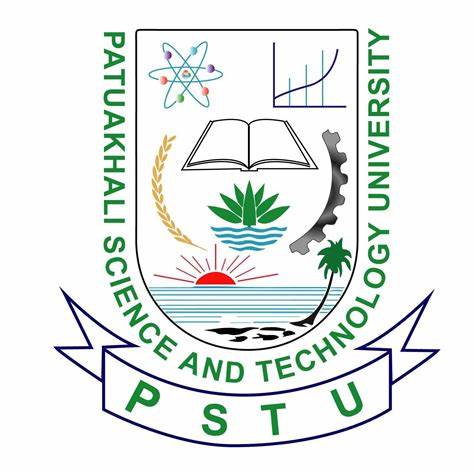
**PATUAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY**



# **Course Code: CIT-111/112**

**SUBMITTED TO:**  
 **Prof.****Md. Mahbubur Rahman**

### **Department of Computer Science and Information Technology Faculty of Computer Science and Engineering**

**SUBMITTED BY:**

Name: **Nazmus Sakib**

ID: **2102066**, Registration No: **10193**

**Faculty of Computer Science and Engineering**

Date of submission: **18th August,2023.**

# Assignment**: Basic Code-08: Function and Pointer + W3 resources-**

# **Function, Pointer, Recursion**

1. Write a program with multiple functions that do not communicate any data between them.

void printline (void);

void value (void);

main()

{

printline();

value();

printline();

}

void printline(void)

{

int i ;

for(i = 1; i <= 35; i++)

printf("%c",'-');

printf("\n");

}

void value(void)

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int year, period;

float inrate, sum, principal;

printf("Principal amount?");

scanf("%f", &principal);

printf("Interest rate?");

scanf("%f", &inrate);

printf("Period?");

scanf("%d", &period);

sum = principal;

year = 1;

while(year <= period)

{

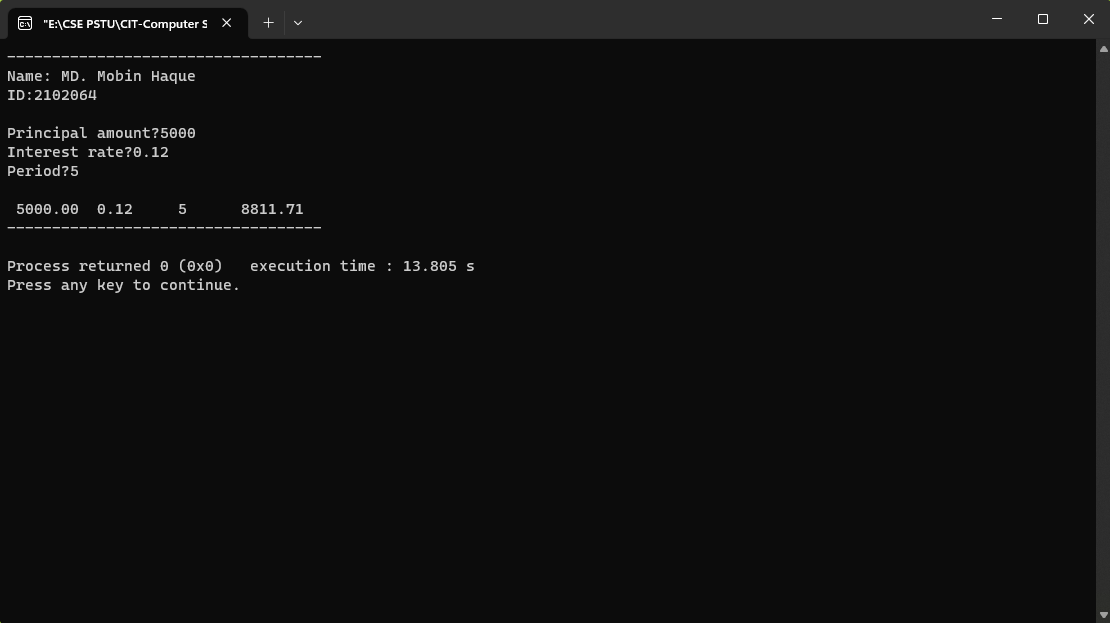
sum = sum \*(1 + inrate);

year = year + 1;

}

printf("\n%8.2f %5.2f %5d %12.2f\n", principal, inrate, period, sum);

}



1. Modify the program of Example 9.1 to include the arguments in the function calls.

void printline (char c);

void value (float, float, int);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

float principal, inrate;

int period;

printf("Enter principal amount, interest");

printf(" rate, and period \n");

scanf("%f %f %d",&principal, &inrate, &period);

printline('Z');

value(principal,inrate,period);

printline('C');

}

void printline(char ch)

{

int i ;

for(i = 1; i <= 52; i++)

printf("%c", ch);

printf("\n");

}

void value(float p, float r, int n)

{

int year ;

float sum ;

sum = p ;

year = 1;

while(year <= n)

{

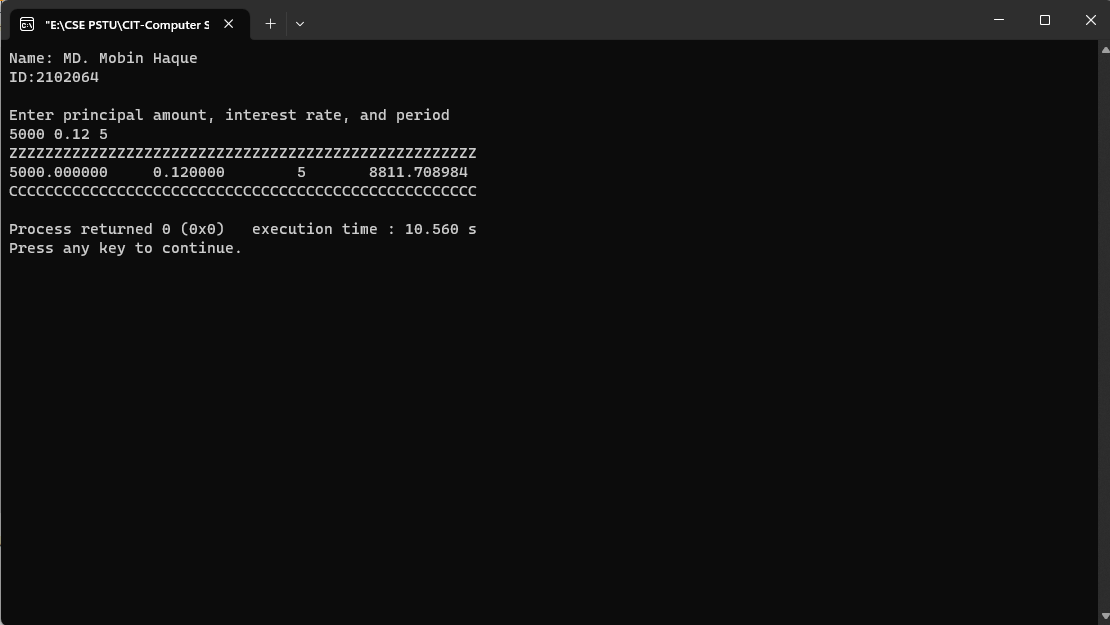
sum = sum \* (1 + r);

year = year + 1;

}

printf("%f\t%f\t%d\t%f\n",p,r,n,sum);

}



1. In the program presented in Fig. 9.7 modify the function **value,** to return the final amount calculated to the **main**, which will display the required output at the terminal. Also extend the versatility of the function **printline** by having it to take the length of the line as an argument.

void printline (char ch, int len);

value (float, float, int);

main( )

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

float principal, inrate, amount;

int period;

printf(“Enter principal amount, interest”);

printf(“rate, and period\n”);

scanf(%f %f %d”, &principal, &inrate, &period);

printline (‘\*’ , 52);

amount = value (principal, inrate, period);

printf(“\n%f\t%f\t%d\t%f\n\n”,principal,

inrate,period,amount);

printline(‘=’,52);

}

void printline(char ch, int len)

{

int i;

for (i = 1; i <= len; i++) printf(“%c”, ch);

printf(“\n”);

}

value(float p, float r, int n) /\* default return type \*/

{

int year;

float sum;

sum = p; year = 1;

while(year <=n)

{

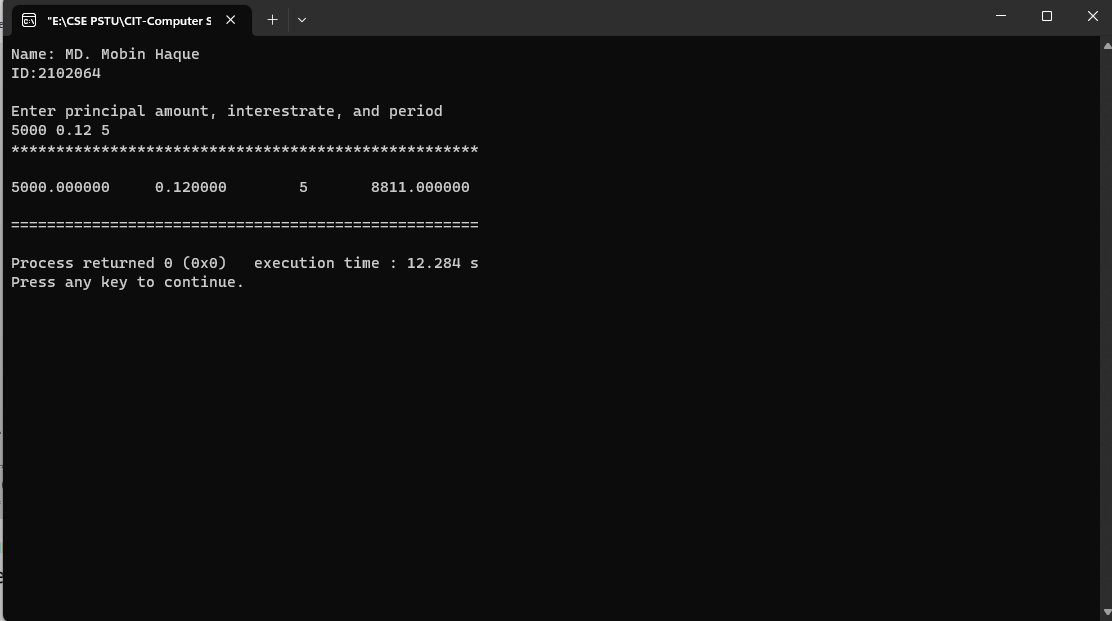
sum = sum \* (l + r);

year = year + 1;

