UTKAL GOURAV MADHUSUDAN INSTITUTE OF TECHNOLOGY, RAYAGDA

C Programming
&
Web Design

Lab Manual designed by

Sisira Kumar Kapat,
Lecturer (Comp. Appl.), UGMIT, Rayagada.
2023

This document presents the experiments which are needed to the students of Diploma, 3rd Semester, E&TC. The students can be able to learn the C Programming using Turbo C editor; also they will be designing the various web pages using HTML programming. This manual is in accordance with SCTE&VT syllabus.

C programming & Web Page Design Lab

Expt. No. **1** **Date:**

Aim of the experiment: To display our college name twenty times on the screen.

Apparatus required: Computer system with turbo C editor

Theory:

Printf statement is used in the program as output statement, and scanf statement is used as the input statement in C Programming.

Program:

```
#include<stdio.h>
void main()
{
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
        printf("UGMIT\n");
}
```

Output:

```
UGMIT
UGMIT
UGMIT
UGMIT.... 20 times.
```

Conclusion: I have successfully displayed the college name 20 times on the screen using loop.

Signature of the student

Expt. No. 2 **Date:**

Aim of the experiment:

i. To understand the Looping Concept. (While, do-while, for loop) of C Programming.
ii. Write “Hello World”, 20 times on the screen using loop.

Apparatus required: Computer system with turbo C editor

Theory:

We can write the program in two ways, such as, normal approach or by using Loop.

Normal approach:

- In normal approach we can write the printf statement 20 times to execute the sentence “Hello World”.
- This is a time consuming process.
- Creates a mesh if the same statement is repeated multiple times.
- Increases the lines of code of the program.

Using Loop:

- Loop can be used to execute the same statement as many times as our wish.
- Though the number of repetition increases, the lines of code of the program remain same.
- Hence we shall use loops to perform this experiment.

Loops in C : There are three types of loops available in C such as,

1. While
2. Do-while
3. For

while loop

The syntax for while loop is,

```
while(condition)
{
        statements;
}
```

do-while loop

The syntax for while loop is,

```
do
{
        statements;
}while(condition);
```

for loop

The syntax for while loop is,

```
for(initialization; condition; termination)
{
        statements;
}
```

Program:

Using while loop

```
// Write “Hello World”, 20 times on the screen using while loop.

     #include<stdio.h>
     void main()
     {
          int i=1;
          while(i<21)
          {
               printf(“hello world\t”);
               i++;
          }
     }
```

Using do-while loop

```
// Write “Hello World”, 20 times on the screen using do-while loop.

     #include<stdio.h>
     void main()
     {
          int i=1;
          do
          {
               printf(“hello world\t”);
               i++;
          } while(i<21);
     }
```

Using for loop

```
// Write “Hello World”, 20 times on the screen using for loop.

     #include<stdio.h>
     void main()
     {
          int i;
          for(i=1;i<=100;i++)
          {
               printf(“hello world\t”);
          }
     }
```

Output:

```
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
Hello World     Hello World
```

Conclusion: I understood the concept of looping in C; and successfully displayed “Hello World”, 20 times on the screen.

Signature of the student

Expt. No. **3** **Date:**

Aim of the experiment: To display all even numbers from 1-100.

Apparatus required: Computer system with turbo C editor

Theory:

- We should find the even numbers from 1 to 100, and display it.
- We have 50 even numbers between 1 to 100.
- The logic to test whether a number is even or odd is same for each number. Hence this will be repeated for 1 to 100.
- Since there is repetition, we will use the concept of loop here.

Program:

Using while loop

```
#include<stdio.h>
void main()
{
        int i=1;
        while(i<=100)
        {
                if(i%2==0)
                        printf("%d\t",i);
                i++;
        }
}
```

Using do-while loop

```
#include<stdio.h>
void main()
{
        int i=1;
        do
        {
                if(i%2==0)
                        printf("%d\t",i);
                i++;
        }while(i<=100);
}
```

Using for loop

```
#include<stdio.h>
void main()
{
        int i;
        for(i=1;i<=100;i++)
        {
                if(i%2==0)
                        printf("%d\t",i);
        }
}
```

Output:

```
2       4       6       8       10      12... 98      100
```

Conclusion: I have successfully displayed 1 to 100, numbers on the screen using loop.

