Practical file

C++

Name: Bhaskar malik

Roll no. : 16020

Course: b.sc(h) computer science

Question 1

#include <bits/stdc++.h>

using namespace std;

double Series(int n)

{

int i;

double sums = 0.0, ser;

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

{

ser = 1 / pow(i, i);

sums += ser;

}

return sums;

}

int main()

{

int n ;

cout<<"S=1-1/2^2 +1/3^3+...+1/n^n\n";

cout<<"enter the value of n: ";

cin >>n ;

cout<<"S = ";

double res = Series(n);

cout << res;

return 0;

}

A screenshot of a computer

Description automatically generated

Question 2

#include<iostream>

using namespace std;

int main ()

{

int A[10], B[10], n, i, j, k = 0;

cout << "Enter size of array : ";

cin >> n;

cout << "Enter elements of array : ";

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

cin >> A[i];

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

{

for (j = 0; j < k; j++)

{

if (A[i] == B[j])

break;

}

if (j == k)

{

B[k] = A[i];

k++;

}

}

cout << "Repeated elements after deletion : ";

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

cout << B[i] << " ";

return 0;

}

A screenshot of a computer

Description automatically generated

Question 3

#include <iostream>

using namespace std;

int main()

{

char str[100];

int i;

int freq[256] = {0};

cout<<"Enter the string: ";

gets(str);

for(i = 0; str[i] != '\0'; i++)

{

freq[str[i]]++;

}

for(i = 0; i < 256; i++)

{

if(freq[i] != 0)

{

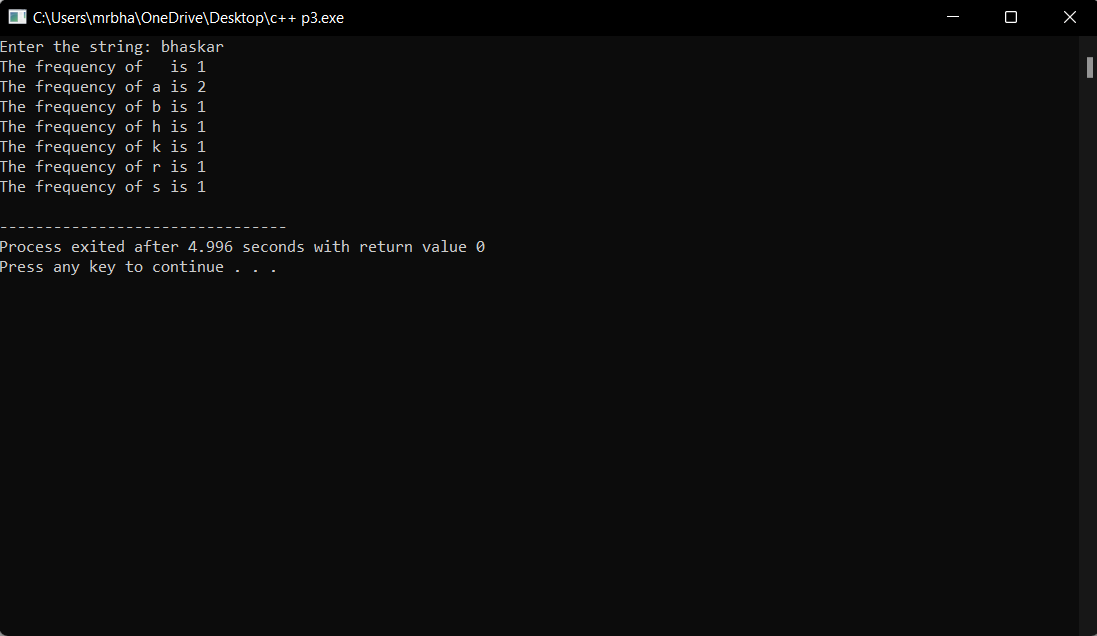
cout<<"The frequency of "<<char(i)<<" is "<<freq[i]<<endl;

}

}

return 0;

}



Question 4

#include <iostream>

#include <cstring>

using namespace std;

void showAddress(const char\* str) {

cout << "Address of each character in the string:" << endl;

while (\*str) {

cout << \*str << ": " << reinterpret\_cast<const void\*>(str) << endl;

str++;

}

}

void concatenateStrings(char\* str1, const char\* str2) {

while (\*str1) {

str1++;

}

while (\*str2) {

\*str1 = \*str2;

str1++;

str2++;

