Multi-Dimensional Arrays In Pascal

In this section of notes you will learn about how and when to use multi-dimensional arrays.

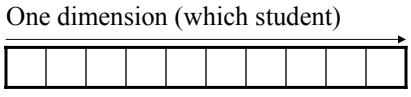
James Tam

When To Use Arrays Of Different Dimensions

- Determined by the data – the number of categories of information determines the number of dimensions to use.

Examples:

- (1D array)
  - Tracking grades for a class
  - Each cell contains the grade for a student i.e., grades[i]
  - There is one dimension that specifies the student



- (2D array)
  - Personal finances program
  - One dimension of information specifies the financial category (cash in or cash out).
  - The other dimension is used to specify the time

James Tam

When To Use Arrays Of Different Dimensions (2)

- (2D array continued)

Time

Financial category

	January	February	March	…
Income				
-Rent				
-Food				
-Fun				
-Car				
-Misc				
Net income				

When To Use Arrays Of Different Dimensions (3)

- (2D array continued)
- Notice that each row is merely a 1D array
- (A 2D array is an array containing rows of 1D arrays)

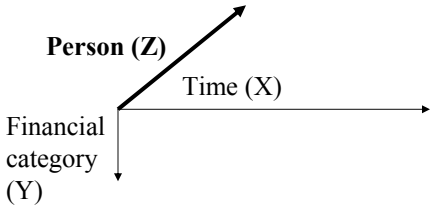
Columns

		[1]	[2]	[3]	[4]
[1]	Income				
[2]	-Rent				
[3]	-Food				
[4]	-Fun				
[5]	-Car				
[6]	-Misc				
[7]	Net income				

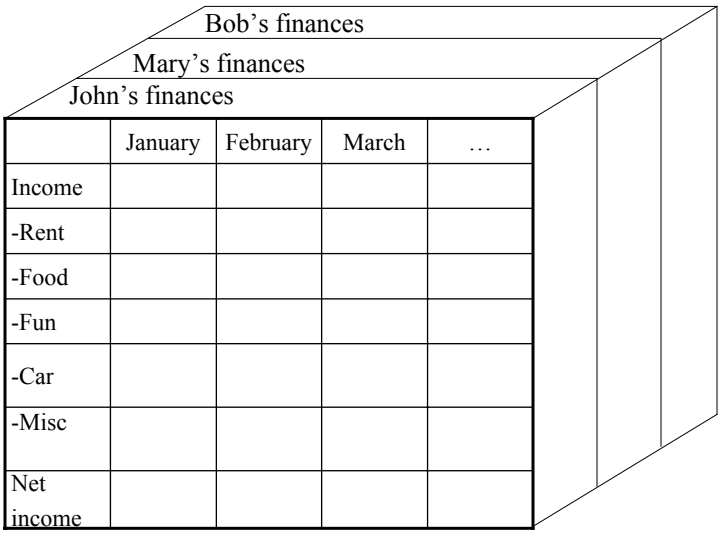
Rows

When To Use Arrays Of Different Dimensions (4)

- (3D array – take the 2D array but allow for multiple people)
- The third dimension specifies whose finances are being tracked.



When To Use Arrays Of Different Dimensions (5)



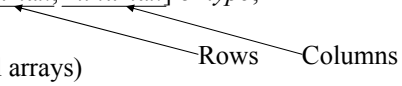
	January	February	March	…
Income				
-Rent				
-Food				
-Fun				
-Car				
-Misc				
Net income				

Declaring Multi-Dimensional Arrays

Format:

(Two dimensional arrays)

Name : array [*min..max*, *min..max*] of *type*;



(Three dimensional arrays)

Name : array [*min..max*, *min..max, min..max*] of *type*;

Example:

```
var johnFinances : array [1..7, 1..7] of real;
var cube         : array[1..3, 1..4, 1..6] of char;
```


Declaring Multi-Dimensional Arrays As A Type

Format:

