SNS COLLEGE OF TECHNOLOGY

Coimbatore-35.
An Autonomous Institution

COURSE NAME : 19CST101 PROGRAMMING FOR PROBLEM SOLVING

I YEAR/ I SEMESTER

UNIT-III ARRAYS AND STRINGS

Topic: Array

Mrs. S.R.JANANI
Assistant Professor
Department of Computer Science and Engineering

Arrays

An array is defined as the collection of similar type of data items stored at contiguous memory locations. Arrays are the derived data type in C programming language which can store the primitive type of data such as int, char, double, float, etc. It also has the capability to store the collection of derived data types, such as pointers, structure, etc. The array is the simplest data structure where each data element can be randomly accessed by using its index number.

An array is a special type of variable used to store multiple values of same data type at a time.

An array can also be defined as follows...

An array is a collection of similar data items stored in continuous memory locations with single name.

Arrays

Declaration of an Array

In C programming language, when we want to create an array we must know the datatype of values to be stored in that array and also the number of values to be stored in that array.

We use the following general syntax to create an array...

```
datatype arrayName [ size ] ;
```

Syntax for creating an array with size and initial values

```
datatype arrayName [ size ] = {value1, value2, ...} ;
```

Syntax for creating an array without size and with initial values

```
datatype arrayName [ ] = {value1, value2, ...} ;
```

In the above syntax, the **datatype** specifies the type of values we store in that array and **size** specifies the maximum number of values that can be stored in that array.

Arrays

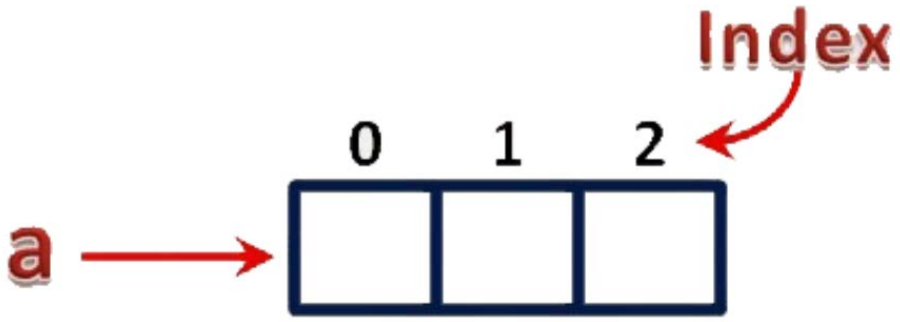
Example Code

```
int a [3] ;
```

Here, the compiler allocates 6 bytes of contiguous memory locations with a single name 'a' and tells the compiler to store three different integer values (each in 2 bytes of memory) into that 6 bytes of memory. For the above declaration, the memory is organized as follows...

Arrays

In the above memory allocation, all the three memory locations have a common name 'a'. So accessing individual memory location is not possible directly. Hence compiler not only allocates the memory but also assigns a numerical reference value to every individual memory location of an array. This reference number is called "Index" or "subscript" or "indices". Index values for the above example are as follows...



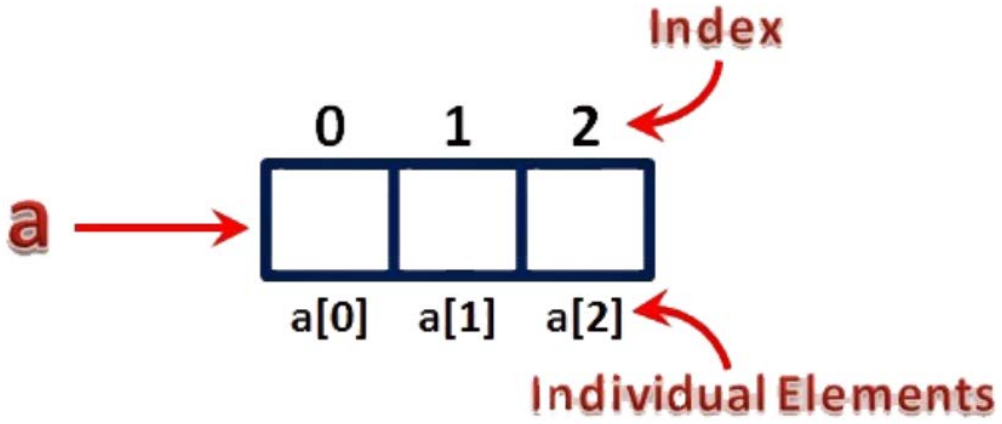
Arrays

Accessing Individual Elements of an Array

The individual elements of an array are identified using the combination of 'arrayName' and 'indexValue'. We use the following general syntax to access individual elements of an array...

```
arrayName [ indexValue ] ;
```

For the above example the individual elements can be denoted as follows...