}

return(sum); /\* returns int part of sum \*/

}



1. Write a function **power** that computes x raised to the power y for integers x and y and returns double-type value

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int x,y; /\*input data \*/

double power(int, int); /\* prototype declaration\*/

printf("Enter x,y:");

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

printf("%d to power %d is %f\n", x, y, power (x,y));

}

double power (int x, int y)

{

double p;

p = 1.0 ; /\* x to power zero \*/

if(y >= 0)

while(y--) /\* computes positive powers \*/

p \*= x;

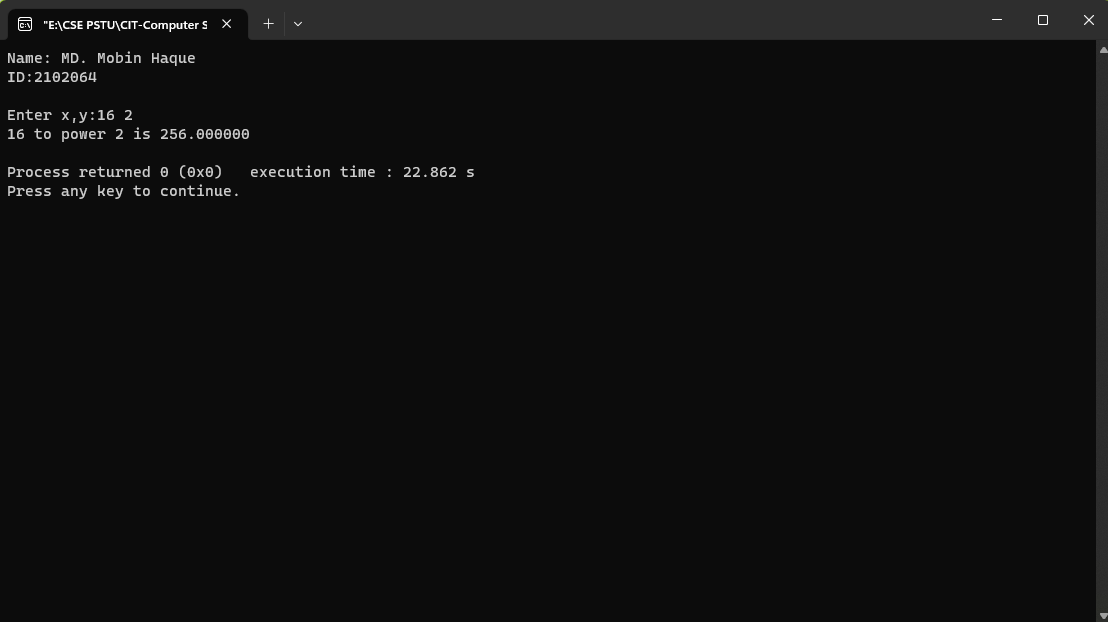
else

while (y++) /\* computes negative powers \*/

p /= x;

return(p);

}



1. A program shows how to write a C program (float x [], int n) that (n) that returns the position of the first minimum value among the first n elements of the given array x.

#include<stdio.h>

#include<stdlib.h>

int minpos(float[], int);

void main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int n;

float x[10] = {12.5, 3.0, 45.1, 8.2, 19.3, 10.0, 7.8, 23.7, 29.9, 5.2};

printf("Enter the value of n: ");

scanf("%d", &n);

if(n < 1 || n > 10)

{

printf("invalid value of n....Press any key to terminate the program..");

exit(0);

}

printf("Within the first %d elements of array, the first minimum value is stored at index %d", n, minpos(x,n));

}

int minpos(float a[], int N)

{

int i, index;

float min = 9999.99;

for(i = 0; i < N; i++)

{

if(a[i] < min)

{

min = a[1];

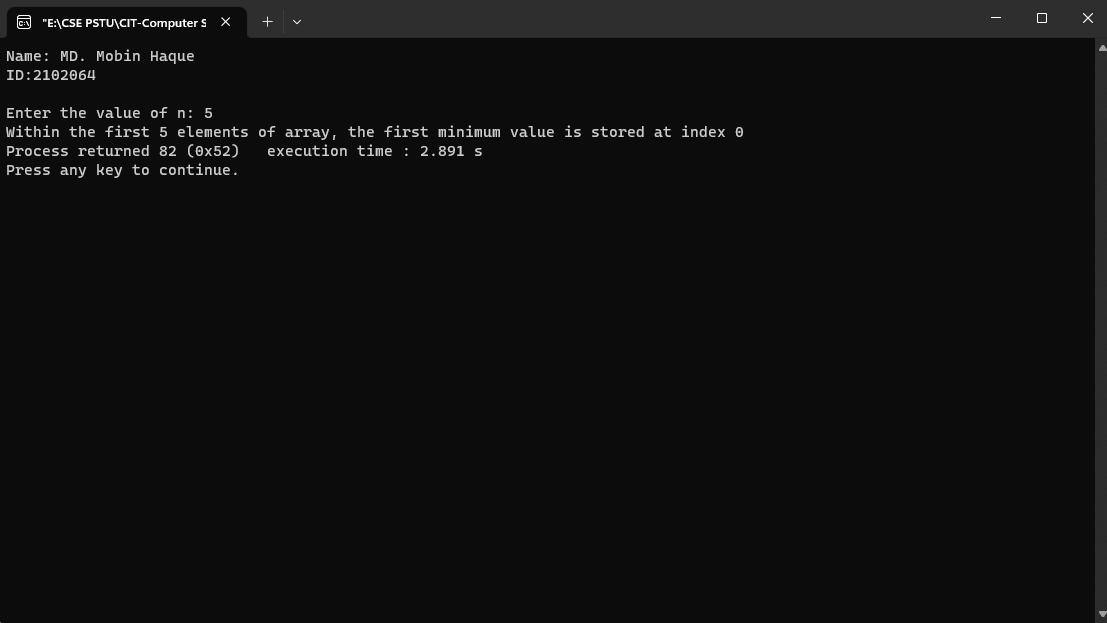
index = i;

}

}

return(index);

}



1. Write a program to calculate the standard deviation of an array of values. The array elements are read from the terminal. Use functions to calculate standard deviation and mean.



#include<math.h>

#define SIZE 5

float std\_dev(float a[], int n);

float mean (float a[], int n);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

float value[SIZE];

int I;

printf("Enter %d float values\n", SIZE);

for (i = 0 ; i < SIZE ; i++)

scanf("%f", &value[i]);

printf("Std.deviation is %f\n", std\_dev(value,SIZE));

}

float std\_dev(float a[], int n)

{

int i;

float x, sum = 0.0;

x = mean (a,n);

for(i = 0; i < n; i++)

sum += (x-a[i])\*(x-a[i]);

return(sqrt(sum/(float)n));

}

float mean(float a[],int n)

{

int i ;

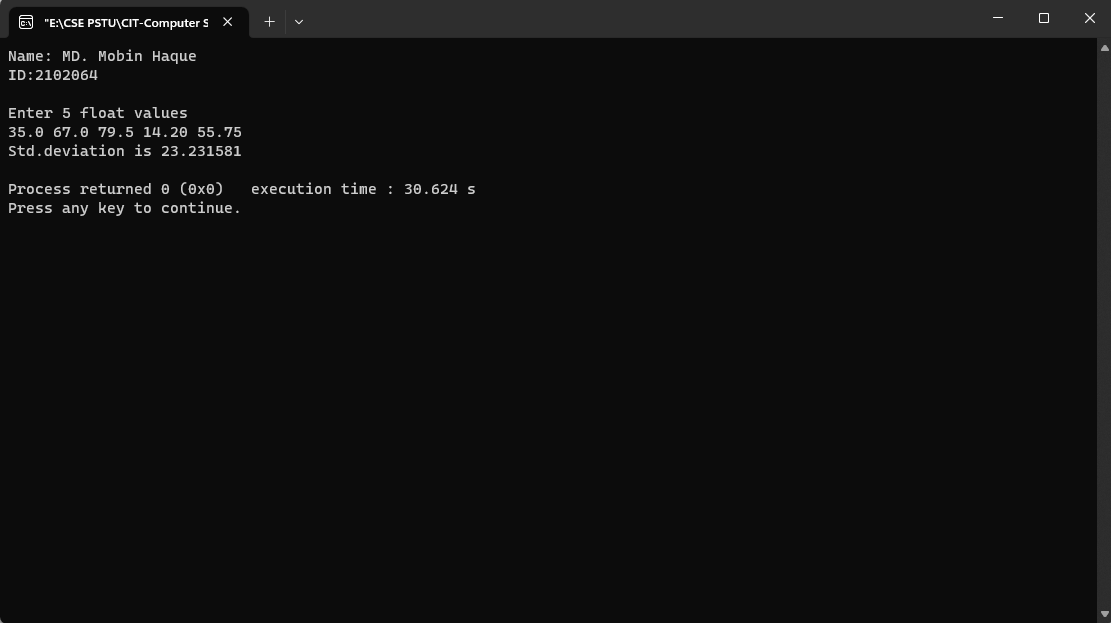
float sum = 0.0;

for(i = 0; i < n; i++)

sum = sum + a[i];

return(sum/(float)n);

