**Lab Report No. 05**



**Course Code:**

CSC141

**Submitted By:**

Haider Ali

**Registration No:**

FA21-BEE-053

**Submitted To:**

Dr. Muhammad Kaleem

**Lab # 05 Working with Custom Functions in C**

# In-Lab Task 1:

Write a C function 'int test\_prime(int);' that takes in a positive number as input and returns true

(1) if the input number is prime or false (0) if the input is not prime. Then using this function, write a C program that takes a number (N) as input from the user and prints out the first N prime numbers.

**Sol:-**

**Code:-**

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

/\* Write a C function 'int test\_prime(int);' that takes in a positive number as input and returns true

(1) if the input number is prime or false (0) if the input is not prime. Then using this function,

write a C program that takes a number (N) as input from the user and prints out the first N prime number\*/

int test\_prime(int); int c = 1;

int main()

{

int num,a;

printf("Enter any positive number: \n"); scanf("%d", &num);

if(num < 0)

{

printf("Invalid Input\n"); return 0;

}

else if( num == 1 || num == 0)

{

c = 0;

printf("%d",c);

}

else

{

a = test\_prime(num); printf("%d",a);

}

}

int test\_prime(int p)

{

for(int i = 2; i < p; i++)

{

if ( p % i == 0)

{

c = 0;

return 0;

}

}

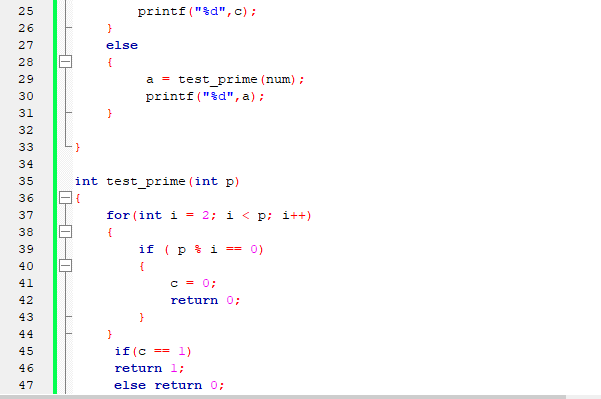
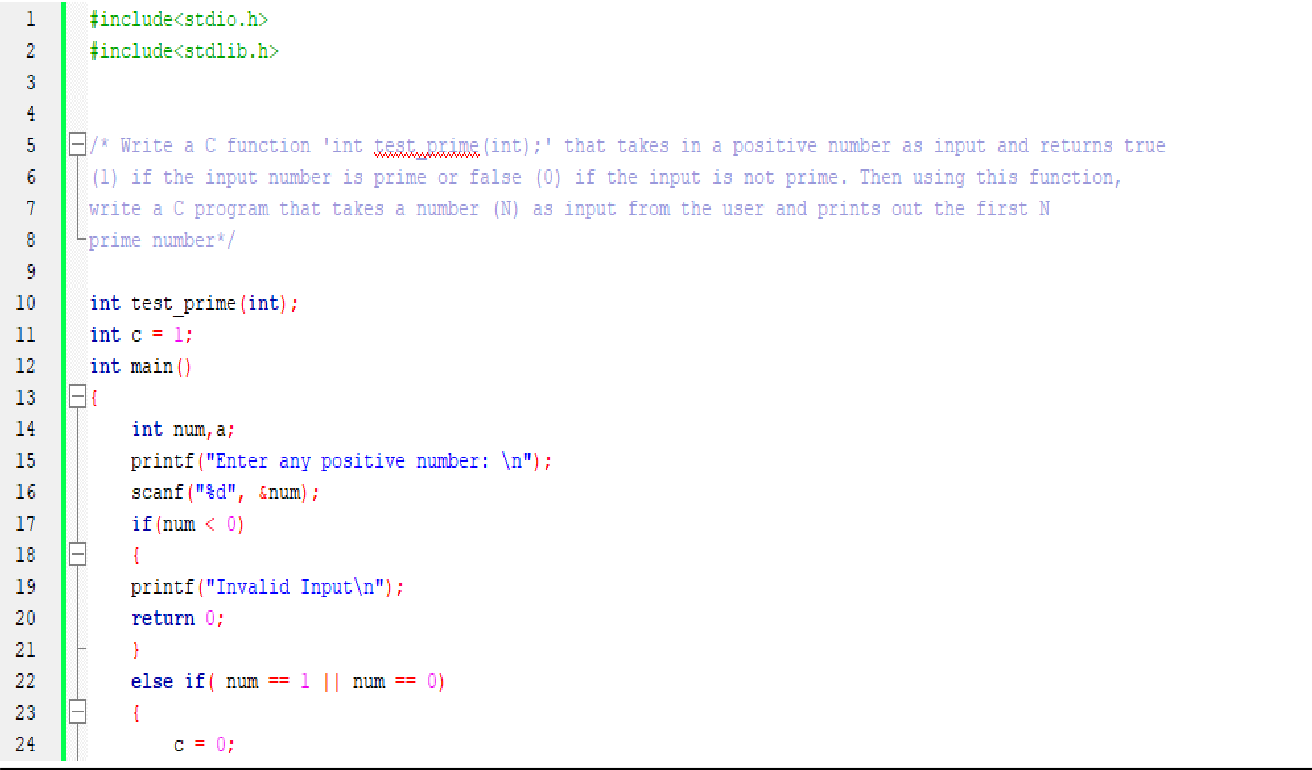
if(c == 1)