Signature of the student

Expt. No. **4** **Date:**

Aim of the experiment: To perform addition of numbers from 1 to 100.

Apparatus required: Computer system with turbo C editor

Theory:

- We should find the sum of numbers from 1 to 100, and display it.
  `1 + 2 + 3 + 4 +...+ 99 + 100 = 5050`
- In LHS, the number is a continous series, so we can use loop concept of C to write the program.
- The number starts with 1 and ends at 100. Hence we can consider a variable ‘i’, initialize i=1, and condition will be, i<=100.
- The next number is one more than the previous number. So, i=i+1 or i++.
- Finally we should display the sum.

Program:

Using while loop

```
#include<stdio.h>
void main()
{
        int i=1, sum=0;
        while(i<=100)
        {
                sum=sum+i;
                i++;
                printf(“%d+”,i)
        }
        printf(“\b=%d”,sum);
}
```

Using do-while loop

```
#include<stdio.h>
void main()
{
        int i=1, sum=0;
        do
        {
                sum=sum+i;
                i++;
                printf(“%d+”,i)
        }while(i<=100);
        printf(“\b=%d”,sum);
}
```

Using for loop

```
     #include<stdio.h>
     void main()
     {
          int i, sum=0;
          for(i=1;i<=100;i++)
          {
               sum=sum+i;
               printf(“%d+”,i)
          }
          printf(“\b=%d”,sum);
     }
```

Output:

```
1 + 2 + 3 + 4 +...+ 99 + 100 = 5050
```

Conclusion: I have successfully calculated the sum of numbers from 1 to 100 using loop and displayed it on the screen.

Signature of the student

Expt. No. **5** **Date:**

Aim of the experiment: To find smallest number from array elements.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- To find the smallest number from an array of ‘n’ elements, we will apply the following logic.
    - Consider a variable smallest, with same data type as the array.
    - Assign the first element of the array to smallest.
      Smallest = array[0]
    - Compare ‘smallest’ with the elements of array from second element onwards.
      From i=1 to n, do the following
    - If (smallest > array[n]), then smallest=array[n].
    - Otherwise do nothing
    - Finally print the value of smallest.

Program:

Using for loop

```c
#include<stdio.h>
#define n 5
void main()
{
	int arr[n],num,i,smallest;
	printf(“How many numbers do you want in the array?”);
	scanf(“%d”,&num);
	printf(“Enter the elements of the array:”);
	for(i=0,i<num;i++)
		scanf(“%d”,&arr[i]);
		smallest=arr[0];
	for(i=1;i<num;i++)
	{
		if(smallest>arr[i])
			smallest=arr[i];
	}

	printf(“The smallest number is=%d”,smallest);
}
```

Output:

```
How many numbers do you want in the array?
5
Enter the elements of the array:
23
53
26
45
65
The smallest number is= 23
```

Conclusion: I have successfully executed the C program and found the smallest number from an array.

Signature of the student

Expt. No. **6** **Date:**

Aim of the experiment: To find largest number from array elements.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- To find the largest number from an array of ‘n’ elements, we will apply the following logic.
    - Consider a variable largest, with same data type as the array.
    - Assign the first element of the array to smallest.
      largest = array[0]
    - Compare ‘largest’ with the elements of array from second element onwards.
      From i=1 to n, do the following
    - If (largest < array[n]), then largest=array[n].
    - Otherwise do nothing
    - Finally print the value of smallest.