}



1. Write a program that uses a function to sort an array of integers.

void sort(int m, int x[ ]);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int i;

int marks[5] = {40, 90, 73, 81, 35};

printf("Marks before sorting\n");

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

printf("%d ", marks[i]);

printf("\n\n");

sort (5, marks);

printf("Marks after sorting\n");

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

printf("%4d", marks[i]);

printf("\n");

}

void sort(int m, int x[ ])

{

int i, j, t;

for(i = 1; i <= m-1; i++)

for(j = 1; j <= m-i; j++)

if(x[j-1] >= x[j])

{

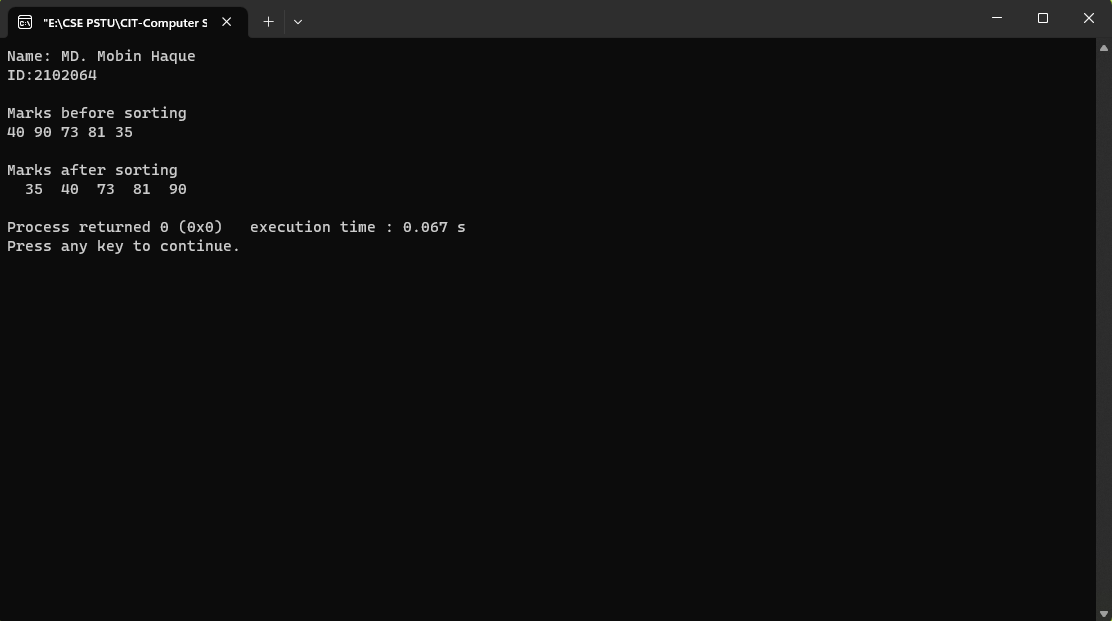
t = x[j-1];

x[j-1] = x[j];

x[j] = t;

}

}



1. Write a multifunction to illustrate how automatic variables work.

void function1(void);

void function2(void);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int m = 1000;

function2();

printf("%d\n",m); /\* Third output \*/

}

void function1(void)

{

int m = 10;

printf("%d\n",m); /\* First output \*/

}

void function2(void)

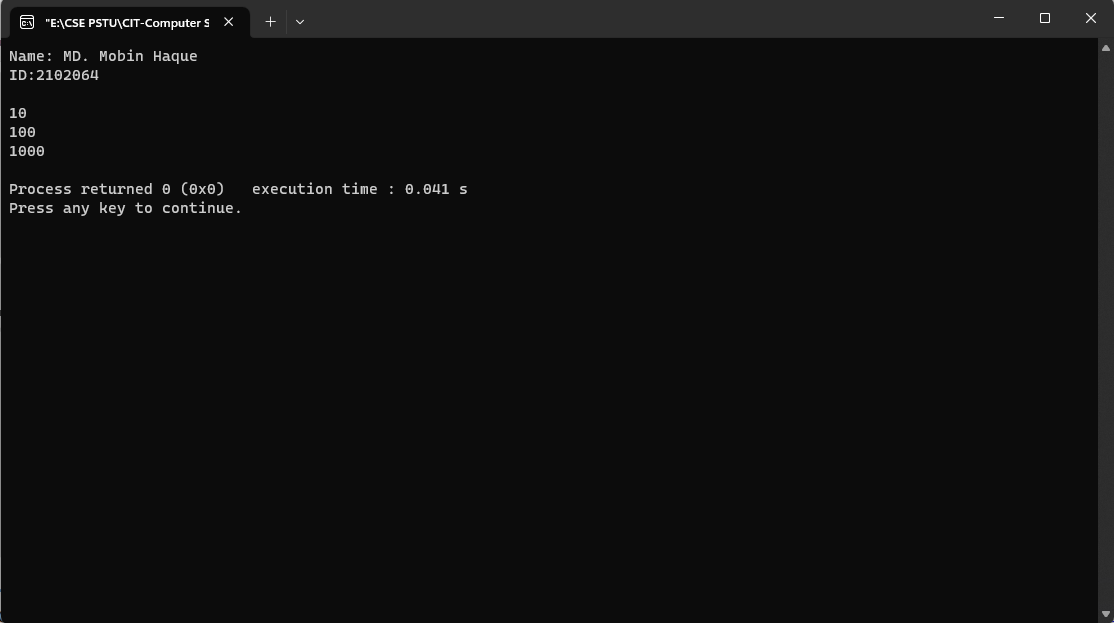
{

int m = 100;

function1();

printf("%d\n",m); /\* Second output \*/

}



1. Write a multifunction program to illustrate the properties of global variables.

int fun1(void);

int fun2(void);

int fun3(void);

int x; /\* global \*/

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

x = 10 ; /\* global x \*/

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

printf("x = %d\n", fun1());

printf("x = %d\n", fun2());

printf("x = %d\n", fun3());

}

fun1(void)

{

x = x + 10;

}

int fun2(void)

{

int x; /\* local \*/

x = 1;

return (x);

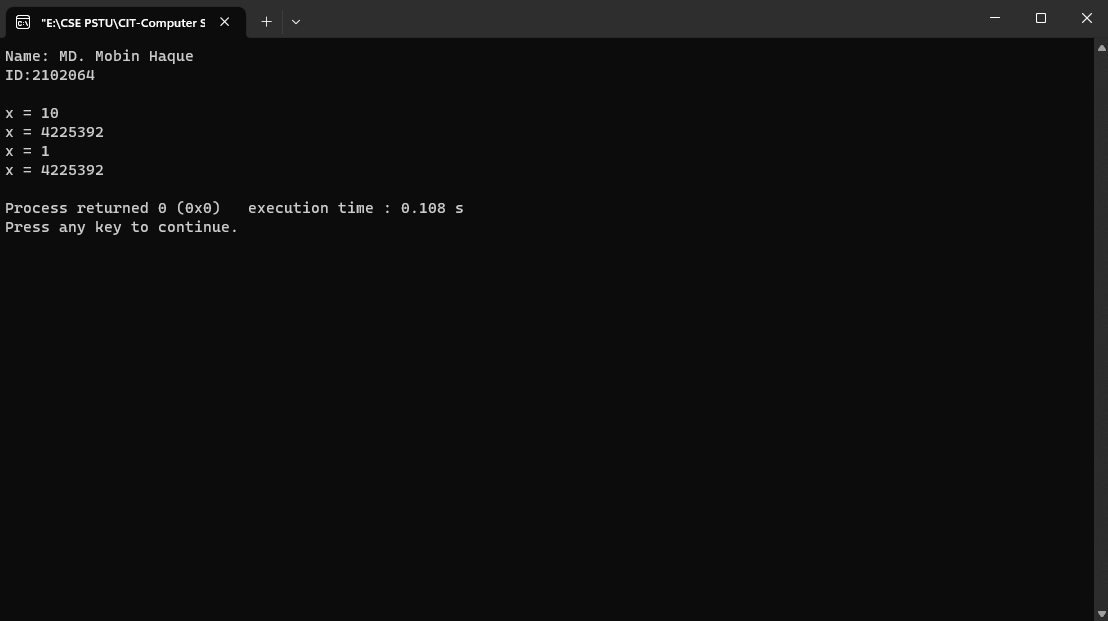
}

fun3(void)

{

x = x + 10; /\* global x \*/

}



1. Write a program to illustrate the properties of a static variable.

void stat(void);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int i;

for(i = 1; i <= 3; i++)

stat();

}

void stat(void)

