Arrays

ESC108A Elements of Computer Science and Engineering
B. Tech. 2017

Course Leaders:

Roopa G.

Ami Rai E.

Chaitra S.

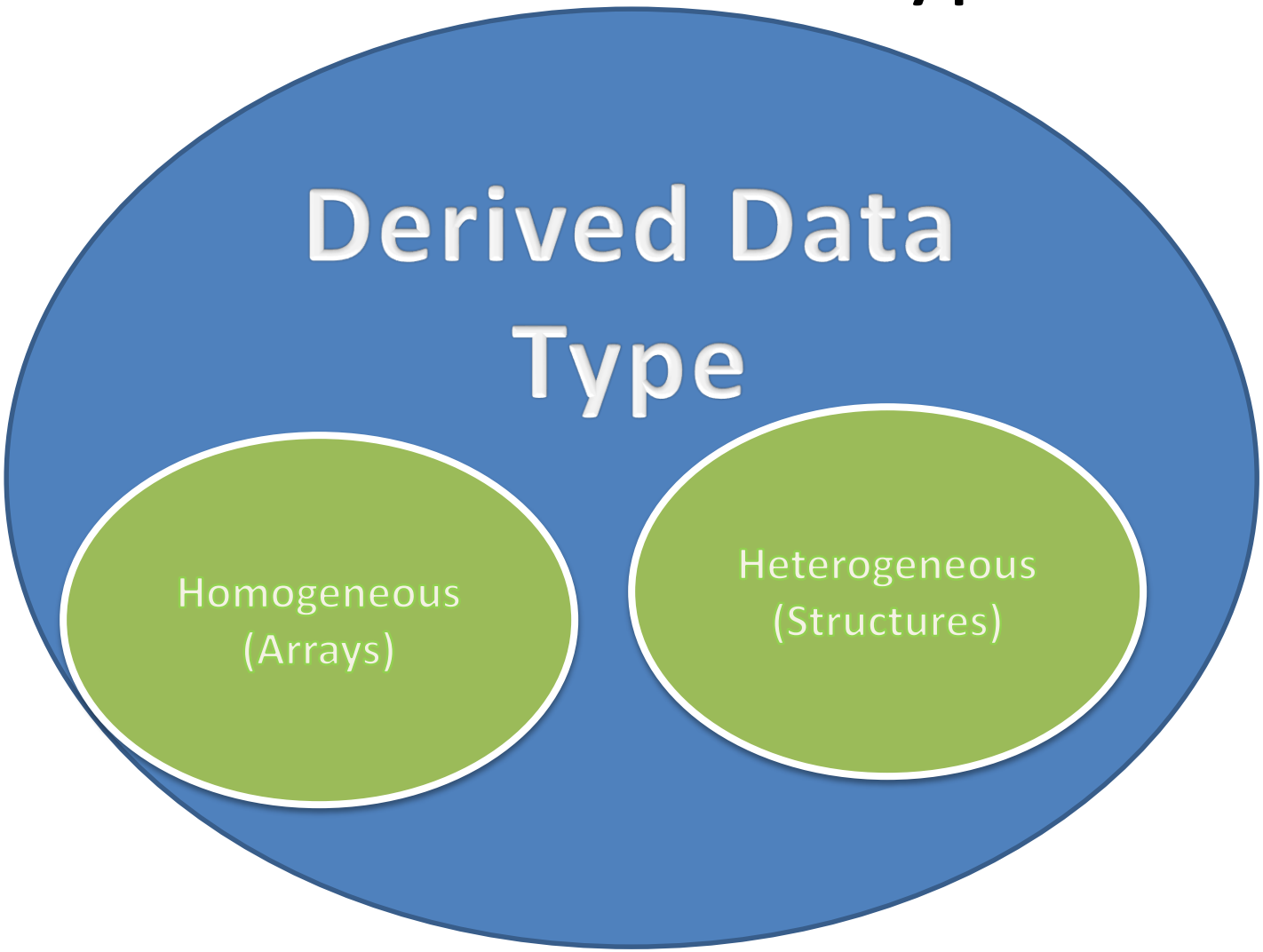
Objectives

- At the end of this lecture, student will be able to
  - Express logic using one-dimensional array variables
  - Express one-dimensional arrays and multi-dimensional arrays in C programming language

