Chapter 16 PROCESSING VARIABLES WITH ARRAYS

CREATING ONE-DIMENSIONAL ARRAYS

A SAS **array** is a temporary grouping of variables under a single name. An array exists only for the duration of the DATA step. Usage of the **ARRAY statement** reduces the number of statements that are required for processing variables. By grouping variables into a one-dimensional array, it is possible to process all these variables in a single DO loop. The syntax is

data *newdataset\_name*;

set *dataset\_name*;

array *array\_name*{*dimension*} *variable1 variable2* …;

do *index\_variable*=1 to *dimension*;

*variable\_name*{*index\_variable*}=*expression(array\_name{index\_variable})*;

end;

run;

* The dimension specifies the number of elements *variable1, variable2*, etc. in the array called array\_name.
* The dimension may be enclosed in either parentheses, braces, or brackets

array days(4) mon tue wed thr; or

array days{4} mon tue wed thr; or

array days[4] mon tue wed thr;

* Specifying the dimension of an array by an asterisk (\*) instructs SAS to determine the dimension by counting the number of elements.

array x{\*} total1 total5 total6 total18;

* To compute the number of elements in an array, use the **DIM** function. DIM(*array\_name*) may be used as the stop value for the *index\_variable* in a

DO loop.

array x{\*} total1 total5 total6 total18;

do i=1 to dim(x);

SAS statements;

end;

* The elements of the array are referenced as *variable1*=*array\_name*{1}, … *variable\_dimension*=*array\_name*{*dimension*}.
* \_NUMERIC\_ specifies a list of array elements as **all numeric variables** already defined in the current DATA step

array x{\*} \_numeric\_;

* \_CHARACTER\_ specifies a list of array elements as **all character variables** already defined in the current DATA step

array x{\*} \_character\_;

* \_ALL\_ specifies a list of array elements as **all variables of the same type (all numeric or all character)** already defined in the current DATA step

array x{\*} \_all\_;

Example. A data set tempF contains temperature in degrees Fahrenheit for four days

data tempF;

input day1-day4;

cards;

86 77 95 77

;

The following program converts the temperature into degrees Celsius.

data tempC (drop=i);

set tempF;

array xx{4} day1-day4;

do i=1 to 4;

xx{i}=5\*(xx{i}-32)/9;

end;

run;

The contents of the data set tempC is

day1 day2 day3 day4

30 25 35 25

Example. The ARRAY statement may be conveniently used to transform a row of observations into a column of observations. Suppose a data set quiz\_average contains average scores for quizzes 1 through 7

data quiz\_average;

input quiz1-quiz7;

cards;

78 84 72 61 87 82 83

;

The following code creates a new data set with a single variable quiz that contains the values of the average scores.

data new;

set quiz\_average;

array x{7} quiz1-quiz7;

do i=1 to 7;

quiz=x{i};

output;

end;

keep quiz;

run;

The contents of the data set new is

quiz

78

84

72

61

87

82

83

* To create an array of character variables, a dollar sign ($) should be added after the array dimension

array *array\_name*{*dimension*} $ *variable1 variable2* … ;

* Sometimes it is needed to assign initial values to array elements. The syntax is

array *array\_name*{*dimension*} *var\_1… var\_dimension* (*initial values*);

The initial values may be separated by blanks or commas. For example,

array invest{4} sum1 sum2 sum3 sum4(1000 1500 600 900);

array color{3} $ color1 – color3 ('red','blue','green');

* It is possible to assign initial values to an array without specifying the elements. The following statement creates variables xx1, xx2 and xx3, and assigns them initial values 1, 2, and 3.

array xx{3} (1 2 3);

* The option \_TEMPORARY\_ creates temporary array elements without creating new variables. The syntax is

array *array\_name*{*dimension*} \_temporary\_ (*initial values*);

Example. The following data contain student id, and actual scores for a quiz, homework, exam, and final.

data scores;

input student\_id$ quiz hw exam final;

cards;

4533 45 32 91 112

614 50 35 98 108

1185 30 23 76 92

;

The temporary array specifies the maximum possible score for the quiz, homework, exam, and final. Then the students’ scores as percentage of the maximum are computed.

data grade (drop=i);

set scores;

array scores{4} \_numeric\_;

array max{4} \_temporary\_ (50 35 100 120);

array percent{4};

do i=1 to 4;

percent{i}=scores{i}/max{i}\*100;

end;

run;

proc print noobs;

format percent1-percent4 6.2;

run;

The contents of data grade is

student\_

id quiz hw exam final percent1 percent2 percent3 percent4

4533 45 32 91 112 90.00 91.43 91.00 93.33

614 50 35 98 108 100.00 100.00 98.00 90.00

1185 30 23 76 92 60.00 65.71 76.00 76.67

TWO-DIMENSIONAL ARRAYS

Suppose variables x1 through x12 are written as a 3x4 array.

|  |  |  |  |
| --- | --- | --- | --- |
| x1 | x2 | x3 | x4 |
| x5 | x6 | x7 | x8 |
| x9 | x10 | x11 | x12 |

To define a **two-dimensional array**, the dimension of the array must be specified as the number of **rows** and the number of **columns** separated by **comma**. For example,

array new{3,4} x1-x12;

To perform calculations that involve elements of a two-dimensional array, use a **nested DO loop**.

Example. A small department store records gross monthly sales (in thousands of dollars) for each of three consecutive years.

data monthly;

input year$ month1-month12;

cards;

2006 54 64 50 47 45 42 48 53 56 60 62 67

2007 66 68 72 72 74 77 78 82 83 87 88 88

2008 92 95 98 104 102 104 108 107 108 110 112 115

;

The following program computes quarterly sales for the four quarters in each of the three years.

data quarterly (drop=i j);

set monthly;

array sales{4,3} month1-month12;

array qtr{4};

do i=1 to 4;

qtr{i}=0;

do j=1 to 3;

qtr{i}+sales{i,j};

end;

end;

run;

proc print noobs;

run;

The contents of the data set quarterly is

m m m

m m m m m m m m m o o o

o o o o o o o o o n n n

y n n n n n n n n n t t t q q q q

e t t t t t t t t t h h h t t t t

a h h h h h h h h h 1 1 1 r r r r

r 1 2 3 4 5 6 7 8 9 0 1 2 1 2 3 4

2006 54 64 50 47 45 42 48 53 56 60 62 67 168 134 157 189

2007 66 68 72 72 74 77 78 82 83 87 88 88 206 223 243 263

2008 92 95 98 104 102 104 108 107 108 110 112 115 285 310 323 337