

# INSTITUTE OF ENGINEERING

Pulchowk Campus, Lalitpur



Subject: C Programming

Lab Report 8

Title: **Arrays**

Submitted by:

Susheel Thapa 077BCT090

Submitted to:

Department of Electronics  
and

Computer Engineering

Checked by

# **Content of Lab Report:**

## **Background Information**

C Programming

Editor Used

Compiler

C Array

## **Code and Output**

Source Code

Output

## **Analysis**

## **Conclusion**

# Background Information

## What is C Programming?

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is the most widely used computer language.

## Why to Learn C Programming?

- Easy to learn
- Structured language
- It produces efficient programs
- It can handle low-level activities
- It can be compiled on a variety of computer platforms

## Editor

Here, I have used Visual Studio Code as my editor. You can download the editor from [Download Visual Studio Code - Mac, Linux, Windows](#) . Select your operating system and download it.

## Compiler

Here, I have used **gcc** as my compiler provided by MinGWw64. You can download it via [Download MinGW-w64 - for 32 and 64 bit Windows from SourceForge.net](#). Your download will start automatically. Run the downloaded .exe file. After, you have installed MinGW-w64, you need to configure it.

- In the Windows search bar, type 'settings' to open your Windows Settings.
- Search for Edit environment variables for your account.
- Choose the Path variable and then select Edit.
- Select New and add the Mingw-w64 destination folder path to the system path. The exact path depends on which version of Mingw-w64 you have installed and where you installed it. If you used the settings above to install Mingw-w64, then add this to the path: **C:\Program Files\mingw-w64\x86\_64-8.1.0-posix-seh-rt\_v6-rev0\mingw64\bin.**
- Select OK to save the updated PATH. You will need to reopen any console windows for the new PATH location to be available.

## Check your installation

Open command prompt or power shell and type:

```
C:\Users\user>gcc --version
gcc (x86_64-posix-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
C:\Users\user>gcc
gcc: fatal error: no input files
compilation terminated.

C:\Users\user>_
```

If you get similar result, you are good to go.

## What is Array?

An array is a collection of one or more values of the same type. Each value is called an element of the array. The elements of the array share the same variable name but each element has its own unique index number (also known as a subscript). An array can be of any type, For example: int, float, char etc. If an array is of type int then its elements must be of type int only

To store roll no. of 100 students, we have to declare an array of size 100 i.e roll\_no[100]. Here size of the array is 100, so it is capable of storing 100 values. In C, index or subscript starts from 0, so roll\_no[0] is the first element, roll\_no[1] is the second element and so on. Note that the last element of the array will be at roll\_no[99] not at roll\_no[100] because the index starts at 0.

Arrays can be single or multidimensional. The number of subscript or index determines the dimensions of the array. An array of one dimension is known as a one-dimensional array or 1-D array, while an array of two dimensions is known as a two-dimensional array or 2-D array.

Let's start with a one-dimensional array.

## One Dimensional Array

Conceptually you can think of a one-dimensional array as a row, where elements are stored one after another.

Syntax:

```
datatype array_name[size];
```

datatype: It denotes the type of the elements in the array.

array\_name: Name of the array. It must be a valid identifier.

size: Number of elements an array can hold. here are some example of array declarations:

```
int num[100];float temp[20];char ch[50];
```

num is an array of type int, which can only store 100 elements of type int.

temp is an array of type float, which can only store 20 elements of type float.

ch is an array of type char, which can only store 50 elements of type char.

Note: When an array is declared it contains garbage values.

## Two Dimensional Array

The syntax declaration of 2-D array is not much different from 1-D array. In 2-D array, to declare and access elements of a 2-D array we use 2 subscripts instead of 1.

Syntax: `datatype array_name[ROW][COL];`

The total number of elements in a 2-D array is ROW\*COL. Let's take an example.

```
int arr[2][3];
```

This array can store  $2*3=6$  elements. You can visualize this 2-D array as a matrix of 2 rows and 3 columns.

The individual elements of the above array can be accessed by using two subscript instead of one. The first subscript denotes row number and second denotes column number. As we can see in the above image both rows and columns are indexed from 0. So the first element of this array is at arr[0][0] and the last element is at arr[1][2]. Here are how you can access all the other elements:

arr[0][0] - refers to the first element

arr[0][1] - refers to the second element

arr[0][2] - refers to the third element

arr[1][0] - refers to the fourth element

arr[1][1] - refers to the fifth element

arr[1][2] - refers to the sixth element

If you try to access an element beyond valid ROW and COL , C compiler will not display any kind of error message, instead, a garbage value will be printed. It is the responsibility of the programmer to handle the bounds.

arr[1][3] - a garbage value will be printed, because the last valid index of COL is 2

arr[2][3] - a garbage value will be printed, because the last valid index of ROW and COL is 1 and 2 respectively

Just like 1-D arrays, we can only also use constants and symbolic constants to specify the size of a 2-D array.