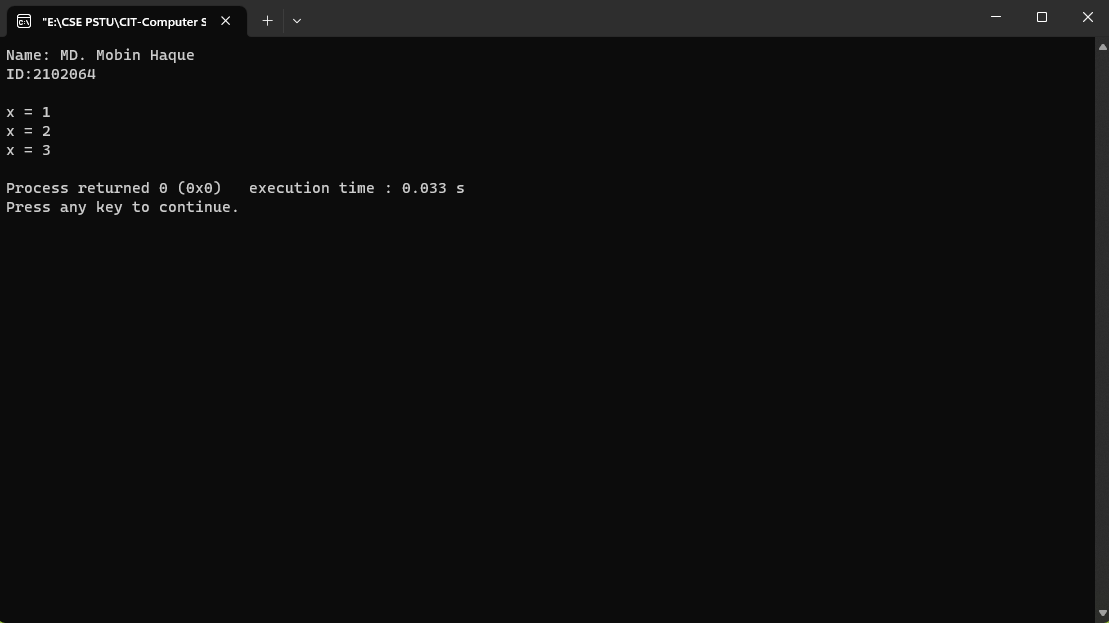
{

static int x = 0;

x = x + 1;

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

}



1. Write a program to print the address of a variable along with its value.

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

char a;

int x;

float p, q;

a = 'A';

x = 125;

p = 10.25, q = 18.76;

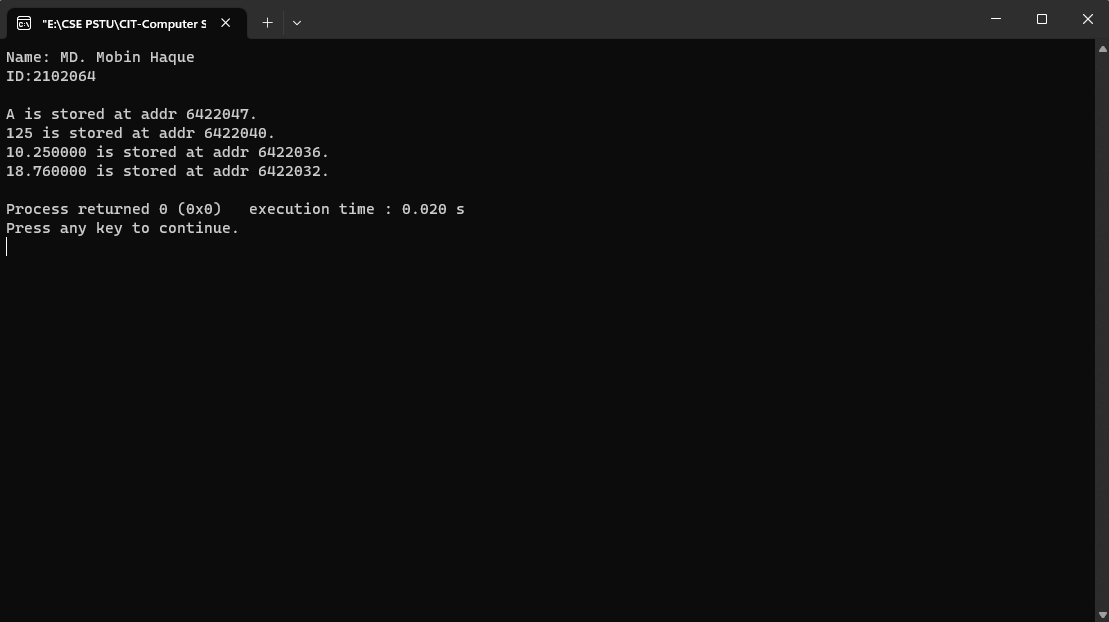
printf("%c is stored at addr %u.\n", a, &a);

printf("%d is stored at addr %u.\n", x, &x);

printf("%f is stored at addr %u.\n", p, &p);

printf("%f is stored at addr %u.\n", q, &q);

}



1. Write a program to illustrate the use of indirection operator '\*' to access the value pointed to by a printer.

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int x, y;

int \*ptr;

x = 10;

ptr = &x;

y = \*ptr;

printf("Value of x is %d\n\n", x);

printf("%d is stored at addr %u\n", x, &x);

printf("%d is stored at addr %u\n", \*&x, &x);

printf("%d is stored at addr %u\n", \*ptr, ptr);

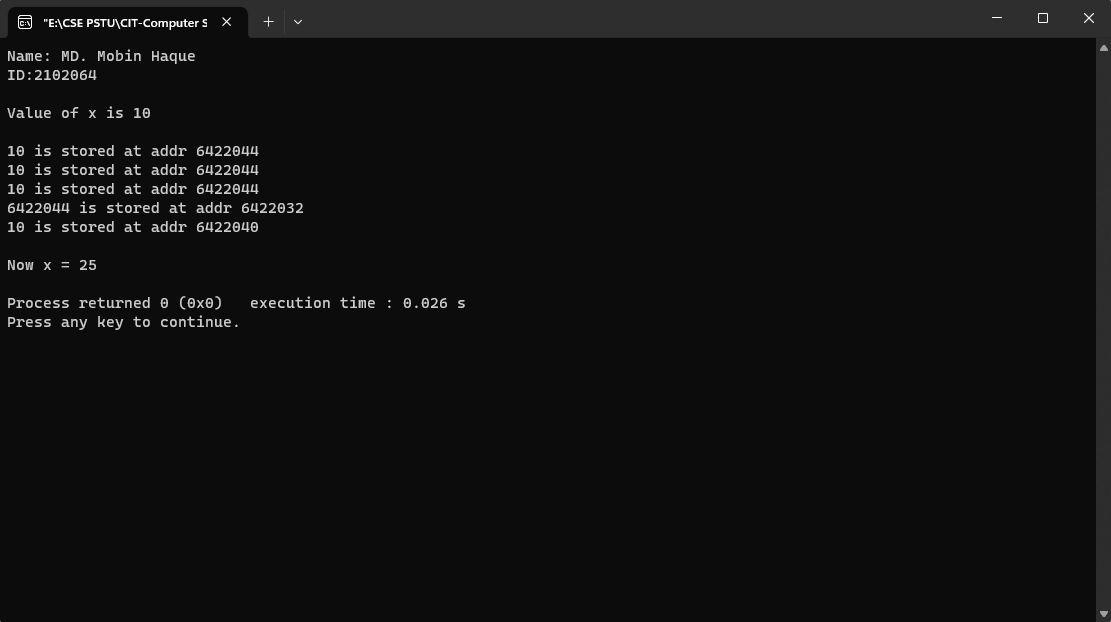
printf("%d is stored at addr %u\n", ptr, &ptr);

printf("%d is stored at addr %u\n", y, &y);

\*ptr = 25;

printf("\nNow x = %d\n", x);

}



1. Write a program to illustrate the use of pointers in arithmetic operations.

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int a, b, \*p1, \*p2, x, y, z;

a = 12;

b = 4;

p1 = &a;

p2 = &b;

x = \*p1 \* \*p2 - 6;

y = 4\* - \*p2 / \*p1 + 10;

printf("Address of a = %u\n", p1);

printf("Address of b = %u\n", p2);

printf("\n");

printf("a = %d, b = %d\n", a, b);

printf("x = %d, y = %d\n", x, y);

\*p2 = \*p2 + 3;