Program:

Using for loop

```
#include<stdio.h>
#define n 5
void main()
{
        int arr[n],num,i,largest;
        printf(“How many numbers do you want in the array?”);
        scanf(“%d”,&num);
        printf(“Enter the elements of the array:”);
        for(i=0,i<num;i++)
                scanf(“%d”,&arr[i]);
                largest=arr[0];
        for(i=1;i<num;i++)
        {
                if(largest<arr[i])
                        largest=arr[i];
        }

        printf(“The largest number is=%d”,largest);
}
```

Output:

```
How many numbers do you want in the array?
5
Enter the elements of the array:
23
53
26
45
65
The largest number is= 65
```

Conclusion: I have successfully executed the C program and found the smallest number from an array.

Signature of the student

Expt. No. **7** **Date:**

Aim of the experiment: To sort array elements in ascending order.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- To sort the elements of an array of ‘n’ elements in ascending order, we will apply the following logic.
    - Enter the elements of the array.
    - Compare one element of the array with its subsequent elements and keep the smallest number in the beginning of the array.
    - Repeat this process till the end of the array.

Program:

Using for loop

```
#include<stdio.h>
#define n 5
void main()
{
        int arr[n],num,i,j,temp;
        printf(“How many numbers do you want in the array?”);
        scanf(“%d”,&num);
        printf(“Enter the elements of the array:”);
        for(i=0,i<num;i++)
                scanf(“%d”,&arr[i]);
        for(i=0;i<num;i++)
        {
                for(j=i+1;j<num;j++)
                {
                        if(arr[i]>arr[j])
                        {
                                temp=arr[i];
                                arr[i]=arr[j];
                                arr[j]=temp;
                        }
                }
        }
        printf(“The sorted sequence is = \n”);
        for(i=0;i<num;i++)
                printf("%d\t",arr[i]);

}
```

Output:

```
How many numbers do you want in the array?
5
Enter the elements of the array:
23
53
26
45
65
The sorted sequence is:
23 26 45 53 65
```

Conclusion: I have successfully executed the C program and sorted the array with 5 elements in ascending order.

Signature of the student

Expt. No. **8** **Date:**

Aim of the experiment: To sort array elements in descending order.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- To sort the elements of an array of ‘n’ elements in descending order, we will apply the following logic.
    - Enter the elements of the array.
    - Compare one element of the array with its subsequent elements and keep the largest number in the beginning of the array.
    - Repeat this process till the end of the array.

Program:

Using for loop

```
#include<stdio.h>
#define n 5
void main()
{
      int arr[n],num,i,j,temp;
      printf(“How many numbers do you want in the array?”);
      scanf(“%d”,&num);
      printf(“Enter the elements of the array:”);
      for(i=0,i<num;i++)
            scanf(“%d”,&arr[i]);
      for(i=0;i<num;i++)
      {
            for(j=i+1;j<num;j++)
            {
                  if(arr[i]<arr[j])
                  {
                        temp=arr[i];
                        arr[i]=arr[j];
                        arr[j]=temp;
                  }
            }
      }
      printf(“The sorted sequence is = \n”);
      for(i=0;i<num;i++)
            printf("%d\t",arr[i]);

}
```

Output:

```
How many numbers do you want in the array?
5
Enter the elements of the array:
23
53
26
45
65
The sorted sequence is:
65 53 45 26 23
```

Conclusion: I have successfully executed the C program and sorted the array with 5 elements in ascending order.

Signature of the student

Expt. No. **9** **Date:**

Aim of the experiment: To enter the elements of a 3x3 matrix and display it.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- Here we need a 2-D array, where one variable refers to the row and another variable refers to the column of the 2-D array.

Program:

```
/*
program to enter the elements of a 3x3 matrix and display
it.
*/
#include<stdio.h>
void main()
{
        int arr[3][3],i,j;
        //enter the elements into the matrix through Keyboard
        printf(“Enter the elements of the array:”);
        for(i=0,i<3;i++)
                for(j=0;j<3;j++)
                        scanf(“%d”,&arr[i][j]);
        //display the matrix
        printf(“Your matrix is:\n”);
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        printf(“%d\t”,arr[i][j]);
                }
                printf(“\n”);
        }
}
```

Output:

```
Enter the elements of the array:
1 2 3 4 5 6 7 8 9
Your matrix is:
1       2       3
4       5       6
7       8       9
```

Conclusion: I have entered the elements of a 3x3 matrix and displayed it. The program is successfully executed.

