SCstatisticalprogramming.com

Introduction

In DATA step programming, you often need to perform the same action on more than one variable. Although you can process variables individually, it is easier to handle them as a group. You can do this by using array processing.

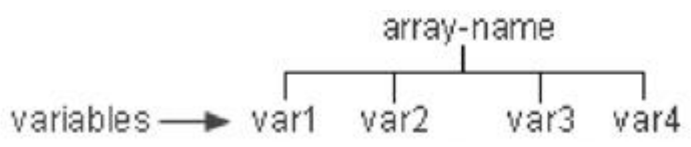
For example, using an array and DO loop, the program below eliminates the need for 365 separate programming statements to convert the daily temperature from Fahrenheit to Celsius for the year.

```
data work.report(drop=i);
   set master.temps;
   array daytemp{365} day1-day365;
   do i=1 to 365;
      daytemp{i}=5*(daytemp{i}-32)/9;
   end;
run;
```

SCstatisticalprogramming.com

Understanding SAS Arrays

A SAS array is a temporary grouping of SAS variables under a single name. An array exists only for the duration of the DATA step.



SCstatisticalprogramming.com

Defining an Array

To group data set variables into an array, we need to use an ARRAY statement.

General form, ARRAY statement:

where

ARRAY array-name{dimension} <elements>;

- array-name specifies the name of the array.
- dimension describes the number and arrangement of array elements.
- elements lists the variables to include in the array. Array elements must be either all numeric or all character. If no elements are listed, new variables will be created with default names.

Caution Do not give an array the same name as a variable in the same DATA step. Also, avoid using the name of a SAS function; the array will be correct, but you won't be able to use the function in the same DATA step, and a warning message will appear in the SAS log.

Caution You cannot use array names in LABEL, FORMAT, DROP, KEEP, or LENGTH statements. Arrays exist only for the duration of the DATA step.

SCstatisticalprogramming.com

Specifying the Array Name

To group the variables in the array, first give the array a name. In this example, make the array name sales.

```
array sales{4} qtr1 qtr2 qtr3 qtr4;
```

Specifying the Dimension

Following the array name, you must specify the dimension of the array. The dimension describes the number and arrangement of elements in the array. There are several ways to specify the dimension.

- In a one-dimensional array, you can simply specify the number of array elements. The array elements are the variables that you want to reference and process elsewhere in the DATA step.

```
array sales{4} qtr1 qtr2 qtr3 qtr4;
```

- The dimension of an array doesn't have to be the number of array elements. You can specify a range of values for the dimension when you define the array. For example, you can define the array sales as follows:

```
array sales{96:99} totals96 totals97 totals98 totals99;
```

- You can also indicate the dimension of a one-dimensional array by using an asterisk (*). This way, SAS determines the dimension of the array by counting the number of elements.

```
array sales{*} qtr1 qtr2 qtr3 qtr4;
```

- Enclose the dimension in either parentheses, braces, or brackets.

```
                ( )
array sales{4} qtr1 qtr2 qtr3 qtr4;
                [ ]
```

SCstatisticalprogramming.com

Specifying Array Elements

When specifying the elements of an array, you can list each variable name that you want to include in the array. When listing elements, separate each element with a space. As with all SAS statements, you end the ARRAY statement with a semicolon (;).

```
array sales{4} qtr1 qtr2 qtr3 qtr4;
```

You can also specify array elements as a variable list. Here is an example of an ARRAY statement that groups the variables Qtr1 through Qtr4 into a one-dimensional array, using a variable list.

```
array sales{4} qtr1-qtr4;
```

SCstatisticalprogramming.com

Variable Lists as Array Elements

You can specify variable lists in the forms shown below. Each type of variable list is explained in more detail in the table.

Variables	Form
a numbered range of variables	Var1-Varn
all numeric variables	_NUMERIC_
all character variables	_CHARACTER_
all character or all numeric variables	_ALL_

Examples:

```
array sales{4} qtr1-qtr4;

array sales{*} _numeric_;

array sales{*} _character_;

array sales{*} _all_;
```

Referencing Elements of an Array

Now let's look at some ways you can use arrays to process variables in the DATA step.

You use an array reference to perform an action on an array element during execution. To reference an array element in the DATA step, specify the name of the array, followed by a subscript value enclosed in parentheses.

General form, ARRAY reference:

array-name(subscript)

where subscript

- is enclosed in parentheses, braces, or brackets
- specifies a variable, a SAS expression, or an integer
- is within the lower and upper bounds of the dimension of the array.

SCstatisticalprogramming.com

Typically, arrays are used with DO loops to process multiple variables and to perform repetitive calculations. When used in a DO loop, the index variable of the iterative DO statement can reference each element of the array.

```
array quarter{4} jan apr jul oct;
do i=1 to 4;
   YearGoal=quarter{i}*1.2;
end;
```

For example, the DO loop above increments the index variable i from the lower bound of the quarter array, 1, to the upper bound, 4. The following sequence illustrates this process:

```
                   1
array quarter{4} jan apr jul oct;
do i=1 to 4;
   YearGoal=quarter{1}*1.2;
end;
                       2
array quarter{4} jan apr jul oct;
do i=1 to 4;
        YearGoal=quarter{2}*1.2;
end;
……
```

SCstatisticalprogramming.com

Using the DIM Function in an Iterative DO Statement

When using DO loops to process arrays, you can also use the DIM function to specify the TO clause of the iterative DO statement. For a one-dimensional array, specify the array name as the argument for the DIM function. The function returns the number of elements in the array.

General form, DIM function:

DIM (array-name)

where array-name specifies the array.

In this example, dim(wt) returns a value of 6.

```
array wt{*} weight1-weight6;
   do i=1 to dim(wt);
      wt{i}=wt{i}*2.2046;
   end;
run;
```

When you use the DIM function, you do not have to re-specify the stop value of an iterative DO statement if you change the dimension of the array.