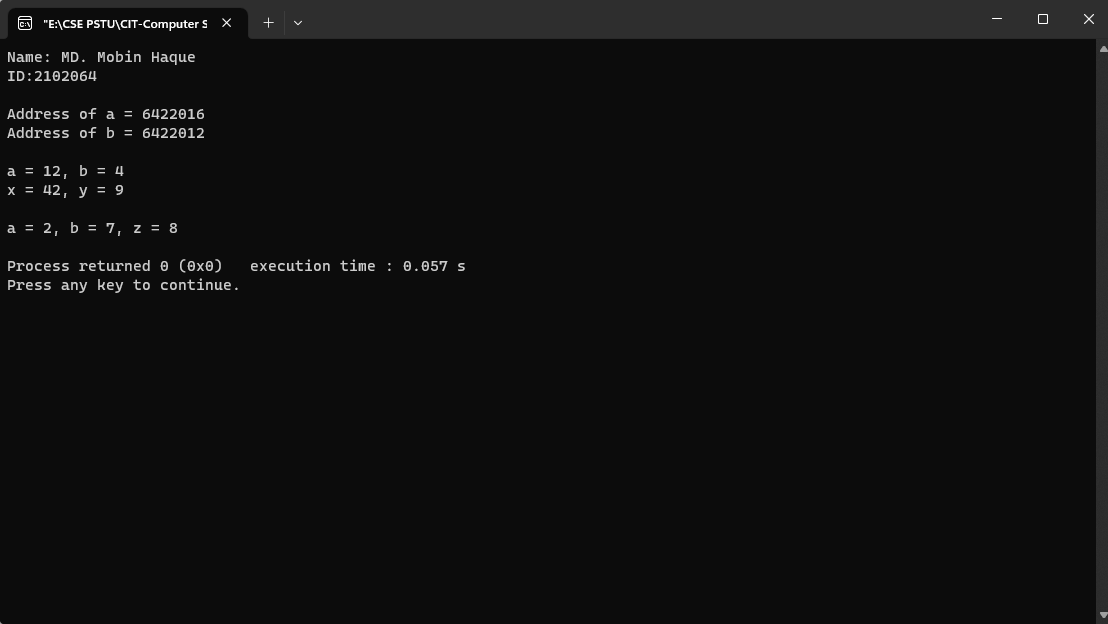
\*p1 = \*p2 - 5;

z = \*p1 \* \*p2 - 6;

printf("\na = %d, b = %d, ", a, b);

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

}



1. Write a program using pointers to compute the sum of all elements stored in an array.

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int \*p, sum, i;

int x[5] = {5,9,6,3,7};

i = 0;

p = x;

printf("Element Value Address\n\n");

while(i < 5)

{

printf(" x[%d] %d %u\n", i, \*p, p);

sum = sum + \*p;

i++, p++;

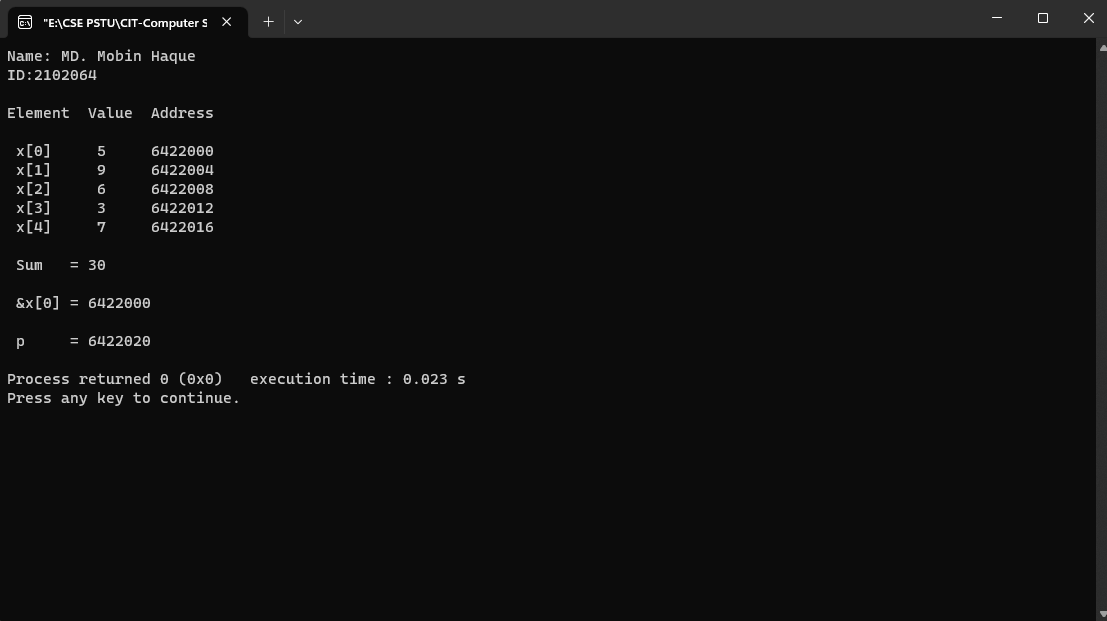
}

printf("\n Sum = %d\n", sum);

printf("\n &x[0] = %u\n", &x[0]);

printf("\n p = %u\n", p);

}



1. Write a program using pointers to determine the length of a character string.

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

char \*name;

int length;

char \*cptr = name;

name = "DELHI";

printf ("%s\n", name);

while(\*cptr != '\0')

{

printf("%c is stored at address %u\n", \*cptr, cptr);

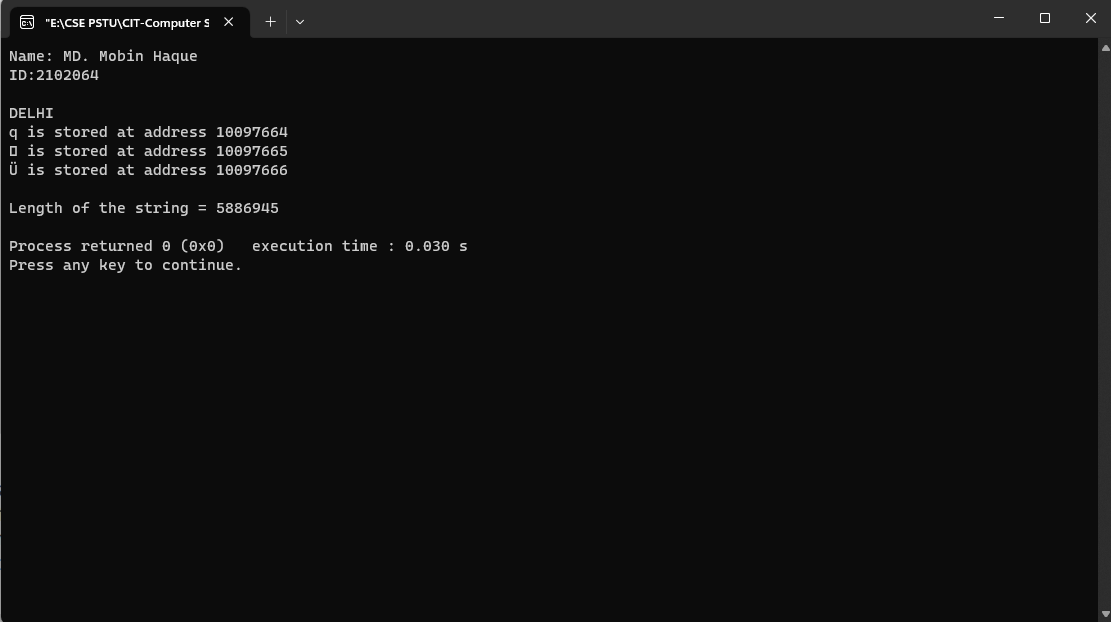
cptr++;

}

length = cptr - name;

printf("\nLength of the string = %d\n", length);

}



1. Write a function using pointers to exchange the values stored in two locations in the memory.

void exchange(int \*, int \*);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int x, y;

x = 100;

y = 200;

printf("Before exchange : x = %d y = %d\n\n", x, y);

exchange(&x,&y);

printf("After exchange : x = %d y = %d\n\n", x, y);

}

exchange (int \*a, int \*b)

{

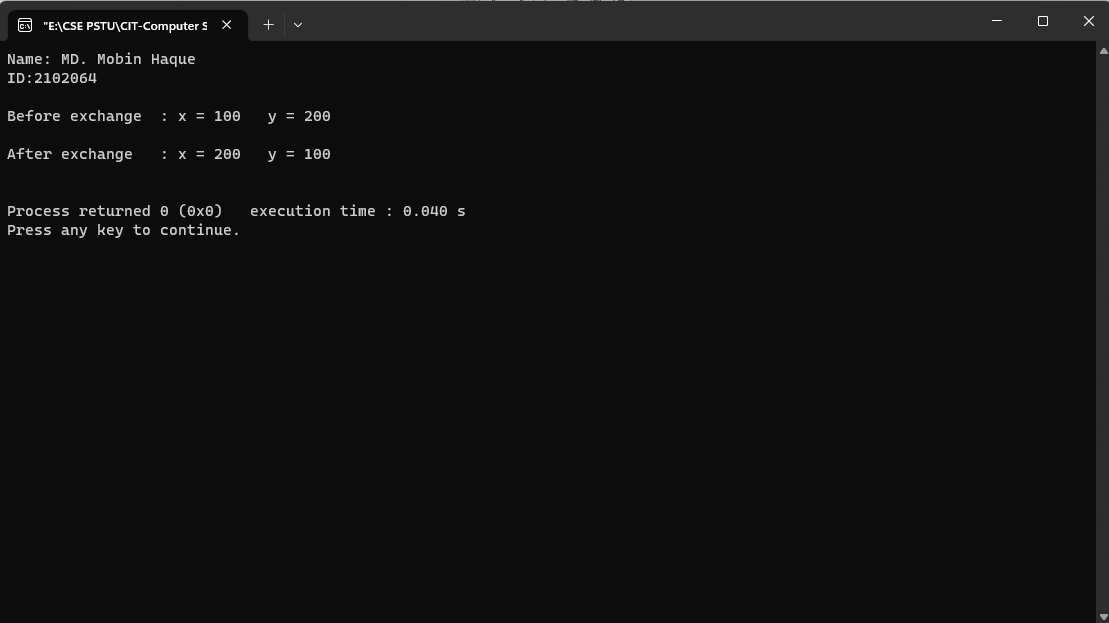
int t;

t = \*a;

\*a = \*b;

\*b = t;

}



1. A program shows how to calculate the sum of two numbers which are passed as arguments using the call by reference method.