Signature of the student

Expt. No. **10** **Date:**

Aim of the experiment: To enter the elements of a matrix of user defined size and display it.

Apparatus required: Computer system with turbo C editor

Theory:

- An array is a sequential memory allocation of homogeneous data elements.
- Here we need a 2-D array, where one variable refers to the row and another variable refers to the column of the 2-D array.
- In this program we don’t know the number of rows and columns. The user will enter the number of row and column at run time. An array can’t be defined with a variable. So, in order to do that, we can add a macro line with a large number, the user can assign the number of row and column less than or equal to the number.

Program:

```
#include<stdio.h>
#define n 5
void main()
{
      int arr[n][n],p,q,num,i,j;
      //define the row and column
      printf(“How many row do you want in your array?”);
      scanf(“%d”,&p);
      printf(“How many column do you want in your array?”);
      scanf(“%d”,&q);
      //enter the elements into the matrix through Keyboard
      printf(“Enter the elements of the array:”);
      for(i=0,i<p;i++)
            for(j=0;j<q;j++)
                  scanf(“%d”,&arr[i][j]);
      //display the matrix
      printf(“Your matrix is:”);
      for(i=0;i<p;i++)
      {
            for(j=0;j<q;j++)
            {
                  printf(“%d\t”,arr[i][j]);
            }
            printf(“\n”);
      }
}
```

Output:

```
How many row do you want in your array? 2
How many column do you want in your array? 3
Enter the elements of the array:
1 2 3
4 5 6
Your matrix is:
1      2      3
4      5      6
```

Conclusion: I have successfully entered the elements of a matrix of user defined size and displayed it.

Signature of the student

Expt. No. **11** **Date:**

Aim of the experiment: To enter two 3x3 matrix and display their sum.

Apparatus required: Computer system with turbo C editor

Procedure:

- Three 2D array need to defined. Two of them are needed to enter two 3x3 input matrix, and another one array is needed to store the sum.
- The sum of the matrix is obtained by adding the elements of first and second matrix, which has occupied the same position.
  Example:
  sum[0][0]=matrix1[0][0]+matrix2[0][0]
  sum[0][1]=matrix1[0][1]+matrix2[0][1]

Program:

```
#include<stdio.h>
void main()
{
    int mat1[3][3],mat2[3][3],sum[3][3],i,j;
    //enter the elements into the matrix through Keyboard
    printf("Enter the elements of first matrix:");
    for(i=0,i<3;i++)
        for(j=0;j<3;j++)
            scanf("%d",&mat1[i][j]);
    printf("Enter the elements of second matrix:");
    for(i=0,i<3;i++)
        for(j=0;j<3;j++)
        {
            scanf("%d",&mat2[i][j]);
            sum[i][j]=0;
        }
    //find the sum
    for(i=0,i<3;i++)
        for(j=0;j<3;j++)
            sum[i][j]= mat1[i][j]+mat2[i][j];
    //display the sum
    printf("The sum is:\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d\t",sum[i][j]);
        }
        printf("\n");
    }
}
```

Output:

```
Enter the elements of first matrix:
1 2 3
4 1 6
2 1 3
Enter the elements of second matrix:
1 2 3
4 5 6
3 1 2
The sum is:
2     4     6
8     6     12
5     2     5
```

Conclusion: I have successfully completed the experiment of addition of two 3x3 matrix.

Signature of the student

Expt. No. **12** **Date:**

Aim of the experiment: To enter two 3x3 matrix and display their product.