}

\*str1 = '\0';

}

int compareStrings(const char\* str1, const char\* str2) {

while (\*str1 && \*str2 && \*str1 == \*str2) {

str1++;

str2++;

}

return \*str1 - \*str2;

}

int calculateStringLength(const char\* str) {

int length = 0;

while (\*str) {

length++;

str++;

}

return length;

}

void convertToLowercase(char\* str) {

while (\*str) {

if (\*str >= 'A' && \*str <= 'Z') {

\*str += 32;

}

str++;

}

}

void convertToUppercase(char\* str) {

while (\*str) {

if (\*str >= 'a' && \*str <= 'z') {

\*str -= 32;

}

str++;

}

}

void reverseString(char\* str) {

char\* end = str;

while (\*end) {

end++;

}

end--;

while (str < end) {

char temp = \*str;

\*str = \*end;

\*end = temp;

str++;

end--;

}

}

void insertString(char\* str1, const char\* str2, int position) {

int length1 = calculateStringLength(str1);

int length2 = calculateStringLength(str2);

if (position < 0 || position > length1) {

cout << "Invalid position!" << endl;

return;

}

for (int i = length1; i >= position; i--) {

\*(str1 + i + length2) = \*(str1 + i);

}

for (int i = 0; i < length2; i++) {

\*(str1 + position + i) = \*(str2 + i);

}

}

int main() {

const int MAX\_SIZE = 100;

char str1[MAX\_SIZE];

char str2[MAX\_SIZE];

int choice;

int position;

cout << "Enter string 1: ";

cin.getline(str1, MAX\_SIZE);

cout << "Enter string 2: ";

cin.getline(str2, MAX\_SIZE);

cout << "String Manipulation Menu:" << endl;

cout << "1. Show address of each character in string" << endl;

cout << "2. Concatenate two strings" << endl;

cout << "3. Compare two strings" << endl;

cout << "4. Calculate length of the string" << endl;

cout << "5. Convert all uppercase characters to lowercase" << endl;

cout << "6. Convert all lowercase characters to uppercase" << endl;

cout << "7. Reverse the string" << endl;

cout << "8. Insert a string in another string at a user-specified position" << endl;

cout << "Enter your choice (1-8): ";

cin >> choice;

switch (choice) {

case 1:

showAddress(str1);

break;

case 2:

concatenateStrings(str1, str2);

cout << "Concatenated string: " << str1 << endl;

break;

case 3:

if (compareStrings(str1, str2) == 0) {

cout << "Strings are equal." << endl;

} else {

cout << "Strings are not equal." << endl;

}

break;

case 4:

cout << "Length of string 1: " << calculateStringLength(str1) << endl;

cout << "Length of string 2: " << calculateStringLength(str2) << endl;

break;

case 5:

convertToLowercase(str1);

convertToLowercase(str2);

cout << "Converted string 1: " << str1 << endl;

cout << "Converted string 2: " << str2 << endl;

break;

case 6:

convertToUppercase(str1);

convertToUppercase(str2);

cout << "Converted string 1: " << str1 << endl;

cout << "Converted string 2: " << str2 << endl;

break;

case 7:

reverseString(str1);

reverseString(str2);

cout << "Reversed string 1: " << str1 << endl;

cout << "Reversed string 2: " << str2 << endl;

break;

case 8:

cout << "Enter the position to insert string 2 in string 1: ";

cin >> position;

insertString(str1, str2, position);

cout << "Modified string 1: " << str1 << endl;

break;

default:

cout << "Invalid choice!" << endl;

break;

}

return 0;

}

A screenshot of a computer program

Description automatically generated with medium confidence

Question 5

#include<iostream>

#include<algorithm>

using namespace std;

void mergeArrays(int arr1[], int arr2[], int n1, int n2, int arr3[])

{

int i = 0, j = 0, k = 0;

while(i < n1){

arr3[k++] = arr1[i++];

}

while(j < n2){

arr3[k++] = arr2[j++];

}

sort(arr3, arr3+n1+n2);

}

int main()

{

int arr1[] = {1, 3, 5, 7};

int n1 = sizeof(arr1) / sizeof(arr1[0]);

int arr2[] = {2, 4, 6, 8};

int n2 = sizeof(arr2) / sizeof(arr2[0]);

int arr3[n1+n2];

mergeArrays(arr1, arr2, n1, n2, arr3);

cout << "Array after merging" <<endl;

for (int i=0; i < n1+n2; i++)

cout << arr3[i] << " ";

return 0;