## **One Dimension Array**

**1]Write,observe and study the following code**

**Source Code:**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```

void main()
{

    int i, num[6] = {4, 5, 3, 2, 12,8};

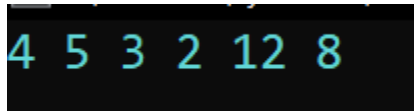
    system("cls");

    for (int i = 0; i < 6; i++)
    {
        printf("%d ", num[i]);
    }

    getch();
}

```

**Output:**



```

4 5 3 2 12 8

```

**2].Write,observe and study the following code**

**Source Code:**

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main()
{
    int i, num[6];

```

```

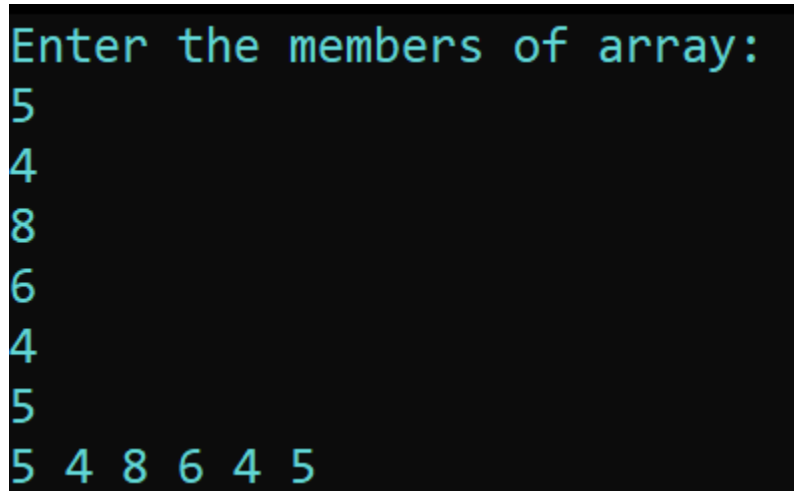
system("cls");

printf("Enter the members of array: \n");
for (int i = 0; i < 6; i++)
{
    scanf("%d", &num[i]);
}
for (int i = 0; i < 6; i++)
{
    printf("%d ", num[i]);
}

getch();
return 0;
}

```

**Output:**



The screenshot shows a terminal window with a black background and green text. The first line is the prompt "Enter the members of array:". This is followed by six lines of input, each containing a single digit: 5, 4, 8, 6, 4, and 5. The final line of the output shows these six digits printed in a single line, separated by spaces: "5 4 8 6 4 5".

**3]Write a program find the sum of elements of an integers array of size 5 that are divisible by 10 not by 15.**

**Source Code:**



```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int main()
{
    /*Declaration of variable*/
    int i, num[5], sum = 0;

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: ");
    for (int i = 0; i < 5; i++)
    {
        scanf("%d", &num[i]);
    }

    /*Check the condition of every number and adding them if they meet condition asked by question*/
    for (int i = 0; i < 5; i++)
    {
        (num[i] % 10 == 0 && num[i] % 15 != 0) ? (sum = num[i] + sum) : (sum = sum);
    }

    /*Printing the result*/
    printf("The sum of number present in the array divisible by 10 not by 15 is %d",sum);

    getch();
    return 0;
}
```

### Output:

```
Enter the members of array: 20
40
30
45
60
The sum of number present in the array divisible by 10 not by 15 is 60.
```

4]Write a program to add the elements at the corresponding position of two array of size n. Read value of n from user

### Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#include "../Array.c" /*External file which contain related to one d array*/

int main()
{
    /*Declaration of needed variable*/
    int number1[100], number2[100], number[100], size;

    system("cls");

    /*Asking the size of array*/
    printf("Size of the array: ");
    scanf("%d", &size);

    /*Taking input of array one and array two*/
    printf("\nArray one:\n");
    inputOneDArray(number1, size);

    printf("\nArray Two:\n");
    inputOneDArray(number2, size);
```

```
/*Adding both element of array*/
for (int i = 0; i < size; i++)
{
    number[i] = number1[i] + number2[i];
}

/*Displaying result*/
printf("\nSum of Two Array:\n");
displayOneDArray(number, size);

getch();
return 0;
}
```

**Output:**

```
Size of the array: 4

Array one:
Value of 1 element is 4
Value of 2 element is 5
Value of 3 element is 6
Value of 4 element is 7

Array Two:
Value of 1 element is 1
Value of 2 element is 5
Value of 3 element is 8
Value of 4 element is 7

Sum of Two Array:
5 10 14 14
```

5]Write a program to find the highest or lowest elements of an array of size 5

**Source Code:**

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Declaration of variable*/
    int i, num[5], lowest, highest;

    system("cls");
```

```

/*Taking input from user*/

printf("Enter the members of array: \n");
for (int i = 0; i < 5; i++)
{
    scanf("%d", &num[i]);
}

/*Initialization of lowest and highest*/

lowest = num[0];
highest = num[0];

/*Iterate over all elements of array and find lowest and highest number from array and store to a variable*/

for (int i = 1; i < 6; i++)
{
    (lowest < num[i]) ? (lowest = lowest) : (lowest = num[i]);
    (highest > num[i]) ? (highest = highest) : (highest = num[i]);
}

/*Printing result*/

printf("\nHighest Number is %d\n", highest);
printf("Lowest Number is %d", lowest);

getch();

return 0;
}

```

**Output:**

```
Enter the members of array:
45
1245
14
458
45

Highest Number is 1245
Lowest Number is 5
```

**6]Write a program to read the element of array in main() pass it to fucntion to sort the array in ascending/descending order. Display the sorted array from main().**

