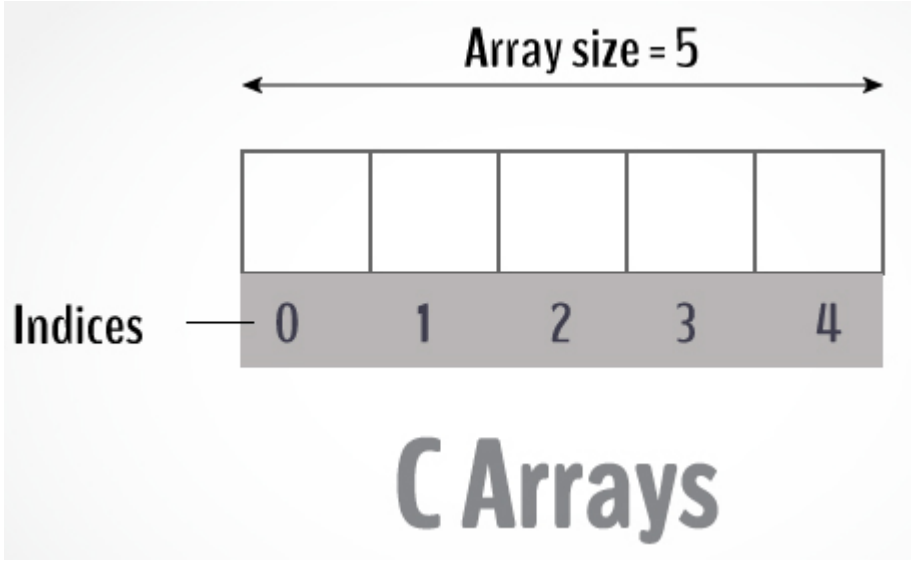
Array

GBS

- **C language provides the capability for the programmer to define a set of ordered data items. An array is just the same to a list.**
- **Just like variables, arrays must be declared before they are used. Giving name to an array takes also the rules of variable naming. Declaration of an array involves declaring the data type of values that will be stored(int, char, float) and indicating also the maximum number of elements(dimension) that will be stored.**
- **Example:**
- **int values[10];**
- **char letters[5];**
- **float values[20];**



- An array is a variable that can store multiple values. For example, if you want to store 100 integers, you can create an array for it.

```
int data[100];
```

How to declare an array?

Syntax:

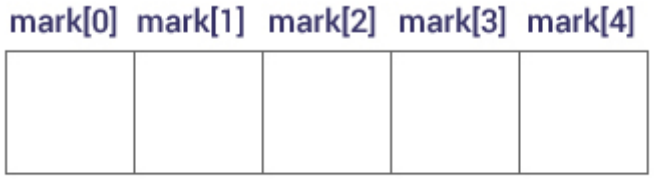
```
dataType arrayName[arraySize];
```

```
float mark[5];
```

- Here, we declared an array, mark, of floating-point type. And its size is 5. Meaning, it can hold 5 floating-point values.
- It's important to note that the size and type of an array cannot be changed once it is declared.

Access Array Elements

- You can access elements of an array by indices.
- Suppose you declared an array mark as above. The first element is mark[0], the second element is mark[1] and so on.



Few keynotes:

•Arrays have 0 as the first index, not 1. In this example, mark[0] is the first element.

•If the size of an array is n, to access the last element, the n-1 index is used.

•In this example, mark[4] suppose the starting address of mark[0] is **2120d**. Then, the address of the mark[1] will be **2124d**. Similarly, the address of mark[2] will be **2128d** and so on.

This is because the size of a float is 4 bytes.

How to initialize an array?