return 1;

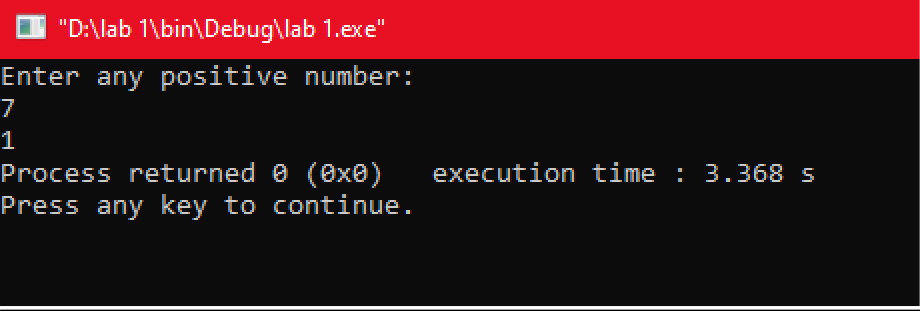
else return 0;

}

**Codeblocks:-**



**Output**



**Partb:-**

**Code:-**

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

/\* Write a C function 'int test\_prime(int);' that takes in a positive number as input and returns true

1. if the input number is prime or false (0) if the input is not prime. Then using this function, write a C program that takes a number (N) as input from the user and prints out the first N prime number\*/

int test\_prime(int, int); int main()

{

int num,a; int c = 1;

printf("Enter the number up to which you want to print prime numbers: \n"); scanf("%d", &num);

if(num == 1 || num == 0)

{

c = 0;

}

else

{

for(int i = 2; i <= num; i++)

{

a = test\_prime(i,c); printf("%d\n",a);

}

}

}

int test\_prime(int p,int c)

{

for(int i = 2; i <= p/2; i++)

{

if ( p % i == 0)

{

c = 0;

break;