#### **Source Code**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
void sortArray(int num[], char sort[5])
```

```
{
```

```
    "Sort the array in ascending or descending order based on second argument passed by  
    user. Here, we use bubble sort algorithm.";
```

```
    int temp;
```

```
    for (int i = 0; i < 5; i++)
```

```
    {
```

```
for (int j = 0; j < 5; j++)
{
    if (strcmp(sort, "Ascend") == 0)
    {
        if (num[i] > num[j])
        {
        }
        else
        {
            temp = num[i];
            num[i] = num[j];
            num[j] = temp;
        }
    }
    else if (strcmp(sort, "Descend") == 0)
    {
        if (num[i] < num[j])
        {
        }
        else
        {
            temp = num[i];
            num[i] = num[j];
            num[j] = temp;
        }
    }
}
```

```

    }
}

int main()
{
    /*Declaration of variable*/
    int i, num[5], lowest, highest;

    system("cls");

    /*Taking input from the user*/
    printf("Enter the members of array: ");
    for (int i = 0; i < 5; i++)
    {
        scanf("%d ", &num[i]);
    }

    sortArray(num, "Ascend"); /*Function call(pass by refrence) to sort array ascending order*/

    /*Printing the sorted array*/
    printf("\nSorted in Ascending Order: ");
    for (int i = 0; i < 5; i++)
    {
        printf("%d ", num[i]);
    }
}

```



```
    sortArray(num, "Descend"); /*Function call(pass by refrence) to sort array
descending order*/
```

```
/*Printing the sorted array*/
```

```
printf("\n\nSorted in Descending Order: ");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

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

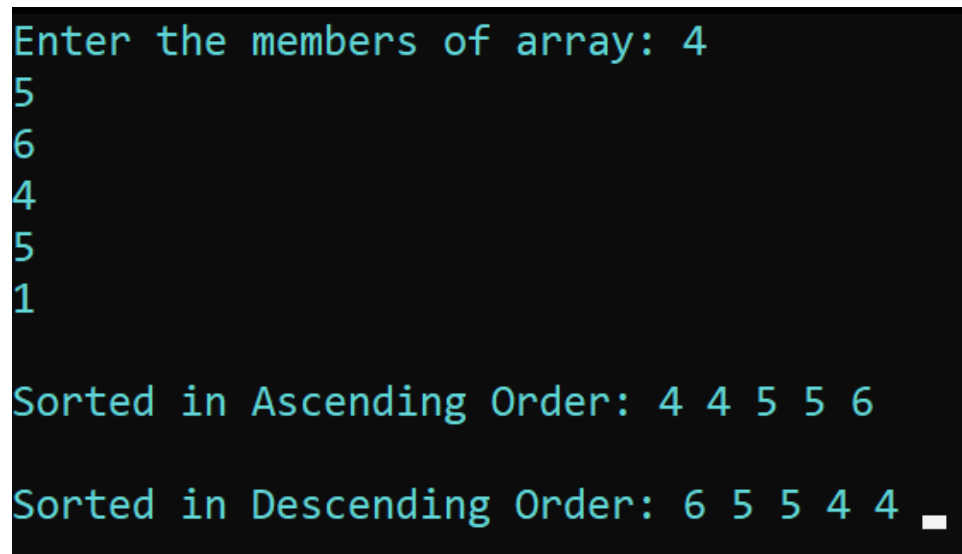
```
}
```

```
getch();
```

```
return 0;
```

```
}
```

**Output:**



```
Enter the members of array: 4
5
6
4
5
1

Sorted in Ascending Order: 4 4 5 5 6

Sorted in Descending Order: 6 5 5 4 4
```

**7]Write a program to raise the power of each member by 3**

**Source Code:**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int main()
```

```
{
```

```
/*Declaration of variable*/
```

```
int i, num[5];
```

```
system("cls");
```

```
/*Taking input from user*/
```

```
printf("Enter the members of array: \n");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

```
scanf("%d", &num[i]);
```

```
}
```

```
/*Raising the value of each element to power of 3 of respective element*/
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

```
num[i] = num[i] * num[i] * num[i];
```

```
}
```

```
/*Printing the new array(raised by 3)*/
```

```
printf("\nNew Array: ");
```

```
for (int i = 0; i < 5; i++)
```

```
{
```

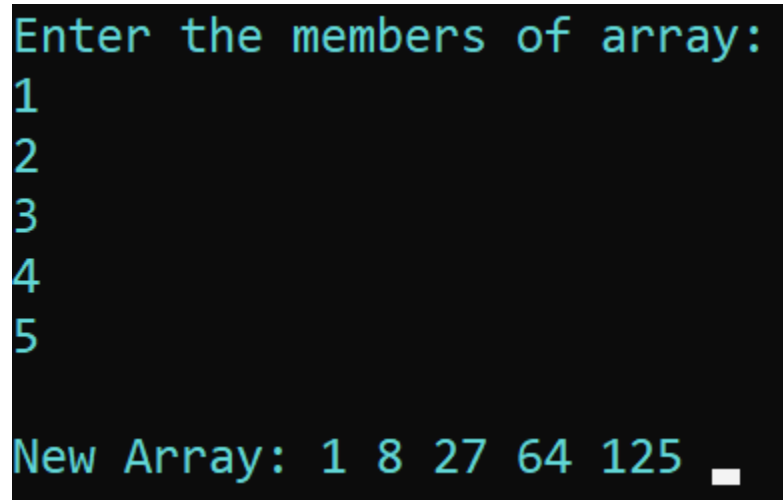
```
printf("%d ", num[i]);
```