Apparatus required: Computer system with turbo C editor

Theory:

- Three 2D array need to defined. Two of them are needed to enter two 3x3 input matrix, and another one array is needed to store the product.
- The product of the matrix is obtained by multiplying the elements of the row of first matrix with corresponding column of the second matrix.
  Example:
  mul[0][0]=mat1[0][0]*mat2[0][0]+mat1[0][1]*mat2[1][0]+mat1[0][2]*mat2[2][0]
  mul[0][1]=mat1[0][0]*mat2[0][1]+mat1[0][1]*mat2[1][1]+mat1[0][2]*mat2[2][1]
  mul[0][2]=mat1[0][0]*mat2[0][2]+mat1[0][1]*mat2[1][2]+mat1[0][2]*mat2[2][2]
  mul[1][0]=mat1[1][0]*mat2[0][0]+mat1[1][1]*mat2[1][0]+mat1[1][2]*mat2[2][0]

Procedure:

we can generalize these statements as,

```
mul[i][j]=0;
for(i=0;i<3;i++)
        for(j=0;j<3;j++)
                for(k=0;k<3;k++)
                        mul[i][j]=mul[i][j]+mat1[i][k]*mul[k][j];
```

Program:

```
#include<stdio.h>
void main()
{
      int mat1[3][3],mat2[3][3],mul[3][3],i,j,k;
      //enter the elements into the matrix through Keyboard
      printf(“Enter the elements of first matrix:”);
      for(i=0,i<3;i++)
            for(j=0;j<3;j++)
                  scanf(“%d”,&mat1[i][j]);
      printf(“Enter the elements of second matrix:”);
      for(i=0,i<3;i++)
            for(j=0;j<3;j++)
            {
                  scanf(“%d”,&mat2[i][j]);
                  mul[i][j]=0;
            }
      //find the product
      for(i=0,i<3;i++)
            for(j=0;j<3;j++)
                for(k=0;k<3;k++)
                    mul[i][j]= mul[i][j]+ mat1[i][k]+ mat2[k][j];
      //display the product
      printf(“The product of two matrix is:\n”);
      for(i=0;i<3;i++)
      {
            for(j=0;j<3;j++)
            {
                  printf(“%d\t”,mul[i][j]);
            }
            printf(“\n”);
      }
}
```

Output:

```
Enter the elements of first matrix:
1 0 0
0 1 0
0 0 1
Enter the elements of second matrix:
1 0 0
0 1 0
0 0 1
The product of two matrix is:
1 0 0
0 1 0
0 0 1
```

Conclusion: I have successfully completed the experiment of multiplication of two 3x3 matrix.

Signature of the student

Expt. No. **13** **Date:**

Aim of the experiment: To demonstrate output of standard library functions strlen (), strcpy (), strcat (), strcmp ().

Apparatus required: Computer system with turbo C editor

Theory:

- There are various library functions available in C.
- Here we study the string functions strlen (), strcpy (), strcat (), strcmp ().
- Strlen : this function displays the length of a given string.
- Strcpy : this function copy the string value of one variable into another variable.
- Strcat : this functions joins two given string and produce the output.
- Strcmp : this function compares two given string.

Procedure:

- To perform the experiment of string functions, we must include the headerfilestring.h into the program.

Program:

```
#include<stdio.h>
#include<string.h>
void main()
{
      char str1[5], str2[5], str3[10];
      int cmp;
      //enter the value of str1
      puts(“Enter the value of string1:”);
      gets(str1); //input the value of str1
      printf(“the length of string 1=%d”,strlen(str1));
      str2=strcpy(str1); //copy the value of str1 into str2
      puts(str2); //display the value of str2
      str3=strcat(“String3=The value of string2 is”,str2);
      puts(str3);
      cmp=strcmp(str3,str1);
      if(cmp>0)
            puts(“string3 is larger than string1”);
      else
            puts(“string1 is larger than string3”);
}
```

Output:

```
Enter the value of string1: UGMIT
the length of string 1= 5
UGMIT
String3=The value of string2 is UGMIT
string3 is larger than string1
```

Conclusion: I have successfully performed the experiment using some of the library functions of string.

Signature of the student