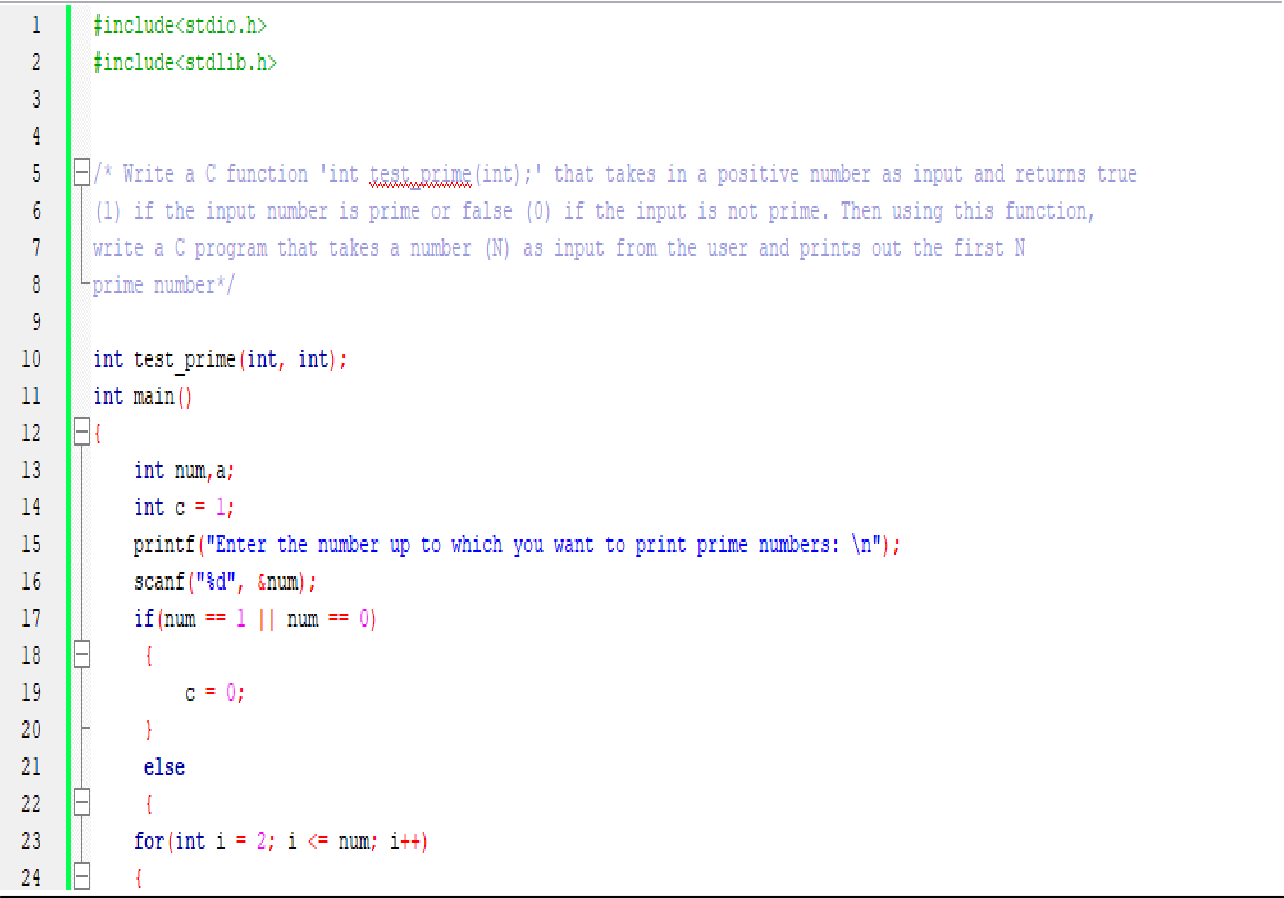
}

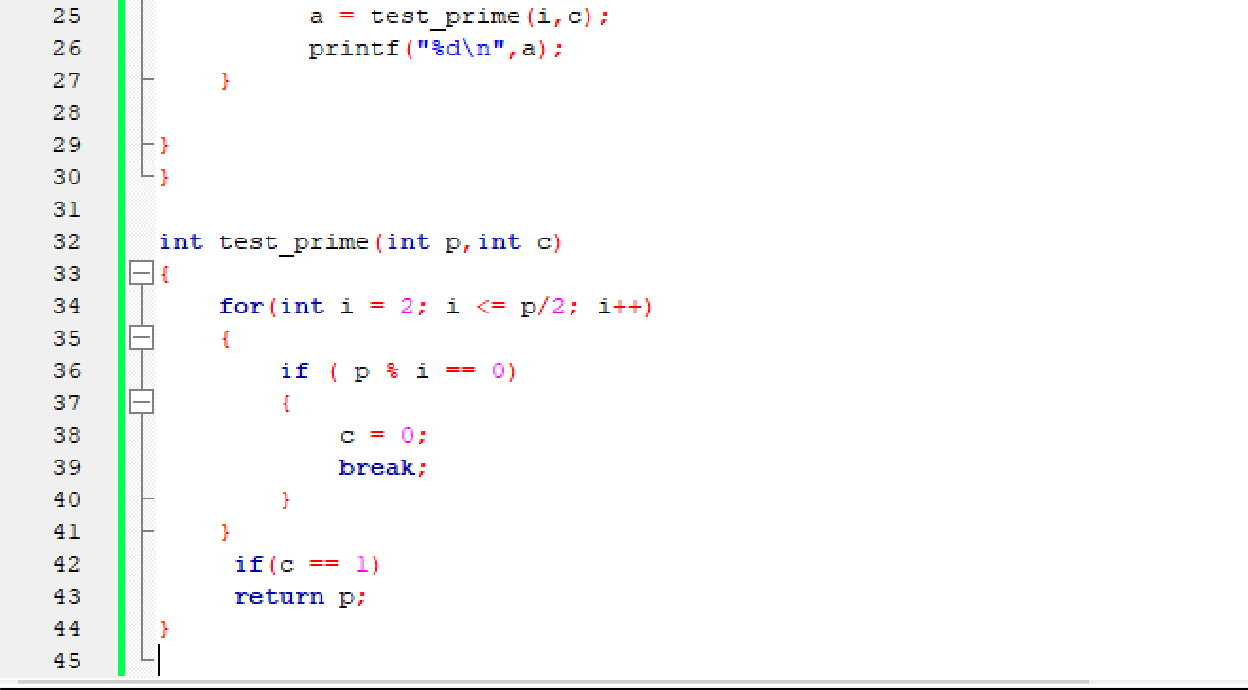
}

if(c == 1) return p;