Contents

- Derived Data Structures – Arrays
- Two dimensional arrays
- Multidimensional arrays

Derived Data Type



Arrays

- Collection of elements of same data type
- All these elements are stored in consecutive memory locations
- Values can repeat – It is not a set

- The collection is represented by one name
- Each individual data in the array is referenced by a subscript or index (positive integer constant or expression) enclosed in a pair of square brackets []

Arrays

- Array contains 12 elements
- The first element in every array is the zeroth element
  - first element of array c is referred to as c[0]
  - second element of array c is referred to as c[1]
- In general, the ith element of array c is referred to as c[i – 1]

Name of array (note that all elements of this array have the same name, c)

c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78

Position number of the element within array c

Defining Arrays

- Arrays occupy space in memory
- Programmer specify the type of each element and the number of elements required by each array
- Syntax

  <data type> <identifier> [<*constant* size>];

- Example

```
int numArray[20]; /*tell the computer to reserve 20 spaces.
        Elements are from numArray[0] to numArray[19]*/
```

or

```
#define N 20
int numArray[N];
```

Array Initialisation

- Individual elements of the array can be initialised
  - Initial values must be constants, never be variables or function calls

- Example

```
int numArray[4]= {10,20,30,40};
        /*4 is size. numArray[0]=10,… numArray[3]=40*/
```

or

```
int numArray[4];
numArray[0]=10;
numArray[1]=20;
```

Array Initialisation contd.

- The array definition

  int n[5] = { 32, 27, 64, 18, 95, 14 };

  causes a syntax error because there are six initializers and only five array elements

- If the array size is omitted from a definition with an initializer list, the number of elements in the array will be the number of elements in the initializer list
- For example,

  int n[] = { 1, 2, 3, 4, 5 };

  would create a five-element array

Address of the Array Elements

- Array name is the same as the address of the array's first element

```
int array[5];  /* define an array of size 5 */
printf( " array = %p \n &array[0] = %p \n &array = %p\n",
array, &array[ 0 ], &array );
```

```
Output : array = 0012FF78
         &array[0] = 0012FF78
         &array = 0012FF78
```

- %p conversion specifier
  - a special conversion specifier for printing addresses
  - Normally outputs addresses as hexadecimal numbers

Algorithms

- Arrays

  <identifier>: array [<initial value> .. <final value>] *of* <Primitive data type>*;*

  <identifier>[<index value>*]*

- Examples

  numArray: array [0 .. n] *of* Integer;

  numArray[10] := 20*;*

Algorithm - Reading an Array

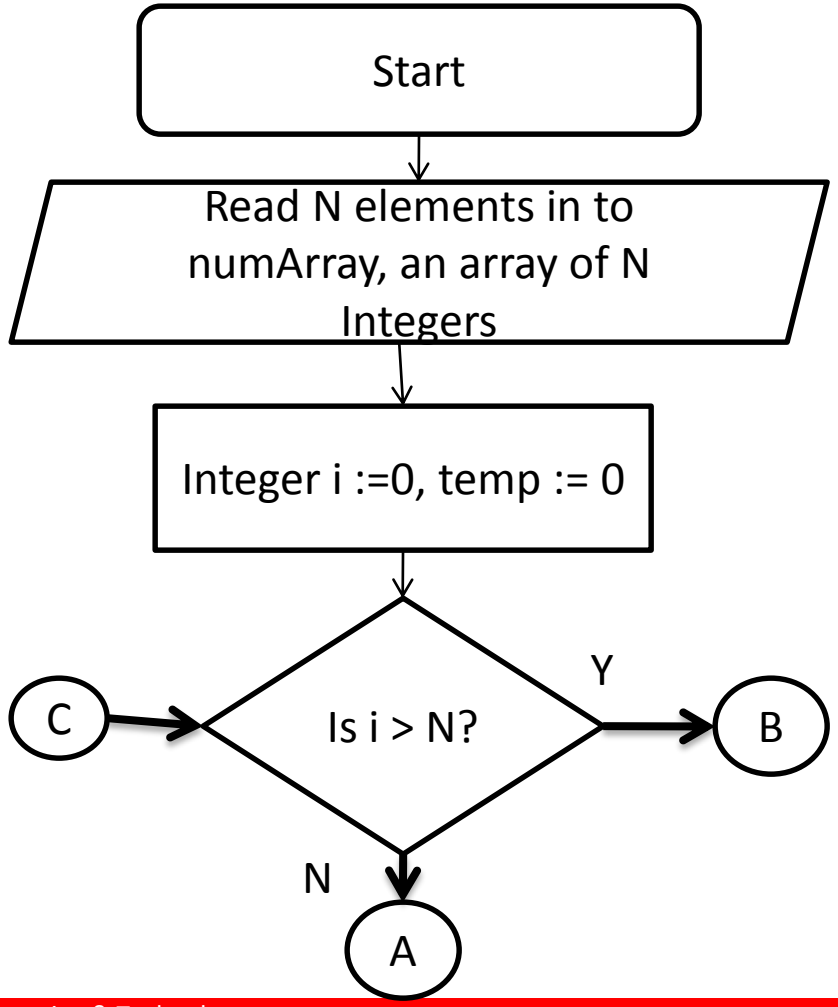
```
Algorithm sigmaN (numArray: Array [0 .. N] of Integer):Integer
var  i, temp: Integer; {temp is the return value}
begin
    for i in 0 to N, step 1 do
    begin
      writeln ('Please enter the number at index ', i, ':');
      readln (numArray[i]);
    end
end
```