}

A screenshot of a computer

Description automatically generated

Question 6

#include<iostream>

using namespace std;

int main()

{

int arr[10], Size, i, Search, Flag;

cout<<" Please Enter the size of an array : ";

cin>>("d",Size);

cout<<"Please Enter d elements of an array: ", Size;

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

{

cin>>arr[i];

}

cout<<" Please Enter the Search Element : ";

cin>>Search;

Flag = 0;

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

{

if(arr[i] == Search)

{

Flag = 1;

break;

}

}

if(Flag == 1)

{

cout<<"\n the element is present", Search, i + 1;

}

else

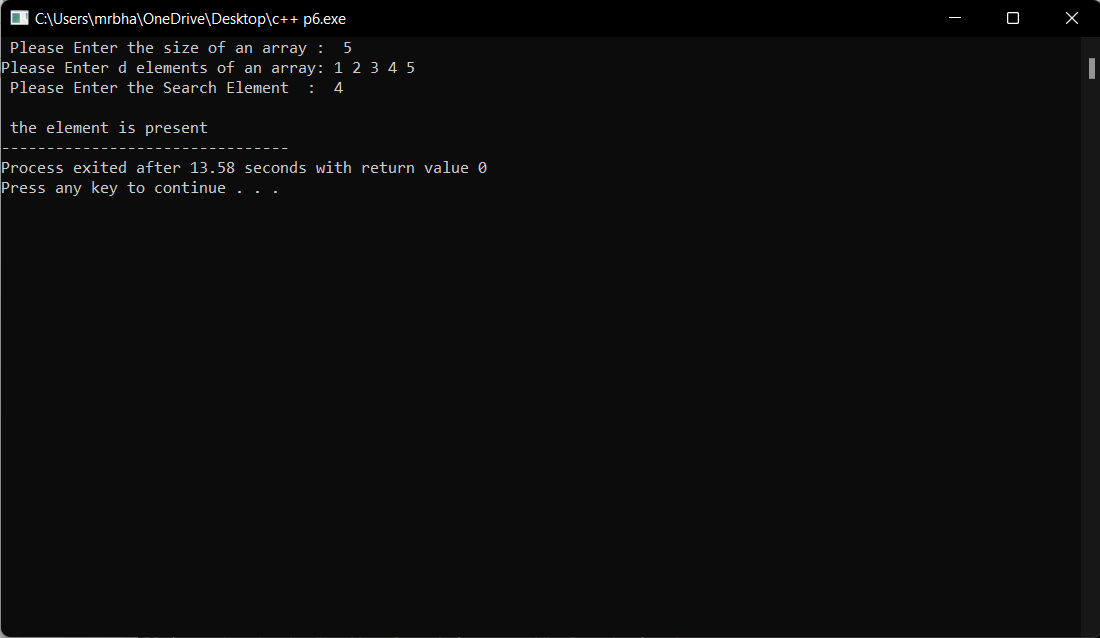
{

cout<<"\n Sorry!! the element is not present ", Search;

}

return 0;

}



Question 7

#include <iostream>

using namespace std;

int main() {

int n1, n2, gcd;

cout << "Enter two numbers: ";

cin >> n1 >> n2;

// swapping variables n1 and n2 if n2 is greater than n1.

if ( n2 > n1) {

int temp = n2;

n2 = n1;

n1 = temp;

}

for (int i = 1; i <= n2; ++i) {

if (n1 % i == 0 && n2 % i ==0) {

gcd = i;