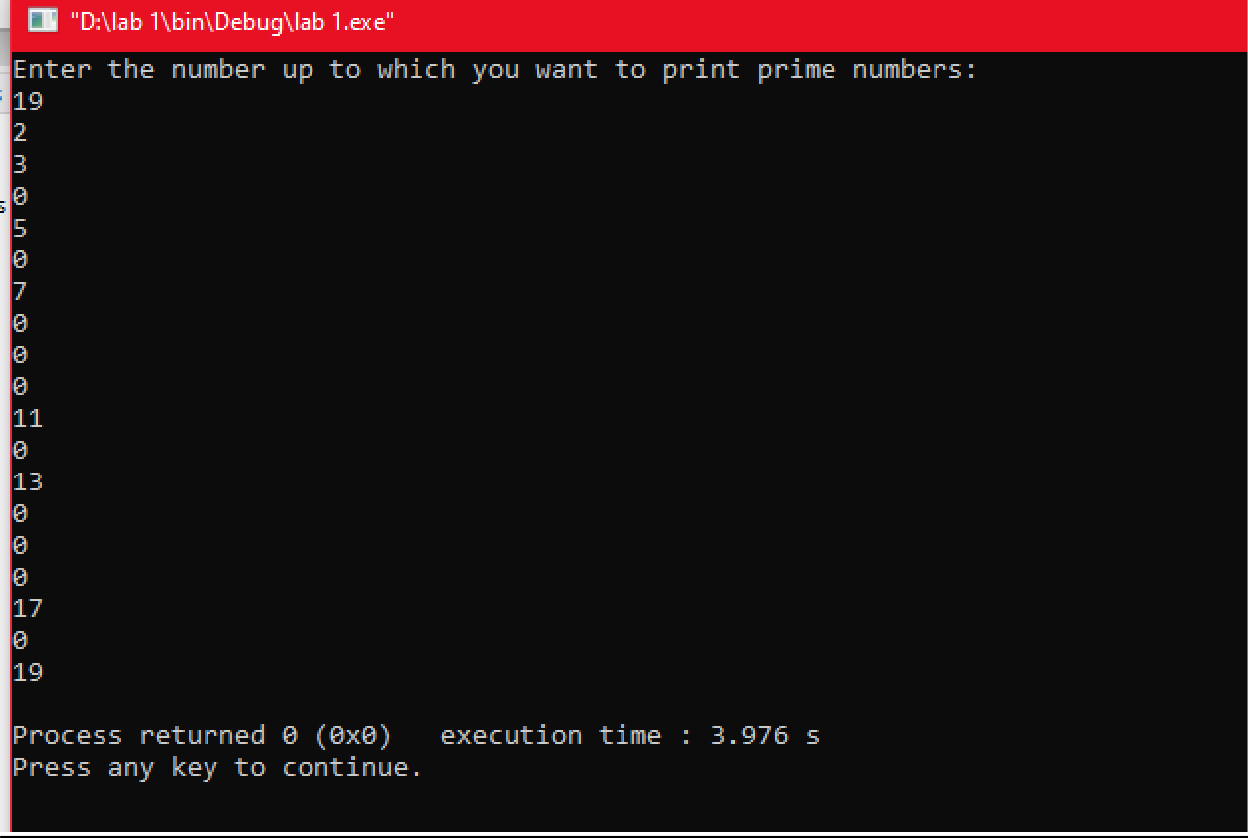
}

**Codeblocks:-**





**Output:-**



# In-Lab Task 2:

* 1. Write a C program that asks user to input a value for in degrees .It should then calculate the value of the mathematical function and print its value on screen. Write separate functions to implement .

**Hint**: include the ‘**math.h**’ library and use the following functions. Remember that these functions expect inputs to be in ***Radians***.

**double sin(double x); double cos(double x);**

* 1. Modify the above

cos ,

2

sin

. 2

program to calculate the value of

**Hint**: include the ‘**math.h**’ library and use the following function:

|  |  |
| --- | --- |
| **double sqrt(double x);** |  |