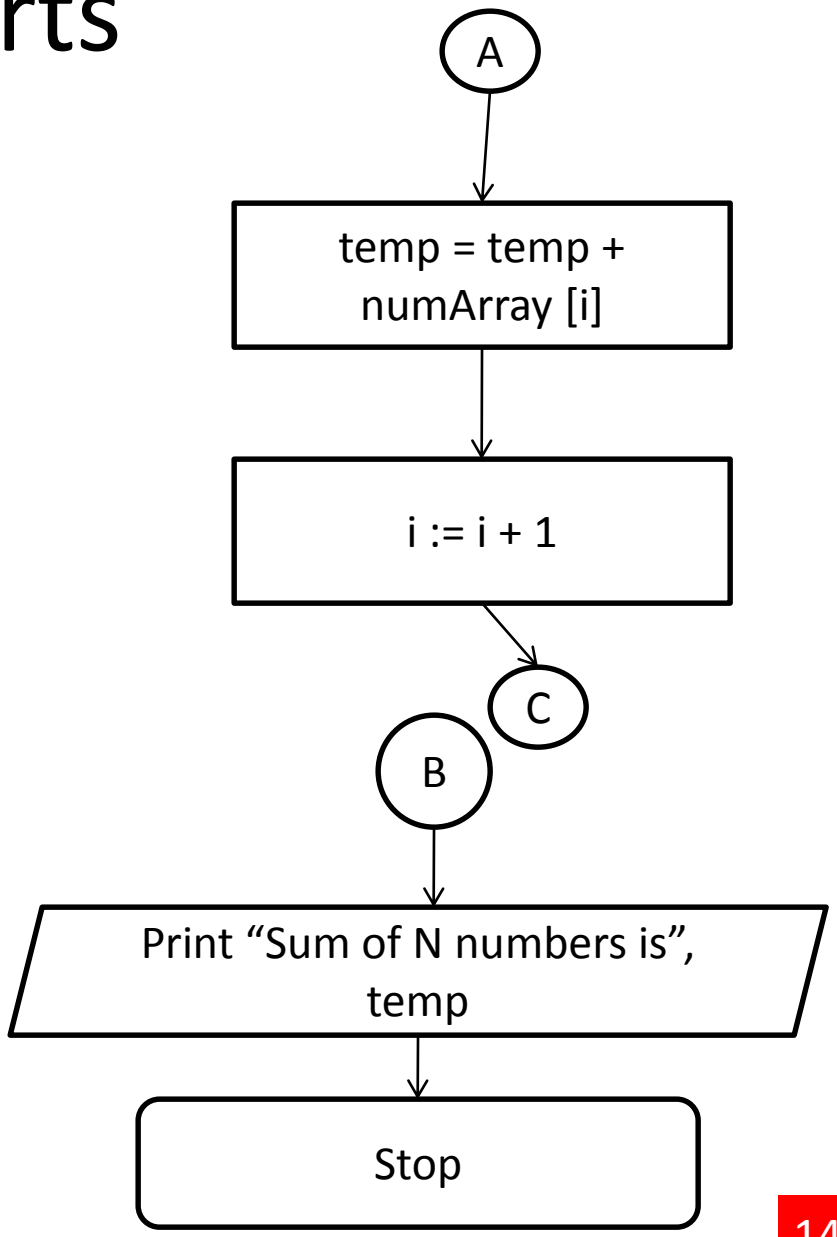
Algorithm – Sum of *N* numbers

```
Algorithm sigmaN (numArray: Array [0 .. N] of Integer):Integer
var  i, temp: Integer; {temp is the return value}
Begin
    temp := 0;
    for i in 0 to n, step 1 do
    begin
      temp := temp + numArray[i];
    end
stop
```

