LAB ASSIGNMENT NO.3

FUNCTIONS

SHREYAS SAWANT

D2A-55

AIM: Programming using C language

THEORY:

Functions: A function is a self-contained block of program statements that performs a particular task. It is often defined as a section of a program performing a specific job. All C programs contain at least one function, called main() where execution starts. Returning from this function the program execution terminates and the returned value is treated as an indication of success or failure of program execution.

Function Definition: The collection of program statements in C that describes the specific task done by the function is called a function definition. It consists of the function header and a function body, which is a block of code enclosed in parentheses. The definition creates the actual function in memory. The general form of the function definition is as follows:

return data_type function name(data_type variable1, data_type variable2,……)

{/* Function Body */ }

The function header in this definition is return_data_type function name(data_type variable1, data_type variable2,……) and the portion of program code within the braces is the function body.

The technique used to pass data to a function is known as parameter passing. Data are passed to a function using one of the two techniques:

Pass by value or call by value: In call by value, a copy of the data is made and the copy is sent to the function. The copies of the value held by the arguments are passed by the function call. Since only copies of values held in the arguments are passed by the function call to the formal parameters of the called function, the value in the arguments remains unchanged.

Pass by reference or call by reference: The second technique, pass by reference, sends the address of the data rather than a copy. In this case, the called function can change the original data in the calling function. Unfortunately, C does not support pass by reference.

Recurison: A recursive function is one that calls itself directly or indirectly to solve a smaller version of its task until a final call which does not require a self-call.

FUNCTIONS

b) Write a program to find the sum of first n natural numbers.

Code:

```
#include <stdio.h>
int sum(int );
int main()
{   int n;
    printf("Enter n: ");
    scanf("%d",&n);
    printf("The sum of %d is %d",n,sum(n));
   return 0;
}
int sum(int x)
{ int s=0;
    for(int i=1;i<=x;i++)
    { s=s+i;}
    return s;
}
```

Output:

```
Enter n: 39
The sum of 39 is 780
```

RECURSION

b) Write a program to calculate factorial of a number.

Code:

```
#include <stdio.h>
int fact(int );
int main()
{   int n;
    printf("Enter n: ");
    scanf("%d",&n);
    printf("The factorial of %d is %d",n,fact(n));
   return 0;
}
int fact(int f)
{  if (f==0)
   return 1;
   else
   return f*fact(f-1);
}
```

Output:

```
Enter n: 10
The factorial of 10 is 3628800
```