APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

STUDY MATERIALS

# MODULE 3

# ARRAY

Explanation:

Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Instead of declaring individual variables, such as number0, number1, ..., and number99, you declare one array variable such as numbers and use numbers[0], numbers[1], and ..., numbers[99] to represent individual variables. A specific element in an array is accessed by an index.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

## Declaring Arrays

To declare an array in C, a programmer specifies the type of the elements and the number of elements required by an array as follows

type arrayName [ arraySize ];

This is called a single-dimensional array. The arraySize must be an integer constant greater than zero and type can be any valid C data type. For example, to declare a 10-element array called balance of type double, use this statement −

double balance[10];

Here balance is a variable array which is sufficient to hold up to 10 double numbers.

## Initializing Arrays

You can initialize an array in C either one by one or using a single statement as follows

double balance[5] = {1000.0, 2.0, 3.4, 7.0, 50.0};

The number of values between braces { } cannot be larger than the number of elements that we declare for the array between square brackets [ ].

If you omit the size of the array, an array just big enough to hold the initialization is created. Therefore, if you write −

```
double balance[] = {1000.0, 2.0, 3.4, 7.0, 50.0};
```

You will create exactly the same array as you did in the previous example.

**processing of Arrays**

An element is accessed by indexing the array name. This is done by placing the index of the element within square brackets after the name of the array. For example −

```
double salary = balance[9];
```

The above statement will take the 10th element from the array and assign the value to salary variable. The following example shows how to use all the three above mentioned concepts viz. declaration, assignment, and accessing arrays.

**Using Arrays**

- Elements of an array are accessed by specifying the index ( offset ) of the desired element within square [ ] brackets after the array name.
- Array subscripts must be of integer type. ( int, long int, char, etc. )
- VERY IMPORTANT: Array indices start at zero in C, and go to one less than the size of the array. For example, a five element array will have indices zero through four. This is because the index in C is actually an offset from the beginning of the array. ( The first element is at the beginning of the array, and hence has zero offset. )
- Landmine: The most common mistake when working with arrays in C is forgetting that indices start at zero and stop one less than the array size.
- Arrays are commonly used in conjunction with loops, in order to perform the same calculations on all ( or some part ) of the data items in the array.

**2D and Multi Dimensional Array**

**Explanation:**

- Multi-dimensional arrays are declared by providing more than one set of square [ ] brackets after the variable name in the declaration statement.

- One dimensional arrays do not require the dimension to be given if the array is to be completely initialized. By analogy, multi-dimensional arrays do not require **the first** dimension to be given if the array is to be completely initialized. All dimensions after the first must be given in any case.
- For two dimensional arrays, the first dimension is commonly considered to be the number of rows, and the second dimension the number of columns. We will use this convention when discussing two dimensional arrays.
- Two dimensional arrays are considered by C/C++ to be an array of ( single dimensional arrays ). For example, "int numbers[ 5 ][ 6 ]" would refer to a single dimensional array of 5 elements, wherein each element is a single dimensional array of 6 integers. By extension, "int numbers[ 12 ][ 5 ][ 6 ]" would refer to an array of twelve elements, each of which is a two dimensional array, and so on.
- Another way of looking at this is that C stores two dimensional arrays by rows, with all elements of a row being stored together as a single unit. Knowing this can sometimes lead to more efficient programs.
- Multidimensional arrays may be completely initialized by listing all data elements within a single pair of curly {} braces, as with single dimensional arrays.
- It is better programming practice to enclose each row within a separate subset of curly {} braces, to make the program more readable. This is required if any row other than the last is to be partially initialized. When subsets of braces are used, the last item within braces is not followed by a comma, but the subsets are themselves separated by commas.
- Multidimensional arrays may be partially initialized by not providing complete initialization data. Individual rows of a multidimensional array may be partially initialized, provided that subset braces are used.

Individual data items in a multidimensional array are accessed by fully qualifying an array element. Alternatively, a smaller dimensional array may be accessed by partially qualifying the array name. For example, if "data" has been declared as a three dimensional array of floats, then data[ 1 ][ 2 ][ 5 ] would refer to a float, data[ 1 ][ 2 ] would refer to a one-dimensional array of floats, and data[ 1 ] would refer to a two-dimensional array of floats.

## Application of Array

Array is used for different verities of applications. Array is used to store the data or values of same data type. Below are the some of the applications of array –

A. Stores Elements of Same Data Type

Array is used to store the number of elements belonging to same data type.

```
int arr[30];
```

Above array is used to store the integer numbers in an array.

```
arr[0] = 10;

arr[1] = 20;

arr[2] = 30;

arr[3] = 40;

arr[4] = 50;
```

Similarly if we declare the character array then it can hold only character. So in short character array can store character variables while floating array stores only floating numbers.