```
}
```

```
getch();
```

```
    return 0;
}
```

**Output:**



```
Enter the members of array:
1
2
3
4
5

New Array: 1 8 27 64 125
```

**8]Write a program to read an unsigned integer array from main and pass it to a function that count armstrong members and return the count to main()\*/**

**Source Code:**

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
```

```
int armstrongArray(int num[])
```

```
{
```

```
    "Takes an array count the occurance of armstrong number and return it";
```

```
    int count = 0, armstrong = 0, n, rem;
```

```
    for (int i = 0; i < 5; i++)
```

```
    {
```

```
        n = num[i];
```

```
        if (n == 0)
```

```
        {
```

```
            count++;
```

```

    }
    else
    {

        while (n != 0)
        {
            rem = n % 10;

            armstrong = n * n * n + armstrong;

            if (num[i] == armstrong)
            {
                count++;
            }

            n = n / 10;
        }

        armstrong = 0;
    }
}

return count;
}

int main()
{
    /*Declaration of variable*/
    unsigned int num[5];

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: ");

```

```

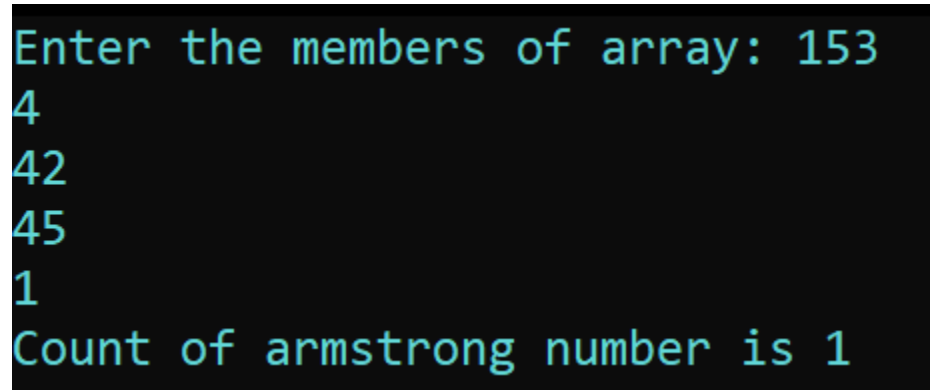
    for (int i = 0; i < 5; i++)
    {
        scanf("%d", &num[i]);
    }

    /*Print the value of count of armstrong numbers in array afer function call*/
    printf("Count of armstrong number is %d", armstrongArray(num));

    getch();
    return 0;
}

```

**Output:**



The screenshot shows the program's execution. It prompts the user to enter the members of an array. The user enters 153, 4, 42, 45, and 1. The program then outputs the count of Armstrong numbers, which is 1.

```

Enter the members of array: 153
4
42
45
1
Count of armstrong number is 1

```

**Program Array.c**

```

#include <math.h>

void inputOneDArray(int array[], int array_length)
{
    "Takes two argument array and length of array. Ask user to input the value in respective array
    element";

    for (int i = 0; i < array_length; i++)
    {
        printf("Value of %d element is ", i + 1);
    }
}

```

```

        scanf("%d", &array[i]);
    }
}

void sortOneDArrayAscend(int array[], int array_length)
{
    "Sorted the array in ascending order";

    for (int i = 0; i < array_length; i++)
    {
        for (int j = 0; j < array_length; j++)
        {
            if (array[i] < array[j])
            {
                array[i] = array[i] + array[j] - (array[j] = array[i]);
            }
        }
    }
}

```

```

void sortOneDArrayDescend(int array[], int array_length)
{
    "Sorted the array in ascending order";

    for (int i = 0; i < array_length; i++)
    {
        for (int j = 0; j < array_length; j++)
        {
            if (array[i] > array[j])
            {

```

```

        array[i] = array[i] + array[j] - (array[j] = array[i]);
    }
}
}
}

```

```

float medianOneDArray(int array[], int array_length)

```

```

{
    "Calculate median and return value of median";

    float position_of_array = (array_length + 1) / 2.0;

    if (position_of_array - (int)position_of_array == 0.0)
    {
        return array[(int)position_of_array - 1];
    }
    else
    {
        return array[(int)position_of_array] + array[(int)position_of_array - 1] / 2.0;
    }
}

```

```

int smallOneDArray(int array[], int array_length)

```

```

{
    "Finds the smallest elements in array";

    int small = array[0];

    for (int i = 1; i < array_length; i++)
    {

