Tutorial Link
https://codequotient.com/tutorials/Arrays/59fde2a0e63d6b7fd5dec004

TUTORIAL

Arrays

Chapter

1. Arrays

Topics

While we use variables in our programs, we need a variable for one purpose. Sometime we need to store same kind of data for many different entities. so C allows us to store it in different and efficient manner i.e. Array. Array is a structured data type in C. An array is defined as finite ordered collection of homogeneous data providing random access of memory.

- It is **finite** because data size must be defined.
- It is **ordered** because all the elements need to be stored in memory at contiguous locations.
- It is **homogeneous** because all the elements must be of same type.

So instead of declaring individual variables we can define a single variable as an array and use it for all variables i.e. instead of storing age of 100 people in age1, age2, age3 ... age99 (declaring so many variables and managing them may result in typographical errors in program also.) we can define an array age of size 100 and can store all the 100 values in it. Later we can access all the individual elements by a single name 'age' and its **index** in the array as shown in the following figure: -

Value	55	67	45	...	62
Index	0	1	2	...	99
Address	1000	1004	1008	...	1396

The above figure illustrate the storage of an array age of size 100 with some random values whose initial address in main memory is 1000 (**Base address of array**) and each element of the array taking 4 bytes. In C, the first element will be stored at index 0, as C uses the offset style of declaration.

Array will provide the random access in memory instead of other data structures like linked list because in array the elements will store physically at contiguous locations and each element is taking same amount of memory. So we can directly access age[19] to access the 20th element of the age array. This random access makes many operations faster and cheaper in problem solving.

Declaring arrays

Like any other variable, arrays must be declared before they are used. To declare an array in C/C++, we have to provide the data type, name and size of the array as follows: -

```
Type array_name[size];
```

This is one-dimensional array just like the RAM of the computer. The size must be an integer expression having value greater than 0. For example, to declare an 100 element age array we have to write: -

```
int age[100];
```

For declaring arrays in Java, we have to dynamically allocate the memory like,

```
type array_name[] = new type[size];
int age[] = new int[100]; // Declares an array age of 100 integer
elements
```

Initialization of array

After declaring we must initialize the array with values before using them, otherwise some garbage values will be used. Array can be initialized at both times (compile time and run time). The general form of array initialization at compile time is:

```
Type array_name[size] = {value1, value2, …} // provide a list of
values of length size.
i.e. int age[5] = {43, 65, 76, 35, 45};
```

if we provide more values than the size of array then compiler will throw an compile time error or it will ignore the extra values, it depends on compiler so we have to take care of it. We can omit the size in case of compile time initialization and compiler will take the size automatically as we provide the values i.e.

```
int age[] = {34, 43, 45, 23, 54, 34, 65, 23, 44, 34}; //
automatically creates an array of size 10.
```

For accessing the array elements we can access individual element by referring as age[0], age[1] and so on, so generally we will iterate over the array with iteration as shown in below program: -

C

```c
#include <stdio.h>
int main()
{
        int i;
        int age[] = {34, 43, 45, 23, 54, 34, 65, 23, 44, 34};
                // automatically creates an array of size 10.
        for(i=0 ; i<10 ; i++) {
      printf("%d\t", age[i]);
    }
    return 0;
}
```

```

```

Java

```
import java.util.Scanner;
// Other imports go here
// Do NOT change the class name
class Main
{
    public static void main(String[] args)
    {
        // Write your code here
        int i;
        int age[] = {34, 43, 45, 23, 54, 34, 65, 23, 44, 34};
                    // automatically creates an array of size 10.
        for(i=0 ; i<10 ; i++) {
            System.out.print(age[i]+"\t");
        }
    }
}
```

C++

```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
int main(){
        int i;
        int age[] = {34, 43, 45, 23, 54, 34, 65, 23, 44, 34};
                // automatically creates an array of size 10.
        for(i=0 ; i<10 ; i++) {
      cout << age[i] << '\t' ;
    }
    return 0;
}
```

The output will be:

```
34   43   45   23   54   34   65   23   44   34
```

We can initialize the array elements at compile time individually also as shown in following example: -

C

```c
#include <stdio.h>
int main()
{
        int i;
        int age[5];
        age[0]= 34;
        age[1]= 43;
        age[2]= 45;
        age[3]= 23;
        age[4]= 54;
        for(i=0 ; i<5 ; i++) {
                printf("%d\t", age[i]);
        }
    return 0;
}
```

Java

```java
import java.util.Scanner;
// Other imports go here
// Do NOT change the class name
class Main
{
    public static void main(String[] args){

        int i;
        int age[] = new int[5];
        age[0]= 34;
        age[1]= 43;
        age[2]= 45;
        age[3]= 23;
        age[4]= 54;
        for(i=0 ; i<5 ; i++) {
            System.out.print(age[i]+"\t");
        }

    }
}

```

C++

```
#include<iostream>
#include<cstdio>
#include<cmath>
using namespace std;
int main(){
        int i;
        int age[5];
        age[0]= 34;
        age[1]= 43;
        age[2]= 45;
        age[3]= 23;
        age[4]= 54;
        for(i=0 ; i<5 ; i++) {
                cout<< age[i]<< '\t';
        }
    return 0;
}
```

The output will be:

```
34   43   45   23   54
```

Second way to initialize the array is run time initialization, where we can use input functions for taking the values from user and store them in the array as shown in the following example: -

```
#include <stdio.h>
int main()
{
   int age[10], i;
    printf("Enter the age of 10 persons :");
    for(i=0; i<10; i++)
   {
      scanf("%d", &age[i]);       //Run time array initialization
using scanf function.
   }
    for(i=0; i<10; i++)
   {
      printf("%d\t", age[i]);
   }
   return 0;
}
```

The above program will read 10 integers from user and store them in the array, in the 2nd loop the array elements are printed one by one using printf function.

Video Explanation

<iframe width="560" height="315" src="https://www.youtube.com/embed/U_H-9mO0WnI" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture" allowfullscreen></iframe>