# Arrays in C

An array is a contiguous collection of objects of the same type. For example: if you want to store 100 integers in sequence, you can create an array for it.

## Array Declaration

Like other variables an array needs to be declared so that the compiler will know what kind of an array and how large an array we want. We declare an Array are by stating the following:

- The type of the values to be stored
- A variable identifier (The name of the array)
- The number of elements in the array

**Example**: The following statement declares an array called ***array_1*** to store 100 ***integer*** values

```
int array_1[100];
```

Note that the size of the array has to be specified when declaring an array; otherwise, an error will be generated.

## Array Initialization

Arrays may be initialized when they are declared, just as any other variables. The initialization data placed on the curly braces {}. If an array is to be completely initialized, the dimension of the array is not required, the compiler will automatically size the array to fit the initialized data.

```
int num[6] = { 2, 4, 12, 5, 45, 5 } ;
int n[] = { 2, 4, 12, 5, 45, 5 } ;
float press[] = { 12.3, 34.2 -23.4, -11.3 } ;
```

### Accessing Elements of an Array

Once an array is declared, let us see how individual elements in the array can be referred. This is done with subscript, the number in the brackets [] following the array name. This number specifies the element's position in the array. All the array elements are numbered, starting with 0. Thus, `press[2]` in the previous example is not the second element of the array, but the third.

**Example:** The following C-code shows how to read the elements of an array from the keyboard, using a *scanf* statement and a *for* loop to cycle through each element in turn.

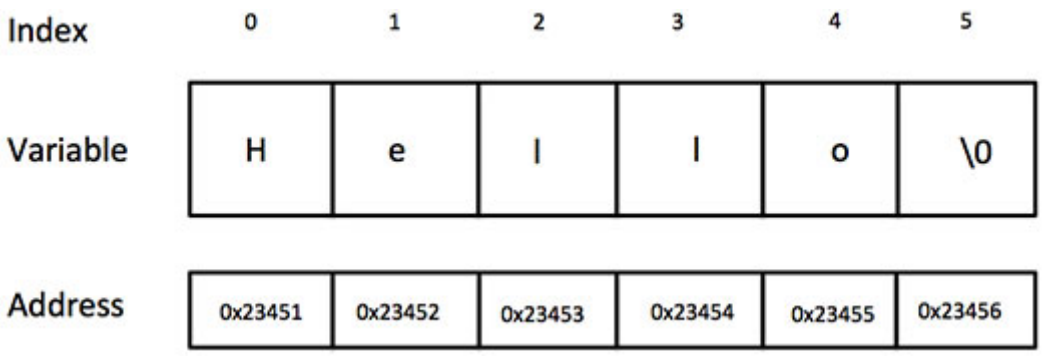
```
#include<stdio.h>
int main()
{
    int arr[4];
    int i, j;
    printf("Enter array element\n");
    for(i = 0; i < 4; i++)
    {
        scanf("%d", &arr[i]) //Run time array initialization
    }
    printf("The array elements that you Enter as follow\n");
    for(j = 0; j < 4; j++)
    {
        printf("Element %d has value %d\n",j+1,arr[j]);
    }
    return 0;
}
```

## Strings in C

Strings are actually one-dimensional Array of characters terminated by a **null character '\0'**. 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."

```
#include <stdio.h>
int main () {
   char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
   char array_name[] = "My Name Is Jacob";
   printf("Greeting message: %s\n", greeting );
   printf("%s\n", array_name );
   printf( "%c" , array_name[ 6 ] );
   return 0;
}
```

Following is the memory presentation of the above defined string in C/C++



## Examples

1- The program below prompts the user to enter his / her name and displays it on the screen.

```
#include<stdio.h>
int main( void )
{
int i = 0;
char array_1[ ] = { 'H' , 'E' , 'L' , 'L' , 'O' , '\0' };
char array_2[ ] = "Write your name";
char array_3[ 30 ];
printf( "%s\n" , array_1 );
while( array_2[ i ] != '\0' )
{
printf( "%c" , array_2[ i ] );
i++;
}
printf( "\n" );
scanf( "%s" , array_3 );
printf( "Your name is : %s" , array_3 );
return 0;
}
```