```

```
        if (small > array[i])
        {
            small = array[i];
        }
    }

    printf("small = %d\n", small);

    return small;
}

int largeOneDArray(int array[], int array_length)
{
    "Finds the largest element of array";

    int large = array[0];

    for (int i = 1; i < array_length; i++)
    {
        if (large < array[i])
        {
            large = array[i];
        }
    }

    printf("large = %d\n", large);

    return large;
}
```



```
int sumOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculates the sum of elements of array";
```

```
    int sum = 0;
```

```
    for (int i = 0; i < array_length; i++)
```

```
    {
```

```
        sum = sum + array[i];
```

```
    }
```

```
    return sum;
```

```
}
```

```
float meanOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculate the mean of elements of integer array";
```

```
    float mean = 0.0;
```

```
    mean = sumOneDArray(array, array_length) / (float)array_length;
```

```
    return mean;
```

```
}
```

```
float sdOneDArray(int array[], int array_length)
```

```
{
```

```
    "Calculates the standard deviation of elements of integer array";
```

```
    float sd, diverse;
```

```

float mean = meanOneDArray(array, array_length);

for (int i = 0; i < array_length; i++)
{
    diverse = (mean - array[i]) * (mean - array[i]) + diverse;
}

sd = pow(diverse / array_length, 0.5);

return sd;
}

float varianceOneDArray(int array[], int array_length)
{
    "Calculate the variance of elements of integer array";

    float variance = pow(sdOneDArray(array, array_length), 2);

    return variance;
}

void displayOneDArray(int array[], int array_length)
{
    "Display the array elements";

    for (int i = 0; i < array_length; i++)
    {
        printf("%d ", array[i]);
    }
}

```

```
}
```

### **9.1]Program to compute median of number in array\*/**

#### **Source Code:**

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 9 /*Defining a constant used in variable*/

#include "../Array.c" /*External files that help to do operation in one d array*/

int main()
{
    /*Array declartion*/
    int median[ARRAY_LENGTH];

    system("cls");

    /*Function call to take value of array from user*/
    inputOneDArray(median, ARRAY_LENGTH);

    /*Function call to sort the array in ascending order*/
    sortOneDArrayAscend(median, ARRAY_LENGTH);

    /*Function call to find the median of number present in the array and print the value of median*/
    printf("The median of array provided is %.2f.", medianOneDArray(median, ARRAY_LENGTH));

    getch();
    return 0;
```

```
}
```

**Output:**

```
Value of 1 element is 4
Value of 2 element is 8
Value of 3 element is 7
Value of 4 element is 2
Value of 5 element is 6
Value of 6 element is 9
Value of 7 element is 4
Value of 8 element is 3
Value of 9 element is 8
The median of array provided is 6.00.
```

## 9.2]Program to compute range of one d array

**Source Code:**

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 8 /*Defining constant*/

#include "../Array.c" /*External files that helps to do operation in one d array*/

int main()
{
    /*Variable declaration*/
    int input[ARRAY_LENGTH], small, large, range;

    /*Taking input from user*/
    inputOneDArray(input, ARRAY_LENGTH);
```

```

/*Function call to find smallest value and storing value in small*/
small = smallOneDArray(input, ARRAY_LENGTH);

/*Function call to find largest value and storing value in large*/
large = largeOneDArray(input, ARRAY_LENGTH);

/*Calculating range*/
range = (large - small);

/*Printing result*/
printf("Range of given data is %d", range);

getch();
return 0;
}

```

**Output:**

```

Value of 1 element is 1
Value of 2 element is 5
Value of 3 element is 8
Value of 4 element is 7
Value of 5 element is 6
Value of 6 element is 9
Value of 7 element is 15
Value of 8 element is 14
small = 1
large = 15
Range of given data is 14

```

### 9.3]Program to compute standard deviation

**Source Code:**

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>


#define ARRAY_LENGTH 5 /*Defining a constant*/


#include "../Array.c"/*External file which have function to do operations in one d array*/


int main()
{
    /*Variable declaration*/
    int input[ARRAY_LENGTH];
    float sd;


    /*Calling function to take input*/
    inputOneDArray(input, ARRAY_LENGTH);


    system("cls");


    /*Calling function to calculate Standard Deviation*/
    sd = sdOneDArray(input, ARRAY_LENGTH);


    /*Printing the result*/
    printf("Standard Deviation is %f\n", sd);


    getch();
    return 0;
}
```

**Output:**

```
Value of 1 element is 10
Value of 2 element is 48
Value of 3 element is 95
Value of 4 element is 15
Value of 5 element is 42
```

```
Standard Deviation is 30.324907
```

#### 9.4]Program to compute variance

##### Source Code:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

#define ARRAY_LENGTH 5 /*Defining constant*/

#include "../Array.c" /*External files that contain function to do operation in one d array*/

int main()
{
    /*Variable Declaration*/
    float variance;
    int input[10];

    /*Calling function to take input*/
    inputOneDArray(input, ARRAY_LENGTH);

    system("cls");
```

```

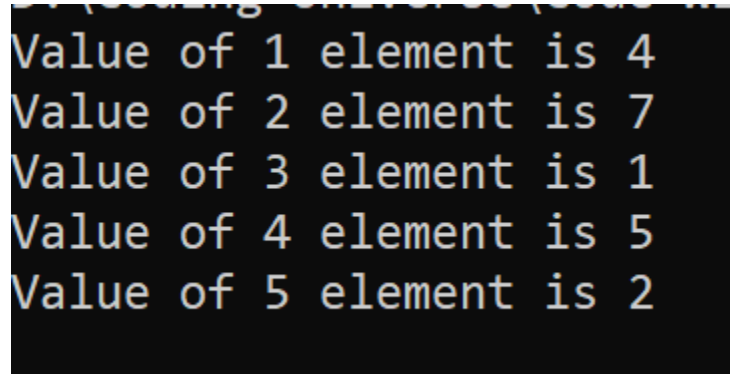
/*Calling function to calculated variance*/
variance = varianceOneDArray(input, ARRAY_LENGTH);

/*Printing result*/
printf("Variance is %.2f", variance);

getch();
return 0;
}