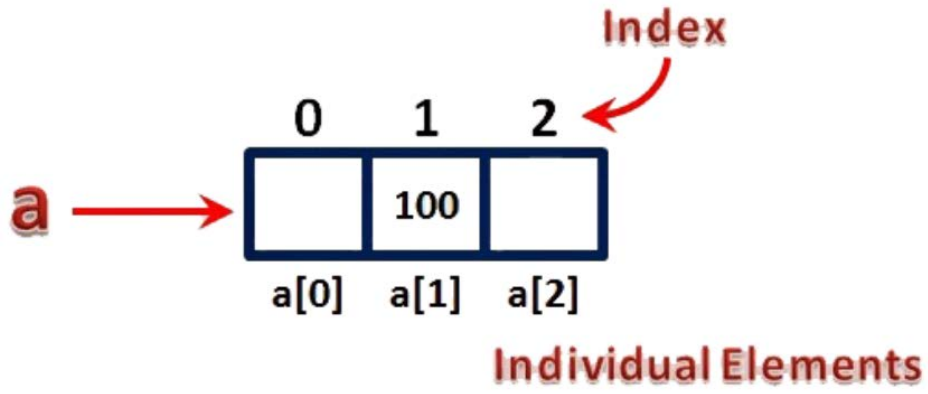
Arrays

For example, if we want to assign a value to the second memory location of above array 'a', we use the following statement...

Example Code

```
a [1] = 100 ;
```

The result of the above assignment statement is as follows...



Arrays

How to initialize an array?

It is possible to initialize an array during declaration. For example,

```
int mark[5] = {19, 10, 8, 17, 9};
```

You can also initialize an array like this.

```
int mark[] = {19, 10, 8, 17, 9};
```

Here, we haven't specified the size. However, the compiler knows its size is 5 as we are initializing it with 5 elements.

Arrays

Here, we haven't specified the size. However, the compiler knows its size is 5 as we are initializing it with 5 elements.

mark[0]	mark[1]	mark[2]	mark[3]	mark[4]
19	10	8	17	9

Here,

```
mark[0] is equal to 19
mark[1] is equal to 10
mark[2] is equal to 8
mark[3] is equal to 17
mark[4] is equal to 9
```

Arrays

Change Value of Array elements

```
int mark[5] = {19, 10, 8, 17, 9}

// make the value of the third element to -1
mark[2] = -1;

// make the value of the fifth element to 0
mark[4] = 0;
```

Input and Output Array Elements

Here's how you can take input from the user and store it in an array element.

```
// take input and store it in the 3rd element
scanf("%d", &mark[2]);

// take input and store it in the ith element
scanf("%d", &mark[i-1]);
```

Arrays

Here's how you can print an individual element of an array.

```
// print the first element of the array
printf("%d", mark[0]);

// print the third element of the array
printf("%d", mark[2]);

// print ith element of the array
printf("%d", mark[i-1]);
```

Arrays

Example 1: Array Input/Output

```c
// Program to take 5 values from the user and store them in an array
// Print the elements stored in the array
#include <stdio.h>

int main() {
  int values[5];

  printf("Enter 5 integers: ");

  // taking input and storing it in an array
  for(int i = 0; i < 5; ++i) {
     scanf("%d", &values[i]);
  }

  printf("Displaying integers: ");

  // printing elements of an array
  for(int i = 0; i < 5; ++i) {
     printf("%d\n", values[i]);
  }
  return 0;
}
```

Arrays

Output

```
Enter 5 integers: 1
-3
34
0
3
Displaying integers: 1
-3
34
0
3
```

Here, we have used a `for` loop to take 5 inputs from the user and store them in an array. Then, using another `for` loop, these elements are displayed on the screen.

Arrays

Example 2: Calculate Average

```c
// Program to find the average of n numbers using arrays

#include <stdio.h>
int main()
{
     int marks[10], i, n, sum = 0, average;

     printf("Enter number of elements: ");
     scanf("%d", &n);

     for(i=0; i<n; ++i)
     {
          printf("Enter number%d: ",i+1);
          scanf("%d", &marks[i]);

          // adding integers entered by the user to the sum variable
          sum += marks[i];
     }

     average = sum/n;
     printf("Average = %d", average);

     return 0;
}
```

Arrays

Output

```
Enter n: 5
Enter number1: 45
Enter number2: 35
Enter number3: 38
Enter number4: 31
Enter number5: 49
Average = 39
```

Here, we have computed the average of `n` numbers entered by the user.

Arrays

Properties of Array

The array contains the following properties.

- Each element of an array is of same data type and carries the same size, i.e., int = 4 bytes.
- Elements of the array are stored at contiguous memory locations where the first element is stored at the smallest memory location.
- Elements of the array can be randomly accessed since we can calculate the address of each element of the array with the given base address and the size of the data element.

Characteristics of array

- An array holds elements that have the same datatype
- Array elements are stored in subsequent memory locations
- Two dimensional array elements are stored row by row in subsequent memory locations
- Array name represents the address of the starting element
- Array size should be mentioned in the declaration. Array size must be constant expression and not a variable.

Arrays

Advantage of C Array

1) Code Optimization: Less code to the access the data.

2) Ease of traversing: By using the for loop, we can retrieve the elements of an array easily.

3) Ease of sorting: To sort the elements of the array, we need a few lines of code only.

4) Random Access: We can access any element randomly using the array.

Disadvantage of C Array

1) Fixed Size: Whatever size, we define at the time of declaration of the array, we can't exceed the limit. So, it doesn't grow the size dynamically like LinkedList which we will learn later.


Thank You!