C Programming Short Notes

Part 5

1. Arrays

- Arrays in C are fundamental data structures that allow you to store **multiple elements of the same data type sequentially in memory.**
- They provide a convenient way to work with **collections of data** and are **widely used in programming**.

Declaration and Initialization:

In C, arrays are declared using the following syntax:

datatype arrayName[arraySize];

Here,

- **datatype** specifies the type of elements the array will hold.
- **arrayName** is the identifier for the array.
- **arraySize** denotes the number of elements the array can store.
- Arrays can be initialized at the time of declaration or later using assignment statements.

Accessing Elements:

- Elements within an array are accessed **using zero-based indexing**, array always starts with 0.
- For example, to access the third element of an array named myArray, you would use myArray[2].
- C does not perform bounds checking, so accessing elements outside the bounds of the array can lead to undefined behavior.

Array Operations:

- ✓ Traversal: Arrays can be traversed using loops such as for or while loops to access and process each element sequentially.
- ✓ Modification: Elements of an array can be modified by assigning new values to them using their indices.

Types of arrays

- ❖ One – Dimensional array
- ❖ Two – Dimensional array
- ❖ Multi – Dimensional array

One – Dimensional array:

- One-dimensional arrays are the most common type of array in C.
- They consist of a single row or sequence of elements (columns).

Declaration: datatype arrayName[arraySize];

Here the arraysize is indicates the number of columns,

```
int a[5]={1,3,4,5,6};
      |
   columns
```

array example :

```
#include<stdio.h>
int main(){
        int a[4]={1,2,3,4};
        for(int i=0;i<4;i++)
        {
                printf("%d ",a[i]);
        }
}
```

Output

```
1 2 3 4
```

Adding two array and get the input using scanf()

```
#include<stdio.h>
int main()
{
   int a[10];
   int b[10];
   int c[10];
   int size;
   printf("Enter the size of array : ");
   scanf("%d",&size);
   printf("\nEnter the 1st array :");
   for(int i=0;i<size;i++)
   {
      scanf("%d",&a[i]);
   }
   printf("\nEnter the 2nd array :");
```

```
    for(int i=0;i<size;i++)
    {
        scanf("%d",&b[i]);
    }
    printf("\nThe A matrix is : ");
    for(int i=0;i<size;i++)
    {
        printf("%d ",a[i]);
    }
    printf("\nThe B matrix is : ");
    for(int i=0;i<size;i++)
    {
        printf("%d ",b[i]);
    }
    printf("\n\nThe A + B matrix is : ");
    for(int i=0;i<size;i++)
    {
        c[i]=a[i]+b[i];
        printf("%d ",c[i]);
    }
    printf("\n\n\n");
}
```

Output:

```
Enter the size of array : 4

Enter the 1st array :1 2 3 4

Enter the 2nd array :5 6 7 8

The A matrix is : 1 2 3 4
The B matrix is : 5 6 7 8

The A + B matrix is : 6 8 10 12
```

Two dimensional arrays

- It has multiple rows and multiple columns
- A two-dimensional array can be thought of as a matrix with rows and columns.

Declaration: datatype arrayName[rows][columns];

Hear the 1st denotes how many rows we need and 2nd denotes how many columns we need

```
int a[3][3]={1,2,3,4,5,6,7,8,9};
       |   |
   rows  columns
```

```
a[0][0] => 1
a[0][1] => 2
a[0][2] => 3
a[1][0] => 4
a[1][1] => 5
a[1][2] => 6
a[2][0] => 7
a[2][1] => 8
a[2][2] => 9
```

3 3	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

Two dimensional array example 1:

```
int main()
{
   int i,j;
         int a[3][3]={1,2,3,4,5,6,7,8,9};
         for(i=0;i<3;i++){
                  for(j=0;j<3;j++){
                           printf("%d ",a[i][j]);
                  }
                  printf("\n");
         }
         return 0;
}
```

Output

```
1 2 3
4 5 6
7 8 9
```

Example 2 :

```
int main()
{
   int i,j;
         int a[3][3];
         printf("Enter the array of 3 rows and 3 columns : \n");
         for(i=0;i<3;i++){
                 for(j=0;j<3;j++){
        printf("a[%d][%d] = ",i,j);
                         scanf("%d",&a[i][j]);
                 }
         }
         printf("The matrix is : \n");
         for(i=0;i<3;i++){
                 for(j=0;j<3;j++){
                         printf("%d ",a[i][j]);
                 }
                 printf("\n");
         }
         return 0;
}
```

Output:

```
Enter the array of 3 rows and 3 columns :
a[0][0] = 1
a[0][1] = 2
a[0][2] = 3
a[1][0] = 4
a[1][1] = 5
a[1][2] = 6
a[2][0] = 7
a[2][1] = 8
a[2][2] = 9
The matrix is :
1 2 3
4 5 6
7 8 9
```

Multi-dimensional array

- Multidimensional arrays store data in multiple dimensions, typically multiple array, multiple rows and multiple columns.

Declaration: datatype arrayName[array][rows][columns];

```
int a[2][2][3]={1,2,3,4,5,6,7,8,9,10,11,12};
         |   |   |
 arrays  |   coloumns
            rows
```

(OR)

```
int a[2][2][3]={{1,2,3,4,5,6},{7,8,9,10,11,12}};
```

array	C R	0	1	2
0	0	1	2	3
	1	4	5	6
1	0	7	8	9
	1	10	11	12

Multi-dimensional array example :

```
int main()
{
   int a[2][2][3]={1,2,3,4,5,6,7,8,9,10,11,12};
   int i,j,k;
   for(i=0;i<2;i++)
   {
      printf("\nThe %d array is : \n",i+1);
      for(j=0;j<2;j++)
      {
         for(k=0;k<3;k++)
         {
            printf("%d ",a[i][j][k]);
         }
         printf("\n");
      }}}
```

Output

```
The 1 array is :
1 2 3
4 5 6

The 2 array is :
7 8 9
10 11 12
```