```

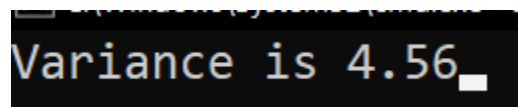
**Output:**



```

Value of 1 element is 4
Value of 2 element is 7
Value of 3 element is 1
Value of 4 element is 5
Value of 5 element is 2

```



```

Variance is 4.56_

```

## Two Dimensional Array

1]Type, run and observe the work flow

**Source:**

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

```

```

void main()
{

```



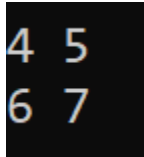
```
int i, j, num[2][2] = {{4, 5}, {6, 7}};
```

```
for (int i = 0; i < 2; i++)  
{  
    for (int j = 0; j < 2; j++)  
    {  
        printf("%d ", num[i][j]);  
    }  
    printf("\n");  
}
```

```
getch();
```

```
}
```

**Output:**



```
4 5  
6 7
```

**2]Type, run and observe the work flow**

**Source Code:**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
void main()
```

```
{
```

```
    int i, j, num[3][3];
```

```
    for (int i = 0; i < 3; i++)
```

```
    {
```

```
        for (int j = 0; j < 3; j++)
```

```
    {  
        scanf("%d", &num[i][j]);  
    }  
}  
  
for (int i = 0; i < 3; i++)  
{  
    for (int j = 0; j < 3; j++)  
    {  
        printf("%d ", num[i][j]);  
    }  
    printf("\n");  
}  
getch();  
}
```

**Output:**

```
1
2
3

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

**3]Write a program find the sum of elements of an integers array of size 5 that are divisible by 10 not by 15**

**Source Code:**

```
#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

int main()

{

    /*Variable declaration*/

    int i, num[3][3], sum = 0;


    system("cls");


    /*Taking input from user*/
```

```

printf("Enter the members of array:\n");

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        scanf("%d", &num[i][j]);
    }
}

/*Iterate over every element and check condition is match or not. if match then it keep on adding those
value*/

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        (num[i][j] % 7 == 0 && num[i][j] % 5 != 0) ? (sum = num[i][j] + sum) : (sum = sum);
    }
}

/*Prints the result*/

printf("The sum of number present in the array divisible by 7 not by 5 is %d", sum);

getch();

return 0;
}

```

**Output:**

```
Enter the members of array:
14
35
28
26
15
45
70
7
26
The sum of number present in the array divisible by 7 not by 5 is 49.
```

**4]Write a program to find the highest or lowest elements of an array of size 3X3\*/**

**Source Code:**

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int main()
{
    /*Variable Declaration*/
    int i, num[3][3], lowest, highest;

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: \n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &num[i][j]);
        }
    }
}
```

```

/*Initialization of lowest and highest*/

lowest = num[0][0];
highest = num[0][0];

/*Iterate over every element and find the highest and lowest storing into a variable*/
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        (lowest < num[i][j]) ? (lowest = lowest) : (lowest = num[i][j]);
        (highest > num[i][j]) ? (highest = highest) : (highest = num[i][j]);
    }
}

/*Prints the result*/

printf("\nHighest Number is %d", highest);
printf("\n\nLowest Number is %d", lowest);

getch();

return 0;

}

```

**Output:**

Enter the members of array:

4

58

74

54

15

24

45

1

4

Highest Number is 74

Lowest Number is 1

6]Write a program to read the members of 3X3 array in main(). Pass the array to function that finds the sum of diagonal elements and returns to main(). Display the returned values

**Source Code:**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int diagonalSum(int num[3][3])
```

```
{
```

```
    "Calculates the sum of elements of diagonal elements";
```

```
    int sum = 0;
```

```
    /*Iterate over every element and sum all the diagonal element i.e a(i,j) where i = j*/
```

```
    for (int i = 0; i < 3; i++)
```

```
{  
    for (int j = 0; j < 3; j++)  
    {  
        (i == j) ? (sum = sum + num[i][j]) : (sum = sum);  
    }  
}  
  
return sum;  
}  
  
int main()  
{  
    /*Variable declaration*/  
    int i, num[3][3], lowest, highest;  
  
    system("cls");  
  
    /*Takes input from user*/  
    printf("Enter the members of array: \n");  
    for (int i = 0; i < 3; i++)  
    {  
        for (int j = 0; j < 3; j++)  
        {  
            scanf("%d", &num[i][j]);  
        }  
    }  
  
    /*Call the function and prints its result*/  
    printf("Sum of diagonol elements is %d", diagonolSum(num));  
}
```



**Output:**