| cos | 2 |
| 1 | cos 2 |
| 0.5 | 2 |
| 1 |  |
| 2 |  |

**Solution:-**

**Code:-**

#include <stdio.h> #include <math.h>

double sin (double x); double cos (double x);

float fun1 (int x); float fun2 (int y);

int main ()

{

int angle; double f1, f2, y;

printf("Enter Angle: "); scanf("%d", &angle);

f1 = fun1(angle); f2 = fun2(angle);

y = f1 + f2;

printf("y = %f", y);

}

float fun1 (int x)

{

double sol1, solf1, angRad;

angRad = x\*(3.14/180);

sol1 = cos( (angRad/2)); solf1 = pow(sol1, 2);

return solf1;

}

float fun2 (int y)

{

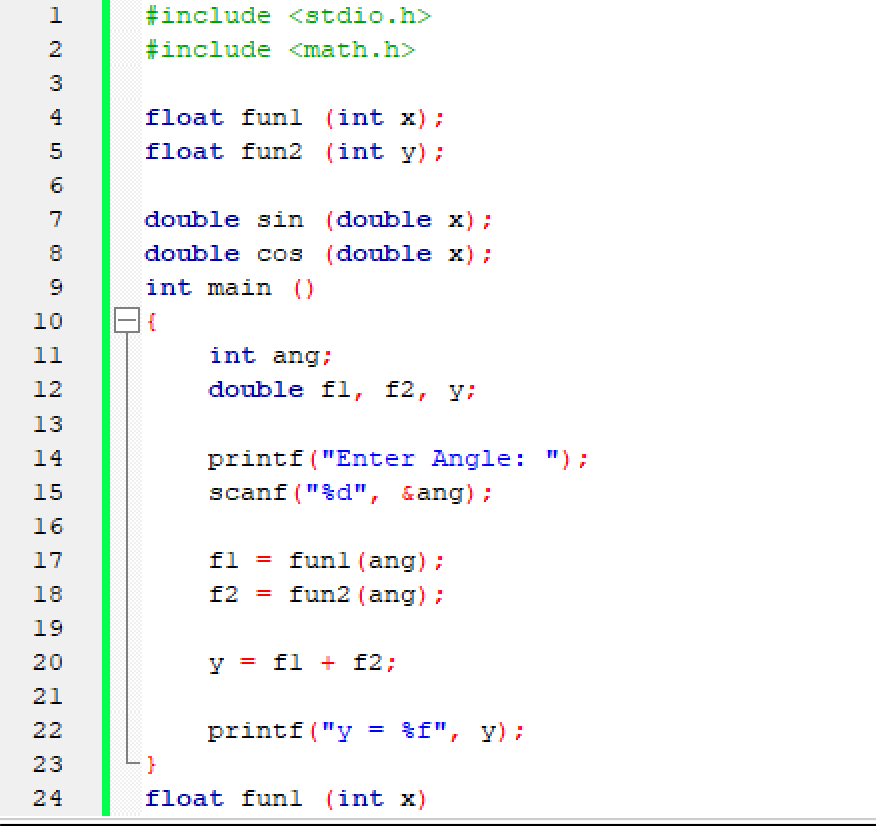
double sol2, solf2, angRad; angRad = y\*(3.14/180); sol2 = sin((angRad/2)); solf2 = pow(sol2, 2);

