

Arrays In SAS Software

Prepared by



International SAS® Training and Consulting

Destiny Corporation - 1321 Silas Deane Highway #A - Wethersfield, CT 06109-4302

Phone: (860) 721-1684 - 1-800-7TRAINING - Fax: (860) 721-9784

Email: destinycorp@compuserve.com Web: www.destinycorp.com

An array is a simple way of giving a single name to a group of related variables.

2 Types of Arrays:

- implicitly subscripted arrays
- explicitly subscripted arrays.

The former relate to older versions of the software; thus, only the latter will be discussed.

Anywhere you need to deal with variables across observations as a single unit or where you have to perform the same action on related variables in the same observation may be helped by the use of arrays.

Uses of Arrays

- create similar variables
- help read certain data structures
- repeat actions for variables
- perform table look-ups

General Form

`ARRAY arrayname {n} $ length array elements (initial-values);`

`arrayname` the name of the array, a valid SAS name

`n` the number of elements in the array, optional

`$` include if the array holds character variables, omit otherwise

`length` optional. Defines the common length for variables that have not been previously assigned lengths

Hands-on Workshops

arrayelements the variables in the array

initial values sets initial values for the corresponding elements in the array. The initial values must be separated by blanks or commas. If there are more elements than initial values, the remaining values are set to missing. Initial values are retained for each iteration of the data step. Place character initial values in quotes.

Rules:

- the array cannot contain a mixture of character and numeric variables
- an array cannot be referenced before being defined
- the array will create variables in the LPDV if they do not already exist
- the array definition does not cross data step boundaries. If a group of variables have been defined as an array in a data step, they will need to be redefined as an array if needed in that way in a subsequent data step.
- the array statement is non-executable
- arrays can be multi-dimensional {n1,n2,n3...} in any release and environment other than on IBM mainframes at Release 5.18 or below. In this instance you can create arrays of arrays.
- array names cannot be used in LABEL, FORMAT, DROP, or KEEP statements. To use these you must use the original variable names. Array names can be used in RETAIN statements.
- individual elements of the array can be referenced in the data step either by the variable name or by the array name with its index value. So, with...

```
array new {4} n1-n4;
```

...the variables can be referred to explicitly - n1, n2 etc... or can be referenced by new{1}, new{2}, new{3}, new{4}.

General Usage

Typical use of an array is with a DO loop in the following style:

```
ARRAY anyname {n} element1-elementn;  
DO indexvar = 1 TO n;  
    SAS statements with anyname{indexvar};  
END;
```

for example:

PROGRAM EDITOR

```
data assign;
  array x {5} x1-x5;
  do i=1 to 5;
    x{i}=100;
  end;
  drop i;
run;
proc print;
run;
```

OUTPUT

OBS	X1	X2	X3	X4	X5
1	100	100	100	100	100

Here the same processing has been applied to all of the array elements.

Note:

While this works, it is an inefficient way to assign values; use a RETAIN statement instead:

```
retain x1-x5 100;
```

Example 1: Converting Temperatures

In the first example, temperatures in Fahrenheit are converted to Celsius using the typical DO loop structure above. Each input observation has a set of six temperatures to be converted - only one is shown here. Note that the target variables are also defined in an array.

PROGRAM EDITOR

```
data convert(keep=c1-c6);
  input f1-f6;
  array f f1-f6;
  array c c1-c6;
  do i = 1 to 6;
    c{i}=(f{i}-32)*5/9;
  end;
  cards;
0 32 56 65 82 212
run;
proc print data=convert;
run;
```

Hands-on Workshops

Example2: Comparisons

In the second example we use two data sets:

The first, CORRECT, gives the correct answers to 6 test questions:

Data Set CORRECT

OBS	A1	A2	A3	A4	A5	A6
1	1	2	3	4	5	6

The second, PAPERS, gives the students' responses to these questions.

Data Set PAPERS

OBS	WHO	Q1	Q2	Q3	Q4	Q5	Q6
1	NICOLA	1	3	3	4	5	6
2	JAMES	1	2	4	5	5	6
3	SUE	1	2	9	6	5	6
4	JACK	1	2	2	4	5	6
5	LUKE	1	2	3	4	5	6

The Test:

More than 4 correct PASS!

4 or less correct FAIL!

The Program:**PROGRAM EDITOR**

```

data correct;
  input a1-a6;
  cards;
1 2 3 4 5 6
run;
data papers;
  input who $ q1-q6;
  cards;
Billy 1 3 3 4 5 6
Pat   1 2 4 5 5 6
Danny 1 2 9 6 5 6
Jack  1 2 2 4 5 6
Luke  1 2 3 4 5 6
run;
data pass(keep=who q1-q6) fail(keep=who q1-q6);
  if _n_ =1 then set correct;
  array correct {6} a1-a6;
  retain a1-a6;
  set papers;
  score = 0;
  array ans{6} q1-q6;
  do i = 1 to 6;
    if ans{i}=correct{i} then score+1;
  end;
  if score > 4 then output pass;
    else output fail;
run;
proc print data=pass;
run;
proc print data=fail;
run;

```

These are the reports:**OUTPUT**

OBS	WHO	Q1	Q2	Q3	Q4	Q5	Q6
1	Billy	1	3	3	4	5	6
2	Jack	1	2	2	4	5	6
3	Luke	1	2	3	4	5	6

OBS	WHO	Q1	Q2	Q3	Q4	Q5	Q6
1	Pat	1	2	4	5	5	6
2	Danny	1	2	9	6	5	6

Hands-on Workshops

Summary

Arrays in the data step of the SAS programming language are simply a short way of writing and referring to a series of variables.

Arrays are used throughout these course notes to solve many programming problems.