Type declaration

Type name = array [*min..max*, *min..max*] of *element type*;

Type name = array [*min..max*, *min..max, min..max*] of *element type*;

Variable declaration

Array name : *Type name*;

Declaring Multi-Dimensional Arrays As A Type (2)

Example

Type declaration

```
Finances = array [1..7, 1..7] of real;
Cube = array[1..3, 1..4, 1..6] of char;
```

Variable declaration

```
var johnFinances : Finances;
var aCube        : Cube;
```


Accessing / Assigning Values To Elements

Format:

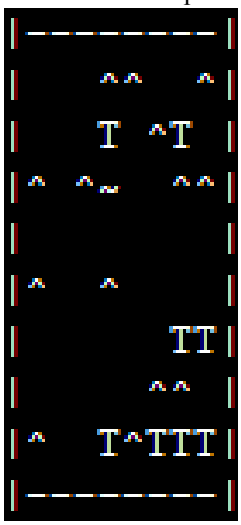
```
name [row][column] :=  name [row][column];
```

Example:

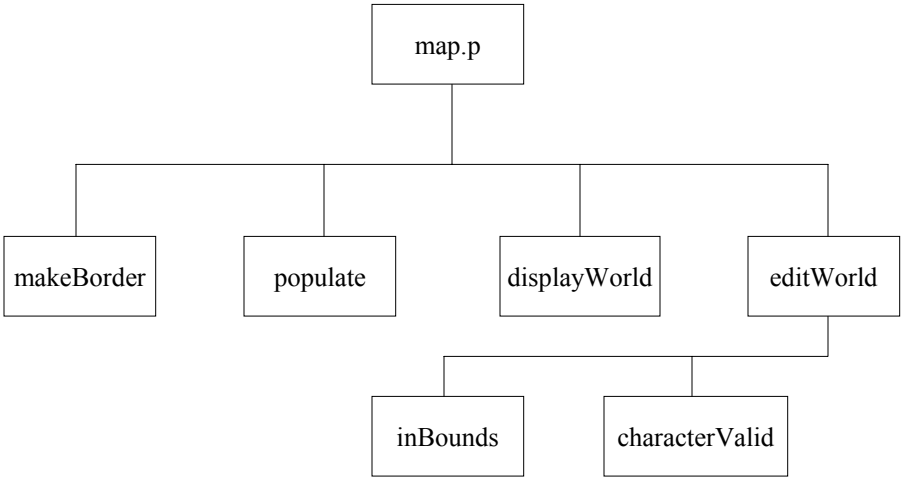
```
finances [1][1] := 4500;
writeln (finances[1][1]);
```

Example Program: Map Generator And Editor

You can find the full program in Unix under:
/home/231/examples/arrays/map.p



Example Program: Map Generator And Editor: Breaking The Problem Down



Example Program: Map Generator And Editor

```
program map (input, output);

const
  MAX_ROWS        = 10;
  MAX_COLUMNS =  10;

type
  Level =  array[1..MAX_ROWS, 1..MAX_COLUMNS] of char;
```


Example Program: Map Generator And Editor (2)

```
procedure makeBorder (var aLevel: Level);
var
  r : integer;
  c : integer;
begin
  for c := 1 to MAX_COLUMNS do
    aLevel[1][c] := '-';

  for c := 1 to MAX_COLUMNS do
    aLevel[MAX_ROWS][c] := '-';

  for r := 1 to MAX_ROWS do
    aLevel[r][1] := '|';

  for r := 1 to MAX_ROWS do
    aLevel[r][MAX_COLUMNS] := '|';

end; (* makeBorder *)
```

Example Program: Map Generator And Editor (3)

```
procedure populate (var aLevel :  Level);
var
  r               : integer;
  c               : integer;
  randomValue : real;
```