Flow Charts

- Summation of N numbers





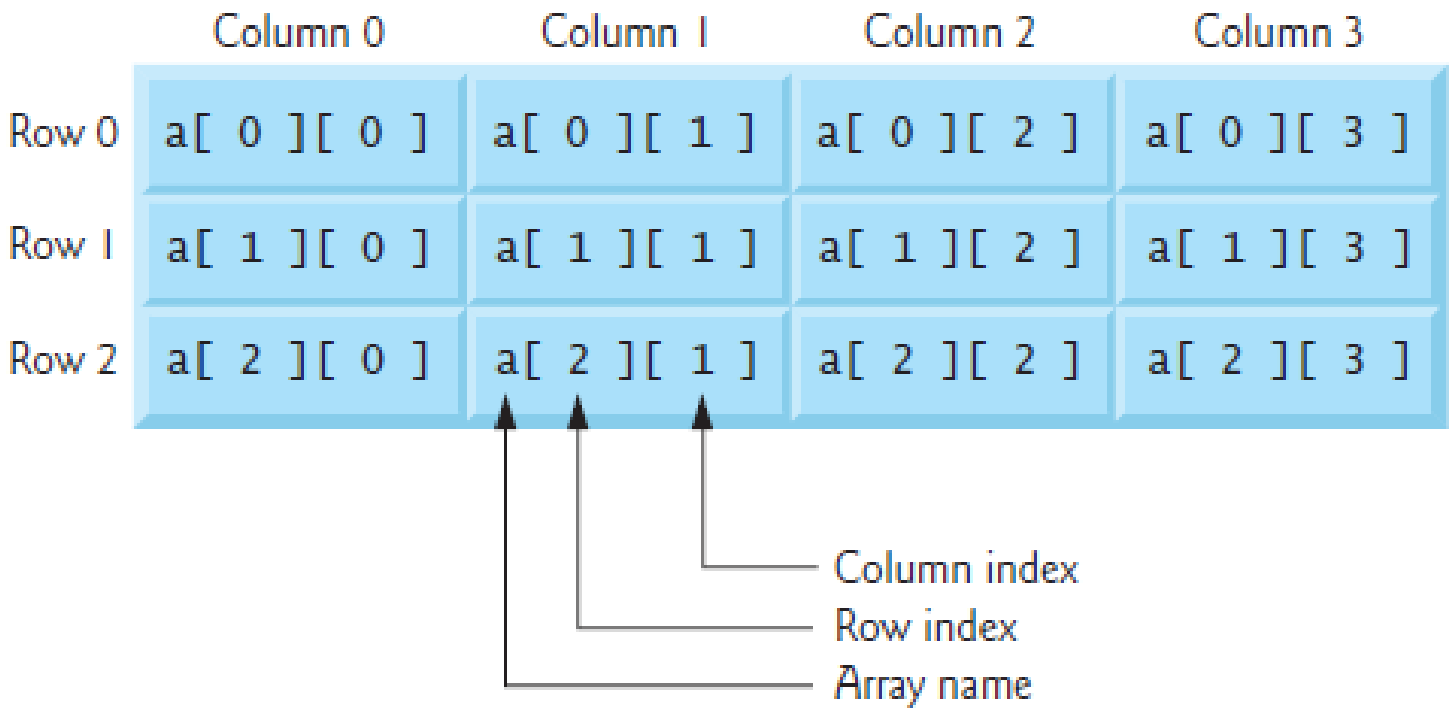
Two-Dimensional Arrays

- The elements of a two-dimensional array are arranged in rows and columns

- In general, an array with *m rows and n* columns is called an *m-by-n array*

Two-Dimensional Arrays

- The array contains three rows and four columns, so it's said to be a 3-by-4 array