**2-** The following program uses loops and arrays to calculate the first twenty Fibonacci numbers

```
#include <stdio.h>
int main( void ) {
int i, fibonacci[ 20 ];
fibonacci[ 0 ] = 0;
fibonacci[ 1 ] = 1;
for( i = 2; i < 20; i++ ){
fibonacci[ i ] = fibonacci[ i - 2 ] + fibonacci[ i - 1 ];
}
 for( i = 0; i < 20; i++ ){
 printf( "Fibonacci[ %d ] = %d\n", i, fibonacci[ i ] );
}
return 0;
}
```

**3-** Program to find the average of n (n < 10) numbers using arrays

```
#include <stdio.h>
int main()
{
     int marks[10], i, n, sum = 0;
      float average;
     printf("Enter n: ");
     scanf("%d", &n);
     for(i=0; i<n; ++i)
     {
          printf("Enter number%d: ",i+1);
          scanf("%d", &marks[i]);
          sum += marks[i];
     }
     average = sum/n;
     printf("Average = %f", average);
     return 0;
}
```

## Passing an Array to a Function

- Passing a single element of an array to a function is similar to passing variable to a function.

**Example:**

```
#include <stdio.h>
void display(int age)
{
    printf("%d", age);
}
int main()
{
    int ageArray[] = {2, 3, 4};
    display(ageArray[2]); //Passing array element ageArray[2]
    return 0;
}
```

❖ Passing an entire array to a function

```
#include <stdio.h>
float average(float age[]);
int main()
{
     float avg, age[] = {23.4, 55, 22.6, 3, 40.5, 18};
     avg = average(age); // Only name of an array is
passed as an argument
     printf("Average age = %f", avg);
     return 0;
}
float average(float age[])
{
     int i;
     float avg, sum = 0.0;
     for (i = 0; i < 6; ++i) {
          sum += age[i];
     }
     avg = (sum / 6);
     return avg;
}
```

To pass an entire array to a function, only the name of the array is passed as an argument. However, notice the use of [] after argument name in float average (float age []). This informs the compiler that you are passing a one-dimensional array to the function.

## Two dimensional Arrays

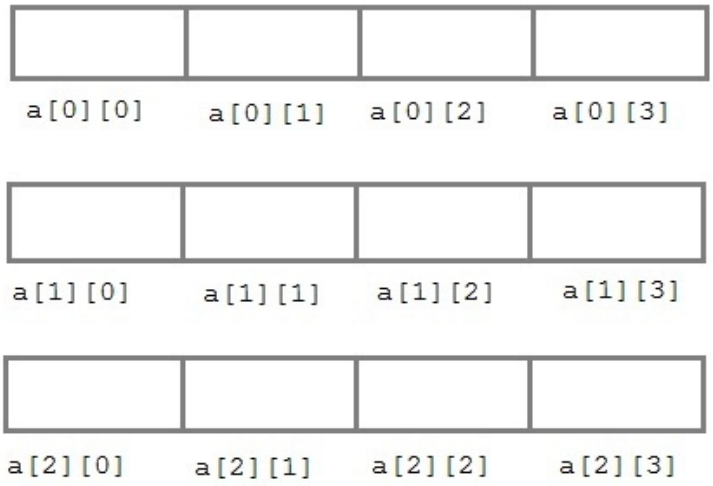
C language supports multidimensional arrays also. The simplest form of a multidimensional array is the two-dimensional array. Both the row's and column's index begins from 0.

Two-dimensional arrays are declared as follows

```
data-type array-name[row-size][column-size]
```

Example

```
int a[3][4];
```




A 2-dimension array can also be declared and initialized together. For example,

```
int arr[][3] = {
    {0,0,0},
    {1,1,1}
};
```

We have not assigned any row value to our array in the above example. It means we can initialize any number of rows. But, we must always specify number of columns, else it will give a compile time error. Here, a 2*3 multi-dimensional matrix is created.

**Example**

```
#include<stdio.h>
int main()
{
    int arr[3][4];
    int i, j, k;
    printf("Enter array element\n");
    for(i = 0; i < 3;i++)
    {
        for(j = 0; j < 4; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }
    printf("the matrix you Entered is\n");
    for(i = 0; i < 3; i++)
    {
        for(j = 0; j < 4; j++)
        {
            printf("%d     ", arr[i][j]);
        }
           printf("\n");
    }
    return 0;
}
```