Example Program: Map Generator And Editor (4)

```
begin
  for r := 2 to (MAX_ROWS-1) do
  begin
    for c:= 2 to (MAX_COLUMNS-1) do
    begin
      randomValue := random;
      if (randomValue <= 0.05) then
        aLevel [r][c] := '~'
      else if (randomValue <= 0.25) then
        aLevel [r][c] := '^'
      else if (randomValue <= 0.40) then
        aLevel [r][c] := 'T'
      else
        aLevel [r][c] := ' ';
    end; (* inner for: traverse columns *)
  end; (* outer for: traverse rows *)
end; (* populate *)
```

Example Program: Map Generator And Editor (5)

```
procedure displayWorld (aLevel :  Level);
var
  r : integer;
  c :  integer;
begin
  for r := 1 to MAX_ROWS do
  begin
    for c := 1 to MAX_COLUMNS do
    begin
      write(aLevel[r][c]);
    end;
    writeln;
  end; (* for loop - displays world *)
end; (* displayWorld *)
```


Example Program: Map Generator And Editor (6)

```
function inBounds (row      : integer;
                   column : integer):boolean;
begin
  if (row < 2) OR
    (row > (MAX_ROWS-1)) OR
    (column < 2) OR
    (column > MAX_COLUMNS-1) then
     inBounds := false
  else
     inBounds := true;
end; (* inBounds *)
```

Example Program: Map Generator And Editor (7)

```
function characterValid (newCharacter :  char) : boolean;
begin
  if (newCharacter = '~') OR
    (newCharacter = '^') OR
    (newCharacter = 'T') OR
    (newCharacter = ' ') then
    characterValid := true
  else
    characterValid := false;
end; (* characterValid *)
```


Example Program: Map Generator And Editor (8)

```
procedure editWorld (var world : Level);
var
  editChoice      : char;
  charToChange  : char;
  rowToEdit       : integer;
  columnToEdit  : integer;
begin
  writeln;
  write('Enter "Y" or "y" if you wish to edit the world or the return ');
  write('key otherwise: ');
  readln(editChoice);
```

Example Program: Map Generator And Editor (9)

```
if (editChoice = 'Y') OR (editChoice = 'y') then
  begin
    writeln;
    write('Enter row (2 - 9) to edit: ');
    readln(rowToEdit);
    write('Enter column (2 - 9) to edit: ');
    readln(columnToEdit);
    if (inBounds(rowToEdit,columnToEdit) = false) then
    begin
      writeln('Value for row and column must be in the range of 2 - 9');
    end
```


Example Program: Map Generator And Editor (10)

```
    else
    begin
      writeln('What do wish to change this square to? Choices include:');
      writeln('"~" for water');
      writeln('"^" for trees');
      writeln('"T" for a town');
      writeln('" " (A space) for an open field');
      write('Enter choice and hit return: ');
      readln(charToChange);
      if (characterValid(charToChange) = true) then
      begin
        writeln('Changed: row ', rowToEdit,
              ', column ', columnToEdit, ' to ', charToChange);
        world[rowToEdit][columnToEdit] := charToChange;
      end
      else
         writeln('You can only populate the world with water, a forest,'
              ' a town or an empty space.');
    end; (* else *)
  end; (* if edit mode chosen. *)
end; (* editWorld *)
```

Example Program: Map Generator And Editor (11)

```
begin
  var outside      : Level;
  var quitChoice   : char;

  makeBorder(outside);
  populate(outside);

  repeat
  begin
    displayWorld(outside);
    editWorld(outside);
    write('Type "Q" or "q" to quit, or return to continue: ');
    readln(quitChoice);
  end; (* repeat loop *)
  until (quitChoice = 'Q') OR (quitChoice = 'q');
end. (* End of main program *)
```


You Should Now Know

- The number of dimensions that should be set for an array
- How to declare arrays of multiple dimensions
- How to access and assign values to different parts (elements, rows etc.) of multi-dimensional arrays
- How to scan selected parts of the array using loops