** = '<Â£100'

**2** = 'Â£100 - Â£500'

**3** = 'Â£500 - Â£1000'

**4** = '>= Â£1000'

**5** = 'no account'

**.** = 'missing'

other = 'ERROR'

;

value cr\_employment

**1** = 'unemployed'

**2** = '< 1 year'

**3** = '1 - 4 years'

**4** = '4 - 7 years'

**5** = '>= 7 years'

**.** = 'missing'

other = 'ERROR'

;

value cr\_married

**1** = 'male :was married'

**2** = 'female :is or was married'

**3** = 'male :single'

**4** = 'male :is married'

**5** = 'female :single'

**.** = 'missing'

other = 'ERROR'

;

value cr\_debtors

**1** = 'none'

**2** = 'co-applicant'

**3** = 'guarantor'

**.** = 'missing'

other = 'ERROR'

;

value cr\_resident

**1** = '1 year'

**2** = '2 years'

**3** = '3 years'

**4** = '>=4 years'

**.** = 'missing'

other = 'ERROR'

;

value cr\_property

**1** = 'real estate'

**2** = 'if not 1: building society loan'

**3** = 'if not 1/2: car or other'

**4** = 'no property'

**.** = 'missing'

other = 'ERROR'

;

value cr\_plans

**1** = 'bank'

**2** = 'stores'

**3** = 'none'

**.** = 'missing'

other = 'ERROR'

;

value cr\_housing

**1** = 'rent'

**2** = 'own'

**3** = 'for free'

**.** = 'missing'

other = 'ERROR'

;

value cr\_job

**1** = 'unemployed'

**2** = 'unskilled'

**3** = 'skilled employee'

**4** = 'management'

**.** = 'missing'

other = 'ERROR'

;

value cr\_telephone

**1** = 'yes'

**2** = 'no'

**.** = 'missing'

other = 'ERROR'

;

value cr\_foreign

**1** = 'yes'

**.** = 'no'

other = 'ERROR'

;

**run**;

**data** CUSTOMER;

infile

pwd(credit.dat)

missover

;

input

customer **1**-**3**

account **5**

duration **7**-**8**

history **10**

purpose $ **12**-**13**

amount **14**-**18**

savings **20**

employment **22**

instalment **24**

married **26**

debtors **28**

resident **30**

property **32**

age **34**-**36**

plans **38**

housing **40**

credits **42**

job **44**

dependents **46**

telephone **48**

foreign **50**

;

label

customer = 'ID Number'

account = 'Chequing account [overdraft?]'

duration = 'Duration in months'

history = 'Credit history'

purpose = 'Purpose [of loan?]'

amount = 'Credit amount [requested?]'

savings = 'Savings accounts/bonds'

employment = 'Present employment since'

instalment = 'Instalment rate % income'

married = 'Personal status and sex'

debtors = 'Other debtors/guarantors'

resident = 'Present residence since'

property = 'Property [purchase method?]'

age = 'Age in years'

plans = 'Other instalment plans'

housing = 'Housing [ownership?]'

credits = 'Number of existing credits'

job = 'Job [type?]'

dependents = 'Number of dependents'

telephone = 'Telephone [line rental?]'

foreign = 'Foreign worker'

;

format

account cr\_account.

history cr\_history.

purpose $cr\_purpose.

amount nlmnlgbp8.0

savings cr\_savings.

employment cr\_employment.

married cr\_married.

debtors cr\_debtors.

resident cr\_resident.

property cr\_property.

plans cr\_plans.

housing cr\_housing.

job cr\_job.

telephone cr\_telephone.

foreign cr\_foreign.

;

**run**;

ods exclude enginehost;

**proc** **contents**

data=CUSTOMER

varnum

;

**run**;

ods select all;

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

format

account

history

purpose

amount

savings

employment

married

debtors

resident

property

plans

housing

job

telephone

foreign

;

**run**;

/\*

requirement= feedback about variable: value

note= observations limit: 10

note= label: present

note= observation number: removed

\*/

**proc** **freq** data = CUSTOMER;

/\* use table statement to choose particular variables\*/

table account

history

purpose

amount

savings

employment

married

debtors

resident

property

plans

housing

job

telephone

foreign

;

**run**;

**proc** **means** data = CUSTOMER n nmiss min max;

var customer

duration

amount

age

instalment

credits

dependents

;

**run**;

**data** CUSTOMER;

set CUSTOMER;

if (history = **9**) then history =**.**;

if (married = **9**) then married =**.**;

if (debtors = **9**) then debtors =**.**;

if (resident = **9**) then resident =**.**;

if (age = **999**) then age =**.**;

if (plans = **9**) then plans =**.**;

if (housing =**9**) then housing =**.**;

if (job = **9**) then job =**.**;

if (telephone = **9**) then telephone =**.**;

**run**;

**proc** **print**

data=CUSTOMER/\*(obs=10)\*/

label

noobs

;

**run**;

**QUESTION 7**

Without altering the existing data step, add a new data step that calculates;

* 1. RestRun as ratio of heartrate while resting to heart rate while running.
  2. Quick as 1 if the runner can run 1 mile in <= 7min and as 0 otherwise (you may to assume that the runner runs at a steady pace).

**ANSWER7**

**1)**

**data** FITNESS;

set FITNESS;

restrun = restpulse/runpulse;

**run**;

2)quick = (runtime \* **2**/**3**) < **7**;

**run**;

3) **proc** **means**

data = FITNESS range;

**run**;