#include<stdio.h>

void swap(int \*, int \*);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int x = 0;

int y = 20;

printf("\nValue of X and Y before swapping are X = %d and Y = %d", x, y);

swap(&x, &y);

printf("\n\nValue of X and Y after swapping are X = %d and Y = %d", x, y);

}

void swap(int \*p, int \*q)

{

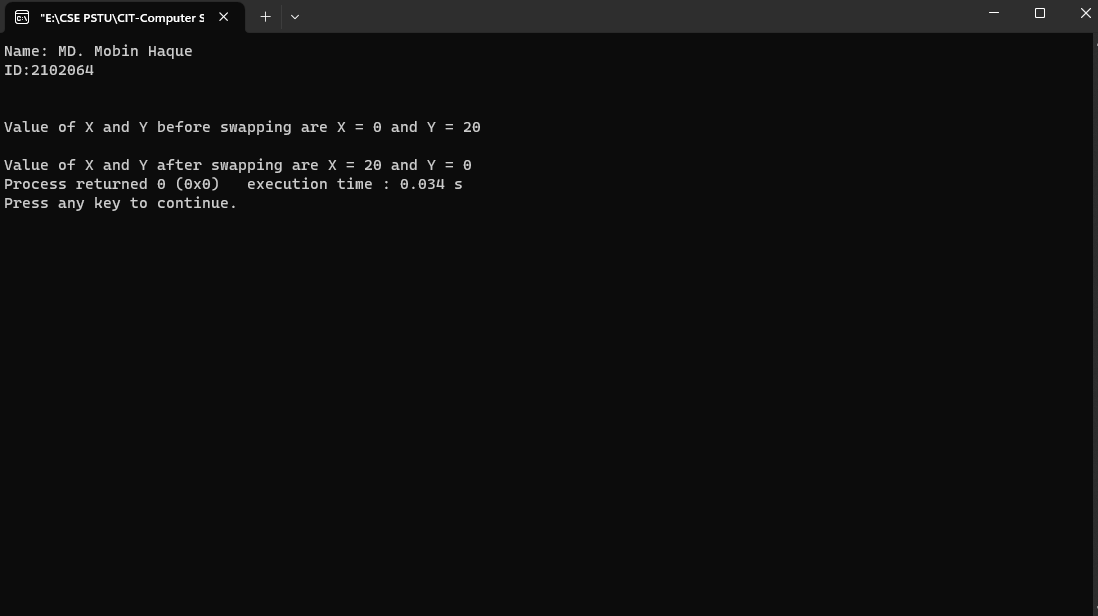
int r;

r = \*p;

\*p = \*q;

\*q = r;

}



1. Write a program that uses a function pointer as a function argument.

#include<math.h>

#define PI 3.1415926

double y(double);

double cos(double);

double table (double(\*f)(), double, double, double);

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

printf("Table of y(x) = 2\*x\*x-x+1\n\n");

table(y, 0.0, 2.0, 0.5);

printf("\nTable of cos(x)\n\n");

table(cos, 0.0, PI, 0.5);

}

double table(double(\*f)(),double min, double max, double step)

{ double a, value;

for(a = min; a <= max; a += step)

{

value = (\*f)(a);

printf("%5.2f %10.4f\n", a, value);

}

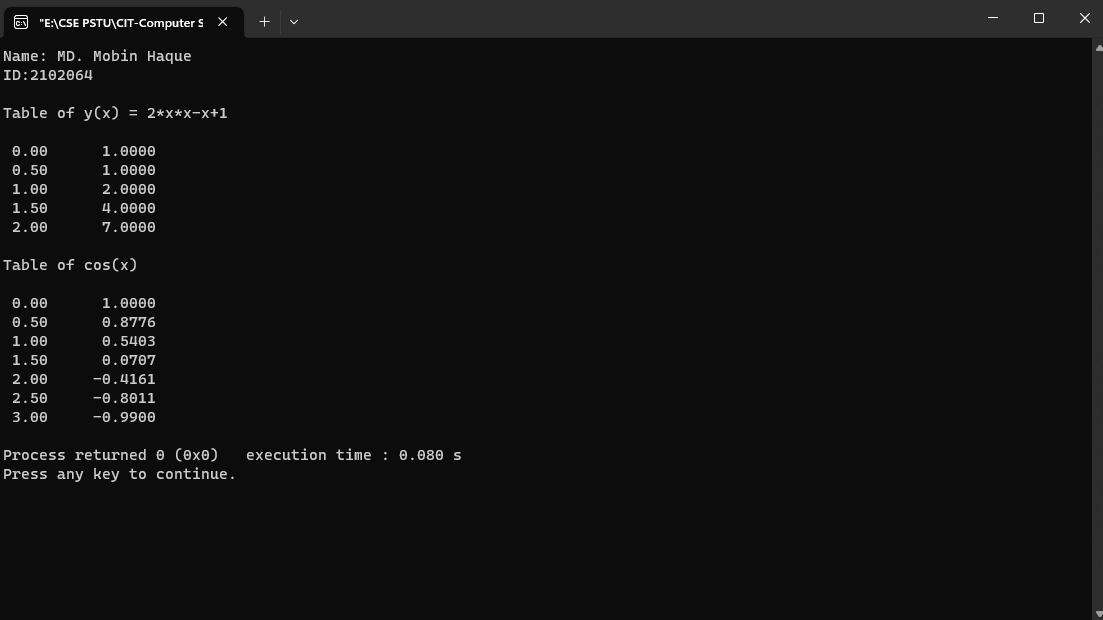
}

double y(double x)

{

return(2\*x\*x-x+1);

}



1. Write a program to illustrate the use of structure pointers.

struct invent

{

char \*name[20];

int number;

float price;

};

main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

struct invent product[3], \*ptr;

printf("INPUT\n\n");

for(ptr = product; ptr < product+3; ptr++)

scanf("%s %d %f", ptr->name, &ptr->number, &ptr->price);

printf("\nOUTPUT\n\n");

ptr = product;

while(ptr < product + 3)

{

printf("%-20s %5d %10.2f\n",

ptr->name,

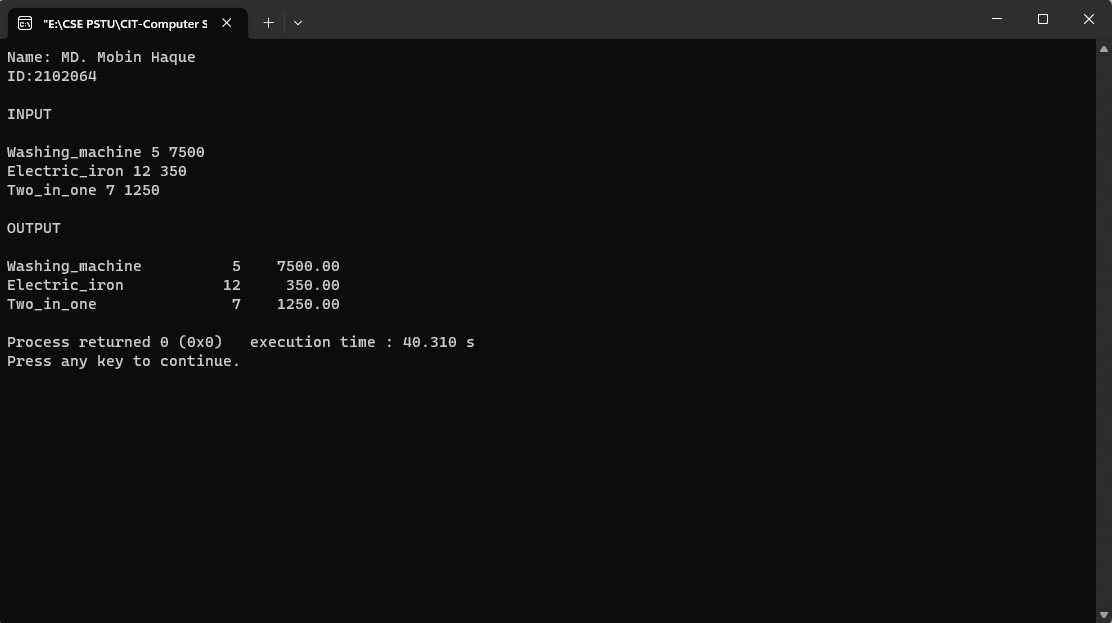
ptr->number,

ptr->price);

ptr++;

}

}



1. Write a program in C to show the simple structure of a function.

#include<stdio.h>

int sum (int, int);//function declaration

int main (void)

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int total;

printf("\n\n Function : a simple structure of function :\n");

printf("------------------------------------------------\n");

total = sum (5, 6);//function call

printf ("The total is : %d\n", total);

return 0;