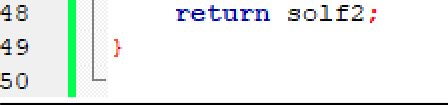
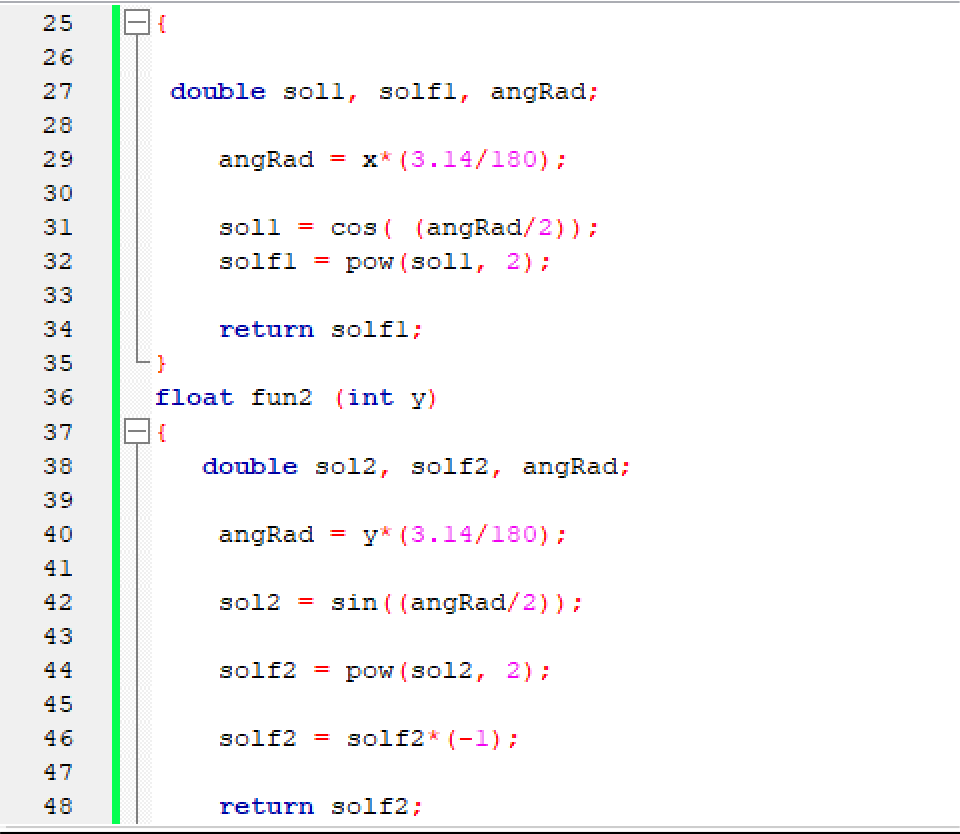
solf2 = solf2\*(-1);

return solf2;