Accessing Two-Dimensional Array Elements

- Array declaration

  <data type> <identifier> [<row size>] [<column size>];

  int matrix[2][3]; //array name is matrix with 2 rows and 3 columns

- An element in 2-dimensional array is accessed by using two subscripts, i.e., row index and column index of the array
- Example: int val = a[2][3];

  // take 4th element from the 3rd row of the array

Initializing Two-Dimensional Arrays

- Multidimensional arrays may be initialized by specifying bracketed values for each row

- An array with 3 rows and each row has 4 columns

```
int a[3][4] = { {0, 1, 2, 3} , {4, 5, 6, 7} , {8, 9, 10, 11}};
```

is equivalent to

```
int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};
```

Initializing Two-Dimensional Arrays contd.

- If there are not enough initializers for a given row, the remaining elements of that row are initialized to 0

```
int b[2][2] = { { 1 }, { 3, 4 } };
```

would initialize

b[0][0] to 1

b[0][1] to 0

b[1][0] to 3

b[1][1] to 4

Multi-dimensional Arrays

- General form of a multidimensional array declaration:

  type name[size1][size2]...[sizeN];

- For example, the following declaration creates a three dimensional 5 . 10 . 4 integer array:

  int threedim[5][10][4];

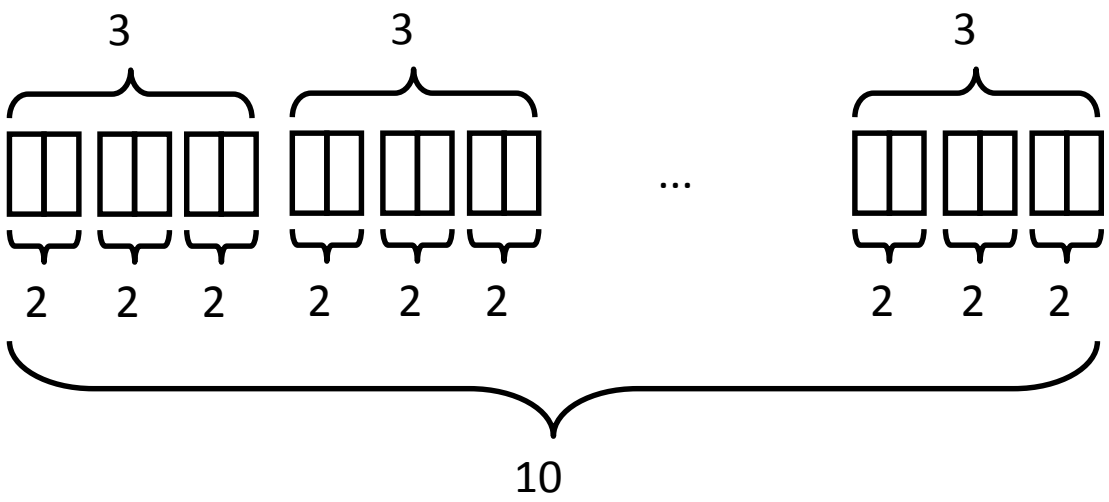
Multidimensional Arrays contd.

- Array declarations read right-to-left

  int a[10][3][2];

  "an array of ten arrays of three arrays of two ints"

- In memory



Summary

- Array is a collection of elements of same data type
- All these elements are stored in consecutive memory locations
- Array name is the same as the address of the array's first element
- The elements of a two-dimensional array are arranged in rows and columns

Further Reading

Dromey, R. (1982) *How To Solve it By Computer.* Noida: Pearson Education Inc.

Kernighan, B. W. and Richie, D. (1992) *The C Programming Language.* 2nd ed., New Delhi:PHI.