libname sales '/home/u62109636/my\_shared\_file\_links/jhshows0/STA5066';

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise One\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **print** data=sales.au\_salesforce; /\*1\*/

**run**;

**data** work.codes; /\*2\*/

set sales.au\_salesforce;

FCode1 = substr(First\_Name,**1**,**1**);

FCode2 = substr(First\_Name,length(First\_Name),**1**);

LCode = substr(Last\_Name,**1**,**4**);

User\_ID = lowcase(cats(FCode1,FCode2,LCode));

**run**;

**proc** **print** data=work.codes; /\*3\*/

var First\_Name FCode1 FCode2 Last\_Name LCode User\_ID;

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Two\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **print** data=sales.newcompetitors; /\*1\*/

**run**;

**data** work.smallstores; /\*2\*/

set sales.newcompetitors;

fnum = left(substr(ID,**3**));

if substr(fnum,**1**,**1**) = '1';

City = propcase(City);

output;

**run**;

**proc** **print** data=work.smallstores; /\*3\*/

VAR ID City Postal\_Code ;

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Three\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **contents** data=sales.contacts; /\*1\*/

**run**;

**proc** **print** data=sales.contacts; /\*2\*/

**run**;

**data** states; /\*3\*/

set sales.contacts;

ss = substr(Address2,length(Address2)-**5**);

Location = zipnamel('ss');

**run**;

**proc** **print** data=states;

var ss Location;

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Four\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **contents** data=sales.customers\_ex5; /\*1\*/

**run**;

**proc** **print** data=sales.customers\_ex5 (obs=**15**); /\*2\*/

**run**;

**data** work.names(); /\*3\*/

keep NewName Name Gender; /\*3a\*/

set sales.customers\_ex5;

fN=scan(Name,-**1**);

lN=scan(Name,**1**,',');

Nn= cat(fN,lN); /\*3b\*/

if Gender = 'M' then /\*3c\*/

NewName = propcase(cat("Mr. ", Nn));

else

NewName = propcase("Ms. " || Nn); /\*3d\*/

**run**;

**proc** **print** data=work.names; /\*4\*/

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Five\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

/\*\*\*\*

proc print data=sales.customers\_ex5; /\*1\*/

**run**;

**data** Silver Gold Platinum; /\*2\*/

set sales.customers\_ex5;

\*if (-00-) in customer\_ID then

(-00-) = (-15-);

Customer\_ID =tranwrd(Customer\_ID, '-00-', '-15-'); /\*2a\*/

if Customer\_ID = ("Gold") then output Gold; /\*2b\*/

if Customer\_ID in ("Silver") then output Silver;

else if Customer\_ID in ('Platinum') then output Platinum;

keep Customer\_ID Name Country; /\*3\*/

**run**;

**proc** **print** data=Silver; /\*4\*/

title"Customers with Silver";

**run**;

**proc** **print** data=Gold;

title"Customers with Gold";

**run**;

**proc** **print** data=Platinum;

title"Customers with Platinum";

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Six\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **print** data=sales.employee\_donations; /\*1\*/

**run**;

**data** split;

set sales.employee\_donations;

PctLoc = find(Recipients,"%");

\*Charity = substr(Recipients, "");

**run**;

**proc** **print** data=split;

**run**;

(a) The data set work.split will have one observation for each combination of employee

and charity to which they donated.

(b) Some employees made two contributions; therefore, they have two observations in the

input data set. These employees will have a % character in the value of Recipients.

(c) Hint: Store the position where the % character was found in a variable PctLoc. This

can make subsequent coding easier.

(d) Create a variable Charity with the name and percent contribution of the appropriate

charity.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Seven\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **print** data=sales.orders\_midyear; /\*1\*/

**run**;

**data** work.sale\_stat; /\*2\*/

set sales.orders\_midyear;

MonthAvg = round(mean(of Month1-Month6)); /\*2a\*/

MonthMax = max(of Month1-Month6); /\*2b\*/

MonthSum = sum(Month1--Month6); /\*2c\*/

**run**;

**proc** **print** data=work.sale\_stat; /\*3\*/

var Customer\_ID MonthAvg MonthMax MonthSum;

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Eight\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **print** data=sales.orders\_midyear; /\*1\*/

**run**;

**data** work.freqcustomers; /\*2\*/

set sales.orders\_midyear;

if nmiss(Month1-Month6) eq **5**;

median = median(of Month1--Month6);

highest = max(of Month1-Month6);

**run**;

**proc** **print** data=work.freqcustomers; /\*3\*/

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Nine\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **contents** data=sales.shipped; /\*1\*/

**run**;

**proc** **print** data=sales.shipped; /\*2\*/

**run**;

**data** shipping\_notes; /\*3\*/

set sales.shipped;

length Comment $ **21.**;

cv = input(Ship\_Date,date9.);

Comment = cat("Shipped on" , cv);

Total = Quantity \* **5**;

**run**;

**proc** **print** data=shipping\_notes noobs;

format Total dollar7.2;

**run**;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Exercise Ten\*\*\*\*\*\*\*\*\*\*\*\*\*\*;

**proc** **contents** data=sales.us\_newhire; /\*1\*/

**run**;

**proc** **print** data=sales.us\_newhire; /\*2\*/

**run**;

**data** US\_converted; /\*3\*/

set sales.us\_newhire;

ID = compress(ID,'-');

ID\_NUM = input(ID, **12.**);

telephone\_char = substr(left(Telephone),**1**,**3**) || "-" || substr(left(Telephone),**4**,**4**);

Birth = input(Birthday, ddmmyy6.);

**run**;

**proc** **contents** data=us\_converted; /\*4\*/

**run**;

**proc** **print** data=us\_converted; /\*5\*/

**run**;

HEFLETCHER-REEVES(1964) method (with or without resetting) is known to have a

descent property when the step size is found by an exact line search, Powell (1984)

has shown that the global convergence for this method holds when an exact line

search is used, In this note we show that both the descent property and the global

convergence property of the Fletcher-Reeves method still hold for an inexact line

search when the step size satisfies certain standard conditions.

The Fletcher-Reeves method aims to solve the unconstrained optimization

problem

minimizefix), x E jR"

by a sequence of line searches

(I)

Xlk+11 =Xlkl+:x(kISlkl (2)

from a user supplied estimate xII). If the line search is exact the step size :xlklis

defined by

:xlk) =arg min f(Xlkl + :XSlkl),