**LAB # 9**

**OBJECT**

***Using Functions***

**THEORY**

## Function

Function is the basic building block of C language. Functions are those parts of program, which perform certain specific task. Different function can be used in order to fulfill different task.

There are four types of functions depending on the return type and arguments:

* Functions that take nothing as argument and return nothing.
* Functions that take arguments but return nothing.
* Functions that do not take arguments but return something.
* Functions that take arguments and return something.

## Use of Functions

There are many programs in which an operation is performed several times in the same way. This increases the amount of programming code, thus increasing the size of program. In order to remove this repetition, a function is defined which perform the operation whenever the main program need the operation. The code of the function is written once and utilized number of time, thus the size of the program is reduced.

### Structure of the function

There are three basic elements of a function. These elements are:

* Function Declaration
* Function Definition
* Function Call

### Function Declaration

A function is declared by name before it is used in the main program. The function declaration tells the compiler the name of the function, the data type of the function returns and the number and data type of the function arguments.

### Function Definition:

Function definition is the function itself. It is the programming code of the operation the function is performing. Function definition includes the function name, number and data type of the values the function is returning. Function definition tells the compiler that a function is being defined which can perform certain specific task.

### Function Call:

Function call defines that a function is called to perform the task it is designed to do. Function call makes the compiler to transfer control to the programming code in the function.

## Example-1

void sum(void);

void main(void)

{

printf(“\nProgram to print sum of two numbers\n”);

sum();

}

void sum(void)

{

int num1,num2,sum;

printf(“Enter 1stnumber:”);

scanf(“%d”,&num1);

printf(“Enter 2ndnumber:”);

scanf(“%d”,&num2);

sum=num1+num2;

printf(“Sum of %d+%d=%d”,num1,num2,sum);

}

**Function That Return Value**

## General form

Function may return any type of data. The general form of such C language function is as:

type function\_name (parameters list)

{

Statements;

}

type describe the return type of the function.

## Returning data type

A function can return any type of data except an array. If no data type specifier is present, then the C language compiler automatically assumes that the function is returning an integer value. In other words the default data type a function returns is an integer.

Before you can use a function that returns a non-integer value, you need to inform the compiler about its return type. If you do not, the compiler will generate code with the assumption that an integer is being returned.

## The return ( ) statement

If a function is desired to return value in a program then ‘return ( )’ statement is used to fulfill this task. The ‘return ( )’ statement has two purposes. First, executing it immediately transfers control from the function back to the calling program. Secondly, whatever is inside the parentheses following the ‘return’ is returned as a value to the calling program.

The ‘return’ statement does not need to be at the end of the function. It can occur anywhere in the function, as soon as it is encountered, control will be transferred to the calling program.

## Example-2

This program converts upper case character to lower case.

#include “stdio.h”

char getlc (void);

void main (void)

{

char chlc;

printf(“Type ‘a’ for first selection ‘b’ for”);

printf(“second:’);

chlc = getlc( );

switch (chlc)

{

case ‘a’:

printf(“\nYou typed an ‘a’.”);

break;

case ‘b’:

printf(“You typed a ‘b’.”);

break;

default:

printf(“\nYou chose a non exixtent”);

printf(“selection”);

}

}

char getlc(void)

{

char ch;

ch = getche( );

if ( ch > 64 && ch < 91 )

ch = ch + 32;

return(ch);

}

### Output

The result of the program will be:

Type ‘a’ for first selection, ‘b’ for second: a

You typed an ‘a’.

Type ‘a’ for first selection, ‘b’ for second: A

You typed an ‘a’.

Type ‘a’ for first selection, ‘b’ for second: c

You chose a non existent selection.

**Built-in Functions**

There are various header files which contain built-in functions. The programmer can include those header files in any program and then use the built-in function by just calling them.

**TASKS TO BE PERFORMED**

1. Using function, write a complete program that prints your name 10 times. The

Function can take no arguments and should not return any value.

# PROGRAM

#include<stdio.h>

Int loop();

Void main(){

Loop();

Getch();

}

Int loop(){

Char name;

Printf(“Enter name: “);

Scanf(“%s”,&name);

For(int x, x<=10,x++){

Printf(“%s”,name);

}

}

1. Write a function to calculate the factorial value of any integer entered through the keyboard.

# PROGRAM

#include <stdio.h>

int fact();

int main(){

int x;

printf("Enter any number to find factorial ");

scanf("%d",&x);

fact(x);

}

int fact(int n){

long int c, fact=1;

for(c=1;c<=n;c++){

fact=fact\*c;

printf("\nFactorial of number %d = %ld",c,fact);

}

getch();

}

1. Write a function which receives a float and an int from main ( ), finds the product of these two and returns the product which is printed through main ( ).

# PROGRAM

#include<stdio.h>

int pro(float, int);

int main(){

int x;

float y;

printf("Enter float and interger numbers:");

scanf("%f %d",&y,&x);

float ans=pro(y,x);

printf("%f",ans);

getch();

}

int pro(float f,int n){

float ans=f\*n;

return ans;

}