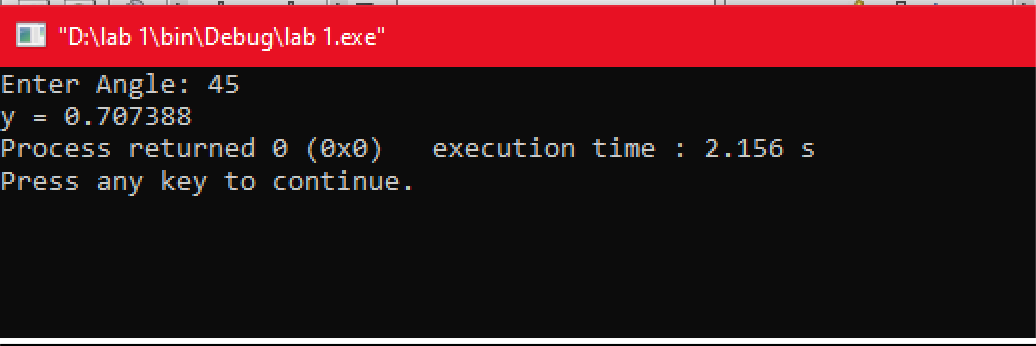
}

**Codeblocks:-**





**Output:-**



**Code:-**

#include <stdio.h> #include <math.h>

double sin (double x); double cos (double x);

float fun1 (int x); float fun2 (int y);

int main ()

{

int angle;

double f1, f2, f3, y;

printf("Enter Angle: "); scanf("%d", &angle);

f1 = fun1(angle); f2 = fun2(angle); f3 = 1/2;

y = f1 + f2 +f3;

printf("y = %f", y);

}

float fun1 (int x)

{

double sol1, solf1, angRad; angRad = x\*(3.14/180); sol1 = cos( (angRad/2)); solf1 = pow(sol1, 2);

return solf1;

}

float fun2 (int y)

{

double sol2, solf2, angRad; angRad = y\*(3.14/180);

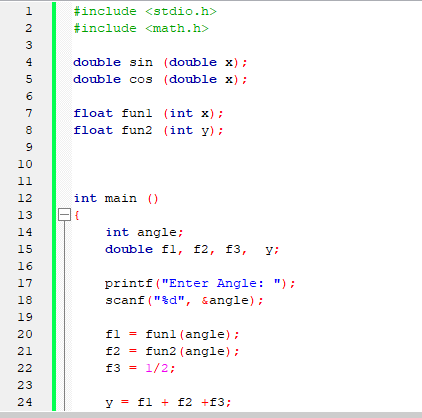
sol2 = (1+cos(2\*angRad))/2; sol2 = sqrt ((sol2));

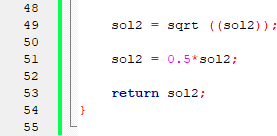
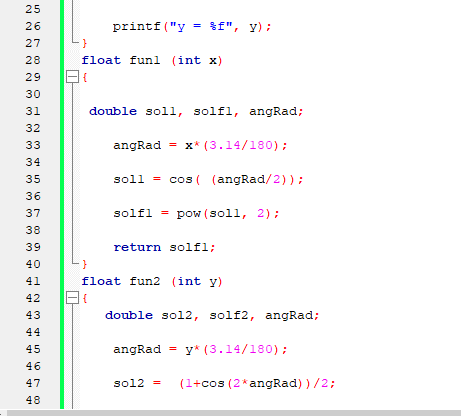
sol2 = 0.5\*sol2;

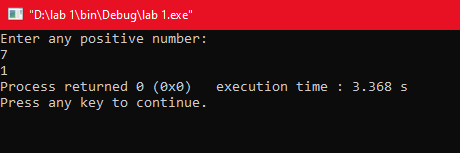
return sol2;

}

**Codeblocks:-**







# Post Lab Task:

Write a C program that takes two floating type inputs from the user and calculates their average, individual factorials, and a function . Use separate C functions to compute

function ‘f’. The program should prin t the results in the

the average, factorial and the ,

function.

**Solution:- Code:-**

**main**

/\* Write a C program that takes two floating type inputs from the user and calculates their

average,

individual factorials, and a function 𝑓ሺ𝑥, 𝑦ሻ ൌ ඥ𝑥ଶ ൅ 𝑦ଶ. Use separate C functions to compute

the average, factorial and the function ‘f’. The program should print the results in the main

function.