}

int sum (int a, int b) //function definition

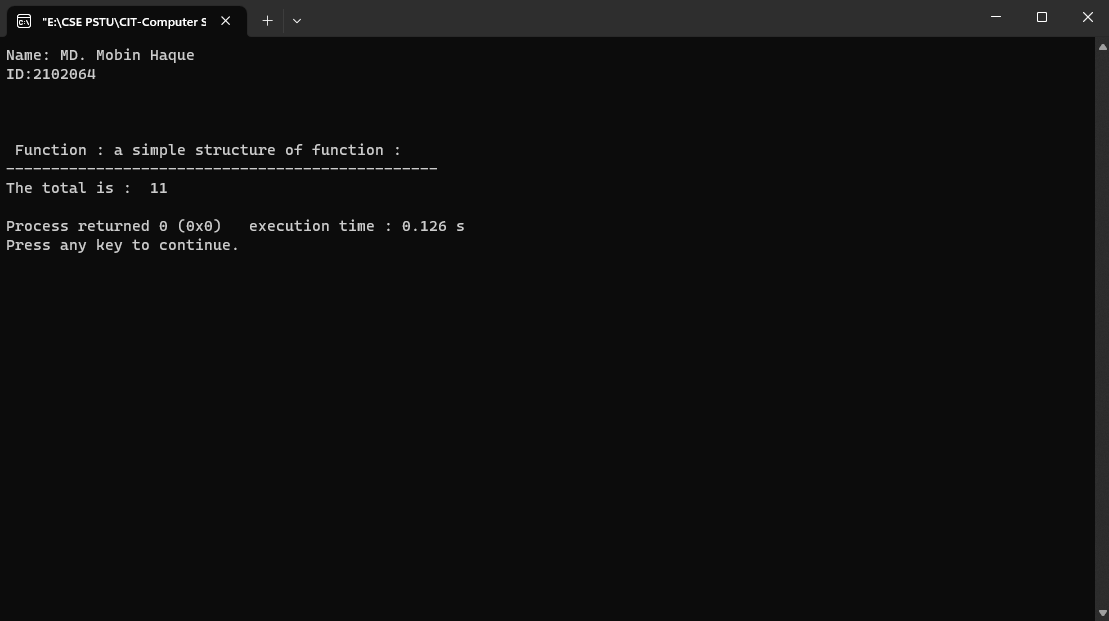
{

int s;

s = a + b;

return s; //function returning a value

}



1. Write a program in C to find the square of any number using the function.

#include <stdio.h>

int square(int num)

{

return num \* num;

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int num;

printf("Enter a number: ");

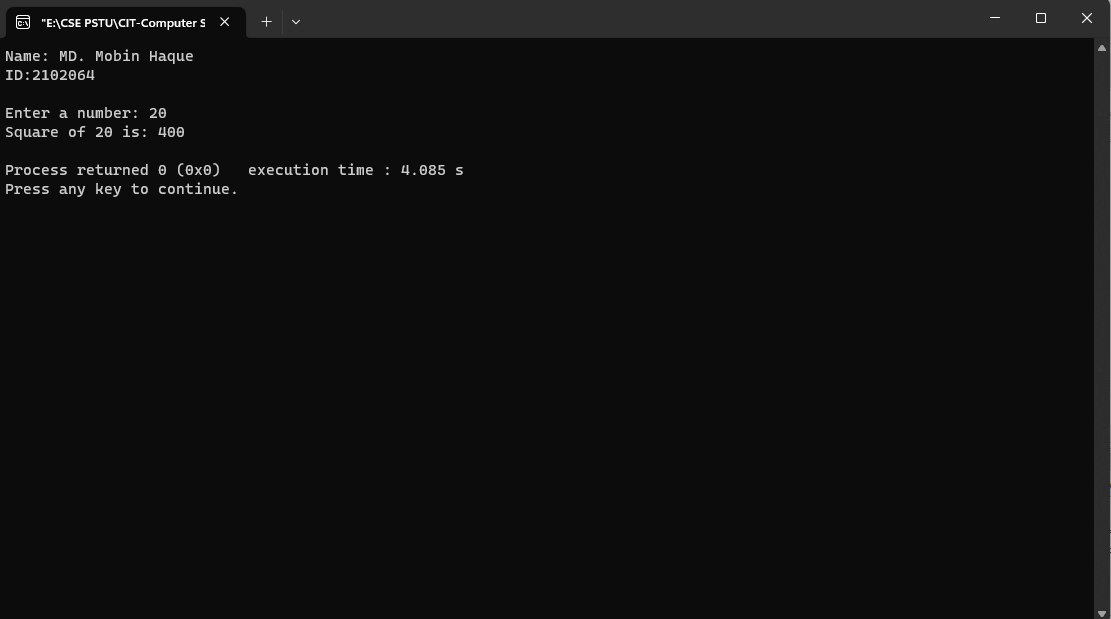
scanf("%d", &num);

int result = square(num);

printf("Square of %d is: %d\n", num, result);

return 0;

}



1. Write a program in C to swap two numbers using a function.

#include<stdio.h>

void swap(int\* num1, int\* num2)

{

int temp = \*num1;

\*num1 = \*num2;

\*num2 = temp;

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int num1, num2;

printf("Enter two numbers: ");

scanf("%d %d", &num1, &num2);

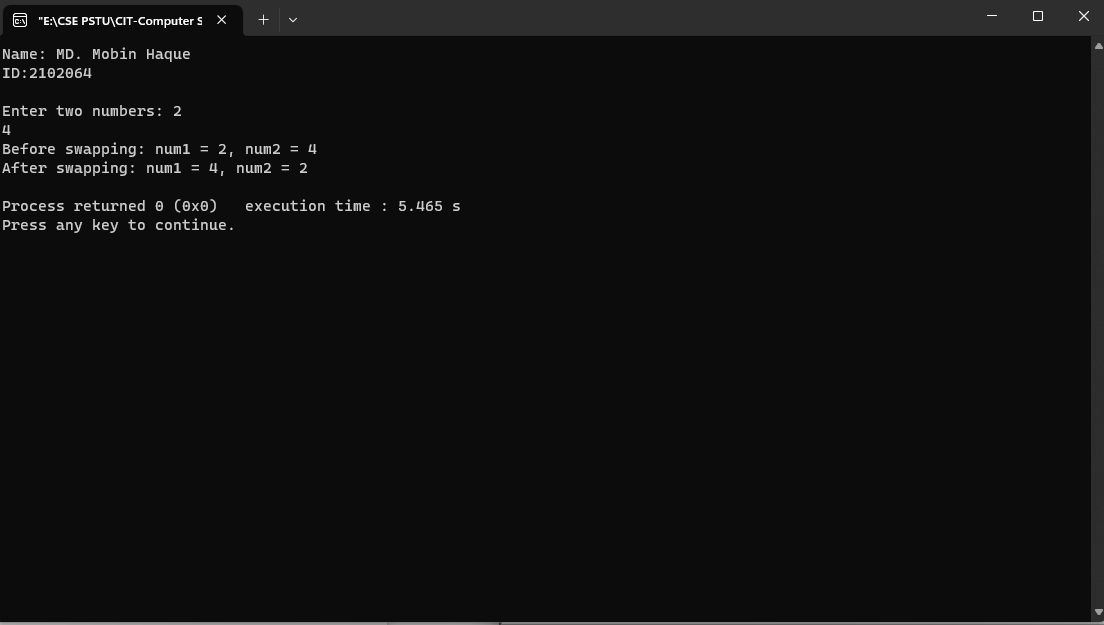
printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

swap(&num1, &num2);

printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

return 0;

}



1. Write a program in C to check if a given number is even or odd using the function.

#include<stdio.h>

int isEven(int num)

{

if (num % 2 == 0)

return 1; // Even

else

return 0; // Odd

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isEven(num))

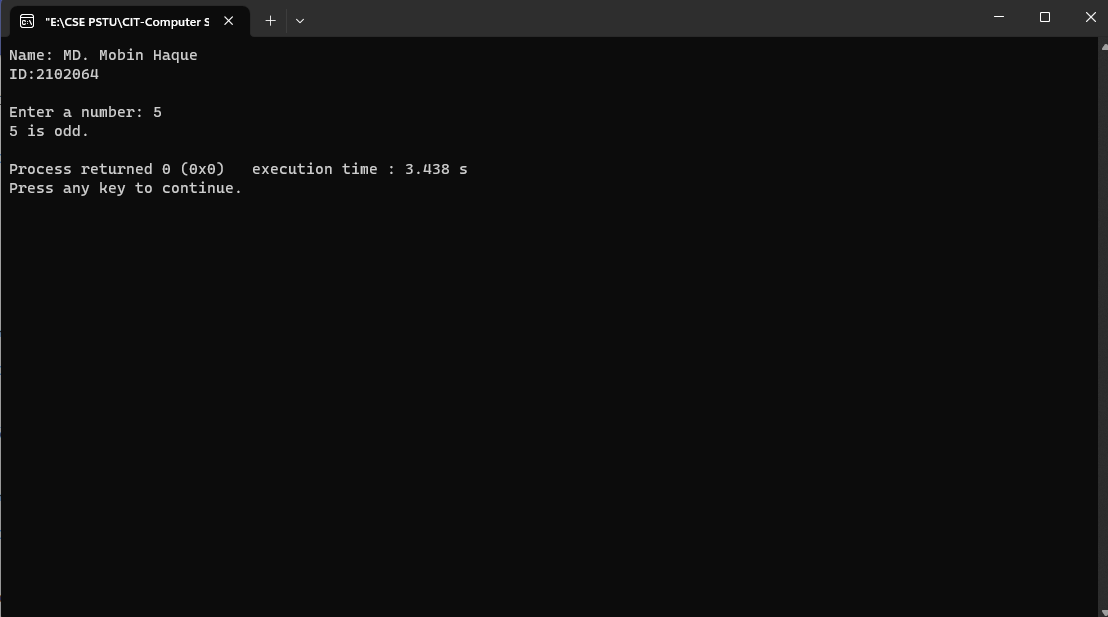
printf("%d is even.\n", num);

else

printf("%d is odd.\n", num);

return 0;