B. Array Used for Maintaining multiple variable names using single name

Suppose we need to store 5 roll numbers of students then without declaration of array we need to declare following –

```
int roll1,roll2,roll3,roll4,roll5;
```

Now in order to get roll number of first student we need to access roll1.

Guess if we need to store roll numbers of 100 students then what will be the procedure.

Maintaining all the variables and remembering all these things is very difficult.

Consider the Array :

```
int roll[5];
```

So we are using array which can store multiple values and we have to remember just single variable name.

C. Array Can be Used for Sorting Elements

We can store elements to be sorted in an array and then by using different sorting technique we can sort the elements.

Different Sorting Techniques are :

Bubble Sort

Insertion Sort

Selection Sort

Bucket Sort

D. Array Can Perform Matrix Operation

Matrix operations can be performed using the array.We can use 2-D array to store the matrix.

[box]Matrix can be multi-dimensional[/box]

E. Array Can be Used in CPU Scheduling

CPU Scheduling is generally managed by Queue. Queue can be managed and implemented using the array. Array may be allocated dynamically i.e at run time. [Animation will Explain more about Round Robin Scheduling Algorithm | Video Animation]

F. Array Can be Used in Recursive Function

When the function calls another function or the same function again then the current values are stores onto the stack and those values will be retrieve when control comes back. This is similar operation like stack.

**String**

Strings are actually one-dimensional array of characters terminated by a null character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a null.

The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};

If you follow the rule of array initialization then you can write the above statement as follows

char greeting[] = "Hello";

C supports a wide range of functions that manipulate null-terminated strings −

| S.N. | Function & Purpose |
|---|---|
| 1 | **strcpy(s1, s2);**<br><br>Copies string s2 into string s1. String 1 have enough space to accommodate string 2. |
| 2 | **strcat(s1, s2);**<br><br>Concatenates string s2 onto the end of string s1. The second string is begin at the null character (‘/0’) of first string 1. |
| 3 | **strlen(s1);**<br><br>Returns the length of string s1 excluding null character. |
| 4 | **strcmp(s1, s2);**<br><br>Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2. |

*try it now*

A KTU STUDENTS PLATFORM

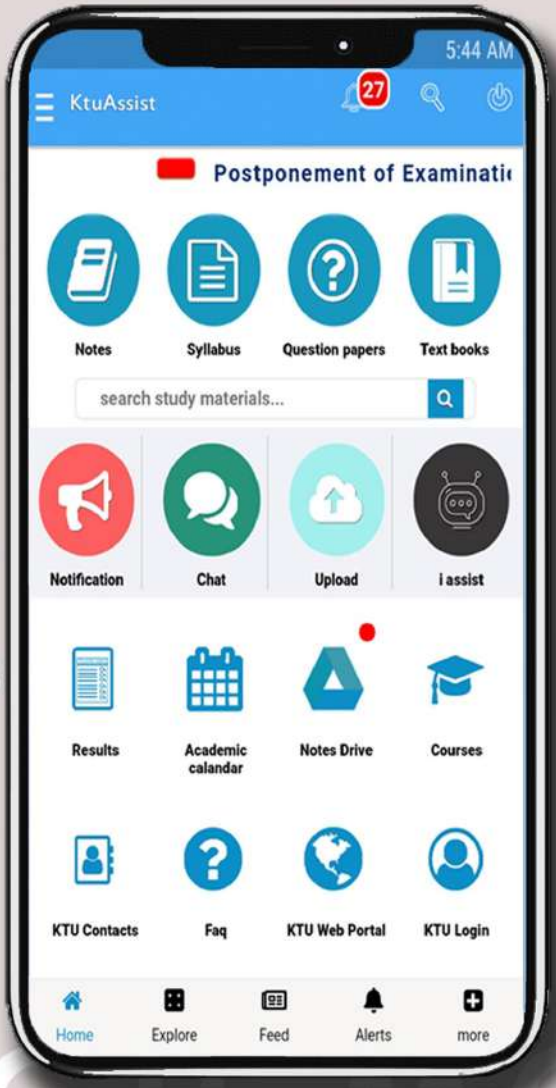
SYLLABUS

NOTES

TEXT BOOKS

QUESTION PAPERS

KTU NOTIFICATION




DOWNLOAD IT FROM GOOGLE PLAY

CHAT

FAQ

LOGIN

CALENDAR

MUCH MORE

DOWNLOAD APP

ktuassist.in

instagram.com/ktu_assist

facebook.com/ktuassist