\*/

/// Start #include<stdio.h> #include<stdlib.h> #include<math.h>

float avg(float , float); float fact(float);

float fun(float, float);

int main()

{

float x; float y;

printf("Enter the two floating input: \n"); scanf("%f%f", &x,&y);

avg(x,y);

printf("Average of %.2f and %.2f is: %.2f\n",x,y, avg(x,y)); fact(x);

fact(y);

printf("Factorial of %f is: %f\n",x,fact(x)); printf("Factorial of %f is: %f\n",y,fact(y)); fun(x, y);

printf("The value of the function is: %f\n",fun(x,y));

}

float fun(float x, float y)

{

float function;

function = sqrt(x\*x + y\*y); return function;

}

float fact(float y){

float i; float f=1.0;

for(i=1.0;i<=y;i++) f=f\*i;

return f;

}

float avg(float a, float b)

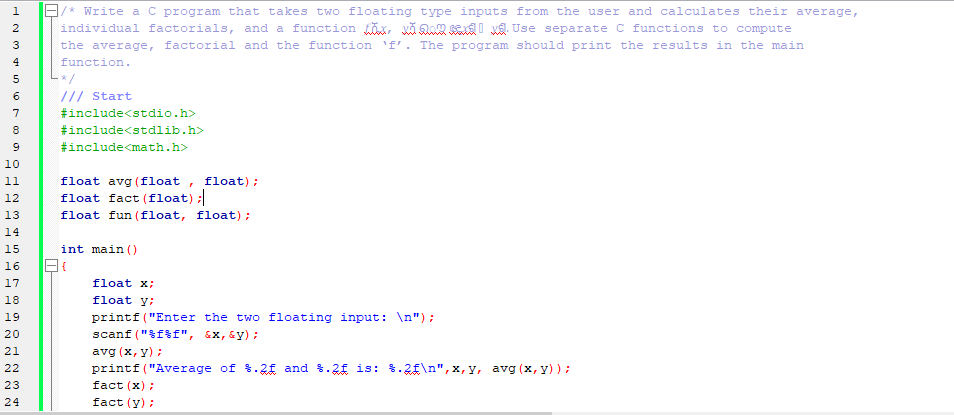
{

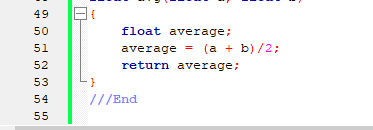
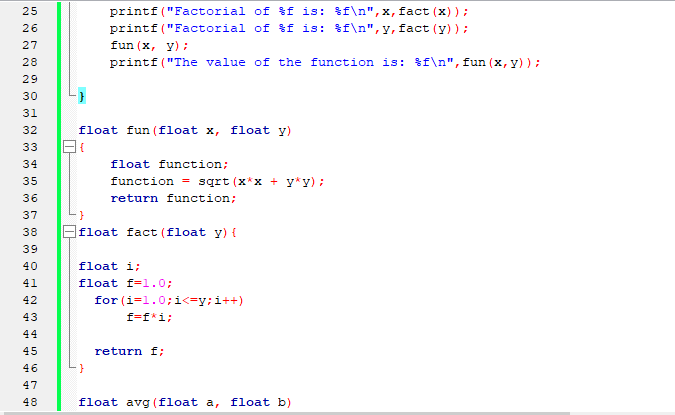
float average; average = (a + b)/2; return average;

}

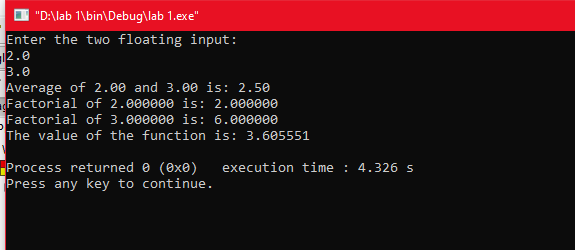
///End

**Codeblocks:-**





**Output:-**



**Critical Analysis:-**

Learnt to use custom functions in C which are actually how to declare, define and use functions.

Learnt to implement mathematical functions in programs using C functions which included use of sine and cosine function with square root functions which are defined in **math.h** library.

Functions improve reusability and help us code effectively.