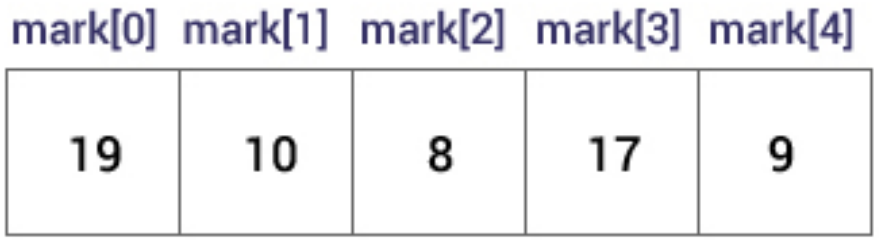
It is possible to initialize an array during declaration. For example,

```
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.



An individual array element can be used just like a variable.

Example:

```
num[0] = 100;

num[1] = num[0];

printf("%d", num[1]);
```

Array elements can also be used in calculations. They are also operands.

Example:

```
sum = num[1] + num[2];

total = values[9] + sum;

num[3]++;

num[3+1];
```

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]);
```

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]);
```

```
1.// Program to take 5 values from the user and store them in an array
2.// Print the elements stored in the array
3.#include <stdio.h>
5.int main() {
6.int values[5];
8.printf("Enter 5 integers: ");
10.// taking input and storing it in an array
11.for(int i = 0; i < 5; ++i) {
12.scanf("%d", &values[i]);
13.}
15.printf("Displaying integers: ");
17.// printing elements of an array
18.for(int i = 0; i < 5; ++i) {
19.printf("%d\n", values[i]);
20.}
21.return 0;
22.}
```

```
1.// Program to find the average of n numbers using arrays
3.#include <stdio.h>
4.int main()
5.{
6.int marks[10], i, n, sum = 0, average;
8.printf("Enter number of elements: ");
9.scanf("%d", &n);
11.for(i=0; i<n; ++i)
12.{
13.printf("Enter number%d: ",i+1);
14.scanf("%d", &marks[i]);
16.// adding integers entered by the user to the sum variable
17.sum += marks[i];
18.}
20.average = sum/n;
21.printf("Average = %d", average);
23.return 0;
24.}
```

Multidimensional Arrays

- In C programming, you can create an array of arrays. These arrays are known as multidimensional arrays. For example,

```
float x[3][4];
```

- Here, x is a two-dimensional (2d) array. The array can hold 12 elements. You can think the array as a table with 3 rows and each row has 4 columns.

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

Initialization of a 2d array

```
1.  // Different ways to initialize two-dimensional array
2.
3.  int c[2][3] = {{1, 3, 0}, {-1, 5, 9}};
4.
5.  int c[][3] = {{1, 3, 0}, {-1, 5, 9}};
6.
7.  int c[2][3] = {1, 3, 0, -1, 5, 9};
```

```c
// C program to find the sum of two matrices of order 2*2

#include <stdio.h>
int main()
{
  float a[2][2], b[2][2], result[2][2];

  // Taking input using nested for loop
  printf("Enter elements of 1st matrix\n");
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      printf("Enter a%d%d: ", i + 1, j + 1);
      scanf("%f", &a[i][j]);
    }

  // Taking input using nested for loop
  printf("Enter elements of 2nd matrix\n");
  for (int i = 0; i < 2; ++i)
    for (int j = 0; j < 2; ++j)
    {
      printf("Enter b%d%d: ", i + 1, j + 1);
      scanf("%f", &b[i][j]);
    }
```

```
    // adding corresponding elements of two arrays
    for (int i = 0; i < 2; ++i)
        for (int j = 0; j < 2; ++j)
        {
            result[i][j] = a[i][j] + b[i][j];
        }

    // Displaying the sum
    printf("\nSum Of Matrix:");

    for (int i = 0; i < 2; ++i)
        for (int j = 0; j < 2; ++j)
        {
            printf("%.1f\t", result[i][j]);

            if (j == 1)
                printf("\n");
        }
    return 0;
}
```

```
Enter elements of 1st matrix
Enter a11: 2;
Enter a12: 0.5;
Enter a21: -1.1;
Enter a22: 2;
Enter elements of 2nd matrix
Enter b11: 0.2;
Enter b12: 0;
Enter b21: 0.23;
Enter b22: 23;

Sum Of Matrix:
2.2         0.5
-0.9        25.0
```