```
Enter the members of array:
1
2
3
4
5
6
7
8
9
Sum of diagonal elements is 15
```

**7]Write a program to raise the power of elements by 5**

**Source Code:**

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
#include <stdlib.h>
```

```
int powerOfNumber(int base, int power)
```

```
{
```

```
    "Raise the n power of number";
```

```
    int value = 1;
```

```
    for (int i = 1; i <= power; i++)
```

```
    {
```

```
        value = base * value;
```

```
    }
```

```
    return value;
```

```
}
```

```
int main()
{
    /*Variable declaration*/
    int i, num[3][3];

    system("cls");

    /*Taking input from user*/
    printf("Enter the members of array: \n");
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            scanf("%d", &num[i][j]);
        }
    }

    /*Raising power with function call*/
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            num[i][j] = powerOfNumber(num[i][j], 5);
        }
    }

    /*Printing the new array formed*/
    for (int i = 0; i < 3; i++)
    {
```

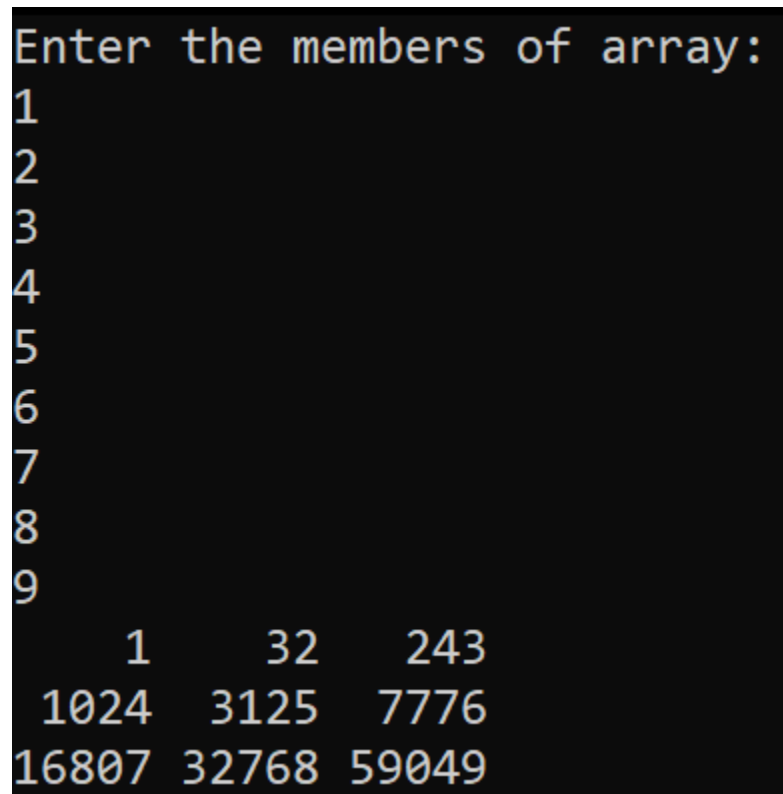
```

    for (int j = 0; j < 3; j++)
    {
        printf("%5d ", num[i][j]);
    }
    printf("\n");
}

getch();
return 0;
}

```

**Output:**



```

Enter the members of array:
1
2
3
4
5
6
7
8
9

      1      32      243
1024  3125  7776
16807 32768 59049

```

8] Write program to generate a matrix of size 4 \* 4 whose elements are given by the expression  $a(i,j) = 4^{(-1(i+j))}$

**Source Code**

```
#include <stdio.h>
```

```
#include <conio.h>
#include <stdlib.h>
#include <math.h>

int main()
{
    /*Declaration of variable*/
    float num[4][4];

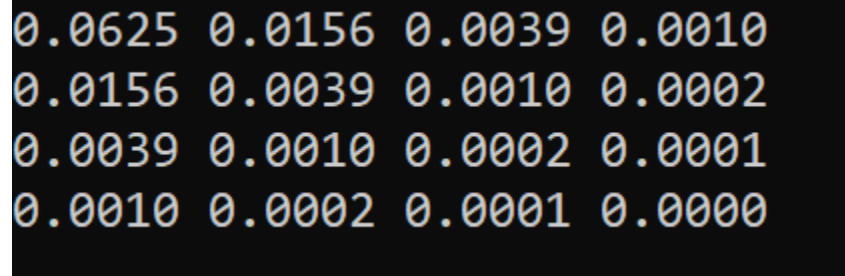
    system("cls");

    /*Creating array from the condition given by the user*/
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            num[i][j] = pow(4, -1 * (i + j + 2));
        }
    }

    /*Printing the formed array*/
    for (int i = 0; i < 4; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            printf("%.4f ", num[i][j]);
        }
        printf("\n");
    }
}
```

```
    getch();  
    return 0;  
}
```

**Output:**



```
0.0625 0.0156 0.0039 0.0010  
0.0156 0.0039 0.0010 0.0002  
0.0039 0.0010 0.0002 0.0001  
0.0010 0.0002 0.0001 0.0000
```

**9]Write a program to find the sum of individual rows of two dimensional array and assign them to a one dimensional array and display the content of the two dimensional array**