}



1. Write a program in C to find the sum of the series 1!/1+2!/2+3!/3+4!/4+5!/5 using the function.

#include<stdio.h>

int factorial(int num)

{

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

return 1;

else

return num \* factorial(num - 1);

}

float calculateSeriesSum()

{

float sum = 0.0;

for (int i = 1; i <= 5; i++)

{

sum += (float)factorial(i) / i;

}

return sum;

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

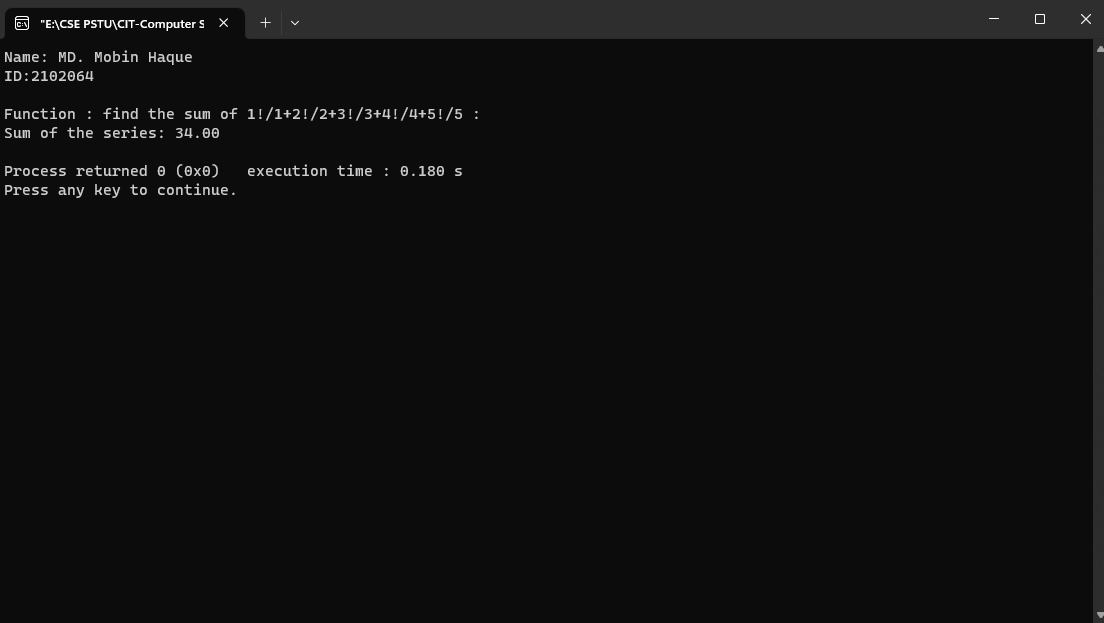
float seriesSum = calculateSeriesSum();

printf("Function : find the sum of 1!/1+2!/2+3!/3+4!/4+5!/5 :\n");

printf("Sum of the series: %.2f\n", seriesSum);

return 0;

}



1. Write a program in C to convert a decimal number to a binary number using the function.

#include<stdio.h>

void decimalToBinary(int decimal)

{

if (decimal > 0)

{

decimalToBinary(decimal / 2);

printf("%d", decimal % 2);

}

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int decimal;

printf("Enter a decimal number: ");

scanf("%d", &decimal);

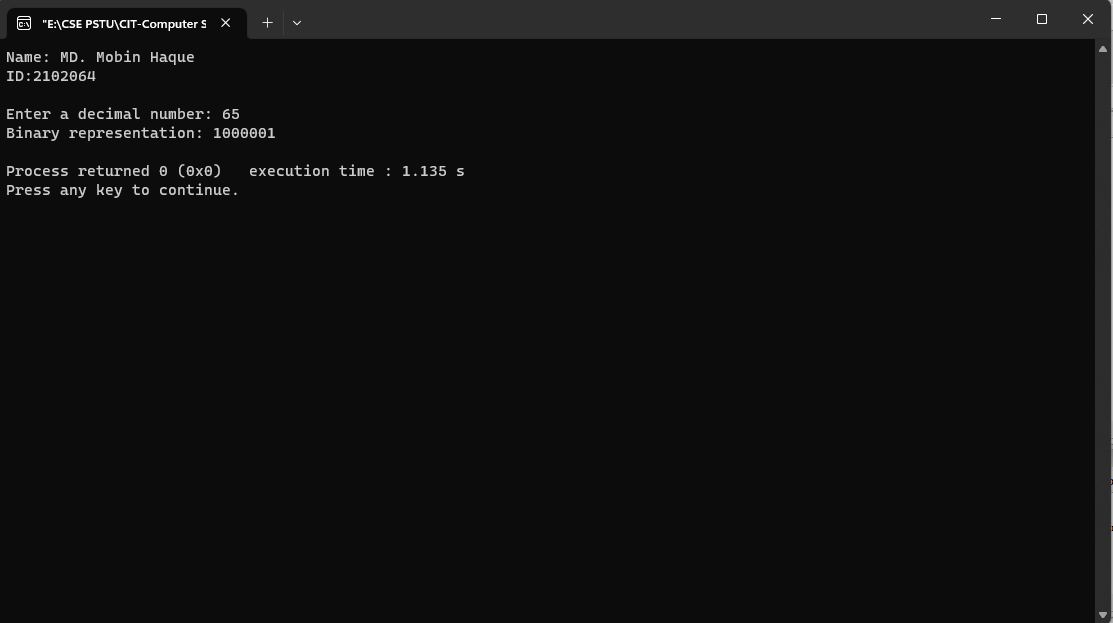
printf("Binary representation: ");

decimalToBinary(decimal);

printf("\n");

return 0;

}



1. Write a program in C to check whether a number is a prime number or not using the function.

#include<stdio.h>

int isPrime(int num)

{

if (num <= 1)

return 0; // Not prime

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

{

if (num % i == 0)

return 0; // Not prime

}

return 1; // Prime

}

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isPrime(num))

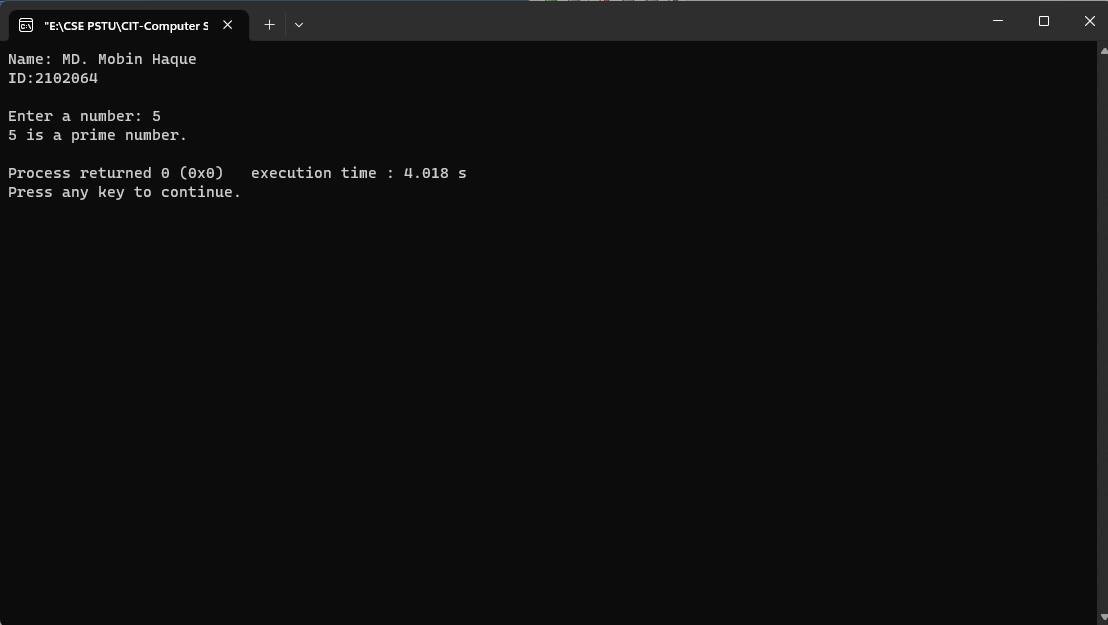
printf("%d is a prime number.\n", num);

else

printf("%d is not a prime number.\n", num);

return 0;

}



1. Write a program in C to get the largest element of an array using the function.

#include<stdio.h>

#define MAX 100

int findMaxElem(int []);

int n;

int main()

{

printf("Name: Nazmus Sakib\nID:2102066\n\n");

int arr1[MAX],mxelem,i;

printf("\n\n Function : get largest element of an array :\n");

printf("-------------------------------------------------\n");

printf(" Input the number of elements to be stored in the array :");

scanf("%d",&n);

printf(" Input %d elements in the array :\n",n);

for(i = 0; i < n; i++)

{

printf(" element - %d : ",i);

scanf("%d", &arr1[i]);

}

mxelem=findMaxElem(arr1);

printf(" The largest element in the array is : %d\n\n",mxelem);

return 0;

}

int findMaxElem(int arr1[])

{

int i = 1, mxelem;

mxelem = arr1[0];

while(i < n)

{

if(mxelem<arr1[i])

mxelem=arr1[i];

i++;

}

return mxelem;

}

