```
* data - 999 used instead of . to indicate MISSING;

data old;
input a b c d e;
datalines;
10 20 30 999 50
50 40 999 20 10
999 1 1 1 999
;
run;

* write an IF statement to examine each variable - not so tedious if
* only 5 variables what about 10, 20, etc.

data new;
set old;
if a = 999 then a = .;
if b = 999 then b = .;
if c = 999 then c = .;
if d = 999 then d = .;
if e = 999 then e = .;
run;

* use an ARRAY - no way to learn about ARRAYS without also learning about DO LOOPS;

data new;
array x(5) a b c d e;
set old;
do i=1 to 5;
if x(i) eq 999 then x(i) = .;
end;
drop i;
run;

title 'NUMERIC 999 FIXED USING AN ARRAY';
proc print data=new noobs;
run;

* try a shortcut - refer to all numeric values with _numeric_
* will not work - location of array statement sometimes matters
* what variables are numeric - not known until after the SET statement;

data new;
array x(5) _numeric_;
set old;
do i=1 to 5;
if x(i) eq 999 then x(i) = .;
end;
drop i;
run;

* move the ARRAY statement AFTER the SET;

data new;
set old;
array x(5) _numeric_;
do i=1 to 5;
if x(i) eq 999 then x(i) = .;
end;
drop i;
run;

title 'NUMERIC 999 FIXED USING AN ARRAY WITH _NUMERIC_';
proc print data=new noobs;
run;
```

```
* same example, more numeric variables;

data old;
input a b c d e f g h i j k l m n o p q r s t;
datalines;
999 20  30 999  50  10  20 30  999 50  10 20 30   999  50  10  20 30  999 999
999 40   0  20  10  50  40 999 20  10  50 40 999  20   10  50  40 999  20  10
999  1   1  1  999 999   1 1    1 999  87  1   1  999 999 999   1   1  1  999
;
run;

* use an ASTERISK in the array statement
* let SAS count the number of numeric variables
* use the DIM function to set the number of iterations for the DO LOOP
* QUESTION - why not use I in the LOOP as in the previous example? ;

data new;
set old;
array x(*) _numeric_;
do ijk=1 to dim(x);
   if x(ijk) eq 999 then x(ijk) = .;
end;
drop ijk;
run;

title 'NUMERIC 999 FIXED USING AN ARRAY WITH _NUMERIC_';
proc print data=new noobs;
run;

* determine what variable was fixed - CALL VNAME;

data new (drop=var_name) fixed (keep=var_name);
length var_name $32;
set old;
array x(*) _numeric_;
do ijk=1 to dim(x);
   if x(ijk) eq 999 then do;
      x(ijk) = .;
        call vname (x(ijk),var_name);
        output fixed;
   end;
end;
output new;
drop ijk;
run;

title 'VARIABLES CHANGED FROM 999 TO MISSING';
proc freq data=fixed;
run;

* three diagnoses entered - n/a used to indicate MISSING (not/available);

data old;
length dx1-dx3 $5;
input dx1-dx3;
datalines;
630 64212 410
6340 78559 n/a
6442 n/a 2875
run;
```

```
* character array - arrays assumed to be numeric unless otherwise stated;

data new;
array diag(3) $5 dx1-dx3;
set old;
do i=1 to 3;
if diag(i) eq 'n/a' then diag(i) = ' ';
end;
drop i;
run;

title 'CHARACTER N/A FIXED USING AN ARRAY';
proc print data=new noobs;
run;

* use _character_;

data new;
set old;
array diag(*) _character_;
do i=1 to dim(diag);
if diag(i) eq 'n/a' then diag(i) = ' ';
end;
drop i;
run;

title 'CHARACTER N/A FIXED USING AN ARRAY AND _CHARACTER_';
proc print data=new noobs;
run;

* three diagnoses entered - n/a used to indicate MISSING (not/available)
* write your own informat;

proc format;
invalue $dx
'n/a' = ' '
;
run;

data new;
set old;
array diag(*) _character_;
do i=1 to dim(diag);
   diag(i) = input(diag(i),$dx5.);
end;
drop i;
run;

title 'CHARACTER N/A FIXED USING AN ARRAY AND _CHARACTER_ AND AND INFORMAT';

proc print data=new noobs;
run;

* screen data as you read it from raw data file;

data old;
informat dx1-dx3 $dx5.;
input dx1-dx3;
datalines;
630 64212 410
6340 78559 n/a
6442 n/a 2875
run;

title 'DATA SCREENED WITH AN INFORMAT';

proc print data=old;
run;
```

```
* multiple arrays;

data convert1;
input ht1-ht3 wt1-wt3;
htcm1 = 2.54 * ht1;
htcm2 = 2.54 * ht2;
htcm3 = 2.54 * ht3;
wtkg1 = wt1 / 2.2;
wtkg2 = wt2 / 2.2;
wtkg3 = wt3 / 2.2;
datalines;
60 66 72 100 150 200
;
run;

title 'DATA SET CONVERT1 - NO ARRAYS';
proc print data=convert1 noobs;
run;

data convert2 (keep=htcm1-htcm3 wtkg1-wtkg3);
array ht(3);
array htcm(3);
array wt(3);
array wtkg(3);
input ht1-ht3 wt1-wt3;
do k = 1 to 3;
htcm(k) = 2.54 * ht(k);
wtkg(k) = wt(k) / 2.2;
end;
datalines;
60 66 72 100 150 200
run;

title 'DATA SET CONVERT2 - ARRAYS';
proc print data=convert2 noobs;
run;

* use the same variable names;

data convert3 (drop=k);
array ht(3);
array wt(3);
input ht1-ht3 wt1-wt3;
do k = 1 to 3;
ht(k) = 2.54 * ht(k);
wt(k) = wt(k) / 2.2;
end;
datalines;
60 66 72 100 150 200
run;

title 'DATA SET CONVERT3 - ARRAYS - SAME VARIABLE NAMES';
proc print data=convert3 noobs;
run;
```

```
* temporary arrays, assigning values to array elements;

data passing;
array pass(5) _temporary_ (65 70 65 80 75);
array score(5);

input id $ score(*);

pass_num = 0;
do i = 1 to 5;
   if score(i) ge pass(i) then pass_num + 1;
end;

drop i;
datalines;
001 64 69 68 82 74
002 80 80 80 60 80
;
run;

title 'DATA SET PASSING';
proc print data=passing noobs;
var id pass_num score1-score5;
run;


* same methodology, use the CLINICAL data and set risk values
* set at-risk values for chol, sbp, dbp, hr
* use PICTURE formats to mark at-risk values;

libname x 'j:\sasclass\sasdata';

proc print data=x.clinical;
format _all_;

data risk;
array risk(4) _temporary_ (350 150 100 80);
array x(4) chol sbp dbp hrate;
set x.clinical;
nrisk = 0;
do j = 1 to 4;
   if x(j) gt risk(j) then  nrisk+1;
end;
format _all_;
run;

proc format;
picture chol
350 - high = '099*'
other      = '099 ';
picture sbp
150 - high = '099*'
other      = '099 ';
picture dbp
100 - high = '099*'
other      = '099 ';
picture hrate
80 - high  = '099*'
other      = '099 ';
run;

title 'DATA SET RISK - * INDICATES VALUE AT RISK';
proc print data=risk noobs;
var patient nrisk chol sbp dbp hrate;
format chol chol. sbp sbp. dbp dbp. hrate hrate.;
run;
```