**Source Code:**

```
#include <stdio.h>  
  
#include <conio.h>  
  
#include <stdlib.h>  
  
  
int main()  
{  
    /*Variable declaration*/  
    int sum, numSum[10], num[10][10];  
  
    system("cls");  
  
    /*Create a array without own logic*/  
    printf("Enter the members of array: ");  
    for (int i = 0; i < 10; i++)  
    {  
        for (int j = 0; j < 10; j++)  
        {
```

```

        num[i][j] = (i)*10 + (j);
    }
}

/*Calculating the sum of element of row and storing it in a one d array*/
for (int i = 0; i < 10; i++)
{
    sum = 0;
    for (int j = 0; j < 10; j++)
    {
        sum = sum + num[i][j];
    }
    numSum[i] = sum;
}

/*Printing the two d array*/
printf("Two D Array: \n");
for (int i = 0; i < 10; i++)
{
    for (int j = 0; j < 10; j++)
    {
        printf("%3d ", num[i][j]);
    }
    printf("\n");
}

/*Printing the one d array which contain sum of elements of row*/
printf("\nSum Array: \n");
for (int i = 0; i < 10; i++)
{

```

```

        printf("%d\n", numSum[i]);
    }

    getch();

    return 0;
}

```

**Output:**

```

Enter the members of array: Two D Array:
 0  1  2  3  4  5  6  7  8  9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69
70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89
90 91 92 93 94 95 96 97 98 99

Sum Array:
45
145
245
345
445
545
645
745
845
945

```

**10]WAP to read matrix A and matrix B of size mXn using a function called read\_matrix. Pass it to a function named process\_matrix. This function should perform the multiplication of the matrices and display the result using display\_matrix. You should give appropriate message to the user if multiplication of two matrix is not possible.**

**Source Code:**

```

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

```

```
void readMatrix(int array[][100], int cols, int rows)
```

```
{
```

```
    "Reads matrix value from user...";
```

```
    for (int i = 0; i < rows; i++)
```

```
    {
```

```
        for (int j = 0; j < cols; j++)
```

```
        {
```

```
            printf("Value of element in %d row and %d column: ", i + 1, j + 1);
```

```
            scanf("%d", &array[i][j]);
```

```
        }
```

```
    }
```

```
    system("cls");
```

```
}
```

```
void processMatrix(int matrix_one[][100], int matrix_two[][100], int matrix_result[][100], int rows, int  
common, int cols)
```

```
{
```

```
    "Multiply two matrices...";
```

```
    for (int i = 0; i < rows; i++)
```

```
    {
```

```
        for (int j = 0; j < cols; j++)
```

```
        {
```

```
            matrix_result[i][j] = 0;
```

```
            for (int k = 0; k < common; k++)
```

```
            {
```

```
                matrix_result[i][j] += matrix_one[i][k] * matrix_two[k][j];
```



```
    }  
    }  
}  
}
```

```
void displayMatrix(int matrix [][100], int rows, int cols)
```

```
{  
    "Display the matrices...";  
  
    for (int i = 0; i < rows; i++)  
    {  
        for (int j = 0; j < cols; j++)  
        {  
            printf("%d ", matrix[i][j]);  
        }  
        printf("\n");  
    }  
}
```

```
int main ()
```

```
{  
    int matrix_one[100][100], matrix_two[100][100], matrix_multi[100][100];  
    int cols_one, rows_one, cols_two, rows_two;
```

```
system("cls");
```

```
/*Taking value of Matrix One*/
```

```
printf("Reading Matrix One: \n");
```

```
printf("Rows: ");
```

```
scanf("%d", &rows_one);
```

```

printf("Cols: ");
scanf("%d", &cols_one);
readMatrix(matrix_one, rows_one, cols_one);

/*Taking value of Matrix Two*/
printf("Reading Matrix Two: \n");
printf("Rows: ");
scanf("%d", &rows_two);
printf("Cols: ");
scanf("%d", &cols_two);
readMatrix(matrix_two, rows_two, cols_two);

if (cols_one == rows_two)
{
    processMatrix(matrix_one, matrix_two, matrix_multi, rows_one, cols_one, cols_two);
    displayMatrix(matrix_multi, rows_one, cols_two);
}
else if (cols_two == rows_one)
{
    processMatrix(matrix_two, matrix_one, matrix_multi, rows_two, cols_two, cols_one);
    displayMatrix(matrix_multi, rows_two, cols_one);
}
else
{
    printf("\nIt cannot be multiplied.");
}

Getch ();
return 0;
}

```

### Output:

```
Reading Matrix One:
Rows: 2
Cols: 2
Value of element in 1 row and 1 column: 1
Value of element in 1 row and 2 column: 8
Value of element in 2 row and 1 column: 6
Value of element in 2 row and 2 column: 8_
```

```
Reading Matrix Two:
Rows: 2
Cols: 2
Value of element in 1 row and 1 column: 1
Value of element in 1 row and 2 column: 7
Value of element in 2 row and 1 column: 8
Value of element in 2 row and 2 column: 9
```

```
65 79
70 114
```

### Analysis

From this lab we have learn how array is use in C programing language how it is declared initialized, how we can access it, modification and many more thng . We learn also learn what kind of operatio can be done the array. We also use 2 d array like matric and perform many calculation of matric through program.

### Conclusion

From this lab we became familiar with array in C.