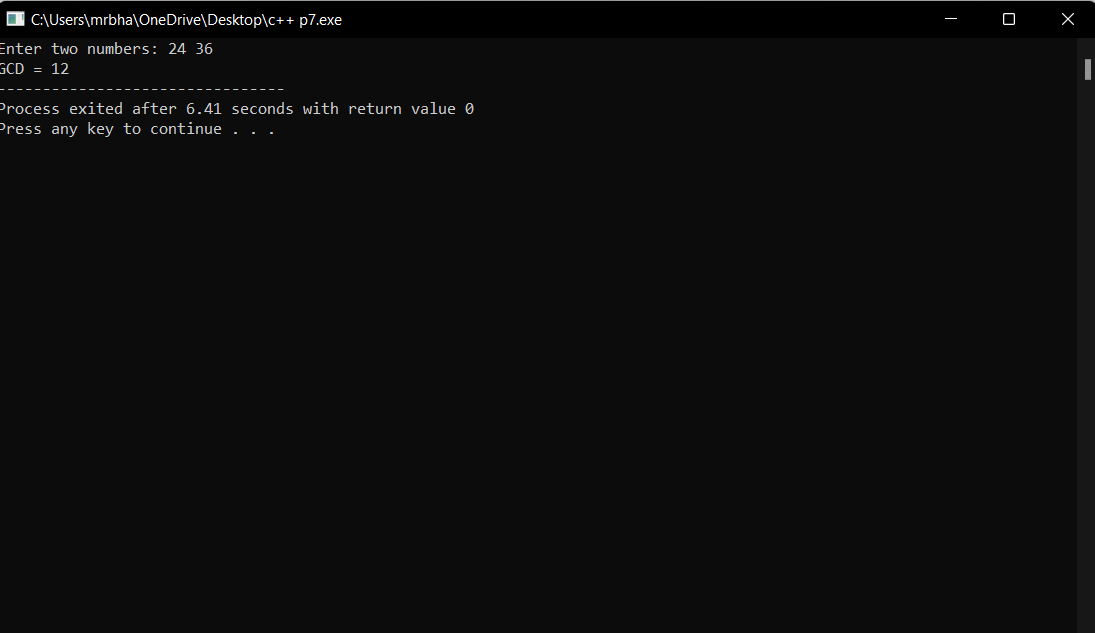
}

}

cout << "GCD = " << gcd;

return 0;

}



Question 8

#include <iostream>

#include <stdexcept>

using namespace std;

class Matrix {

private:

int rows;

int columns;

int\*\* data;

public:

Matrix(int r, int c) : rows(r), columns(c) {

data = new int\*[rows];

for (int i = 0; i < rows; i++) {

data[i] = new int[columns];

}

}

~Matrix() {

for (int i = 0; i < rows; i++) {

delete[] data[i];

}

delete[] data;

}

void readMatrix() {

cout << "Enter the elements of the matrix:" << endl;

for (int i = 0; i < rows; i++) {

for (int j = 0; j < columns; j++) {

cin >> data[i][j];

}

}

}

void displayMatrix() {

cout << "Matrix:" << endl;

for (int i = 0; i < rows; i++) {

for (int j = 0; j < columns; j++) {

cout << data[i][j] << " ";

}

cout << endl;

}

}

Matrix add(const Matrix& other) {

if (rows != other.rows || columns != other.columns) {

throw runtime\_error("Matrices are incompatible for addition.");

}

Matrix result(rows, columns);

for (int i = 0; i < rows; i++) {

for (int j = 0; j < columns; j++) {

result.data[i][j] = data[i][j] + other.data[i][j];

}

}

return result;

}

Matrix multiply(const Matrix& other) {

if (columns != other.rows) {

throw runtime\_error("Matrices are incompatible for multiplication.");

}

Matrix result(rows, other.columns);

for (int i = 0; i < rows; i++) {

for (int j = 0; j < other.columns; j++) {

for (int k = 0; k < columns; k++) {

result.data[i][j] += data[i][k] \* other.data[k][j];

}

}

}

return result;

}

Matrix transpose() {

Matrix result(columns, rows);

for (int i = 0; i < columns; i++) {

for (int j = 0; j < rows; j++) {

result.data[i][j] = data[j][i];

}

}

return result;

}

};

int main() {

int rows, columns;

cout << "Enter the number of rows in the matrices: ";

cin >> rows;

cout << "Enter the number of columns in the matrices: ";

cin >> columns;

Matrix matrix1(rows, columns);

Matrix matrix2(rows, columns);

cout << "Enter the elements for Matrix 1:" << endl;

matrix1.readMatrix();

cout << "Enter the elements for Matrix 2:" << endl;

matrix2.readMatrix();

cout << "Matrix 1:" << endl;

matrix1.displayMatrix();

cout << "Matrix 2:" << endl;

matrix2.displayMatrix();

try {

Matrix sumMatrix = matrix1.add(matrix2);

cout << "Sum of Matrix 1 and Matrix 2:" << endl;

sumMatrix.displayMatrix();

} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

try {

Matrix productMatrix = matrix1.multiply(matrix2);

cout << "Product of Matrix 1 and Matrix 2:" << endl;

productMatrix.displayMatrix();

} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

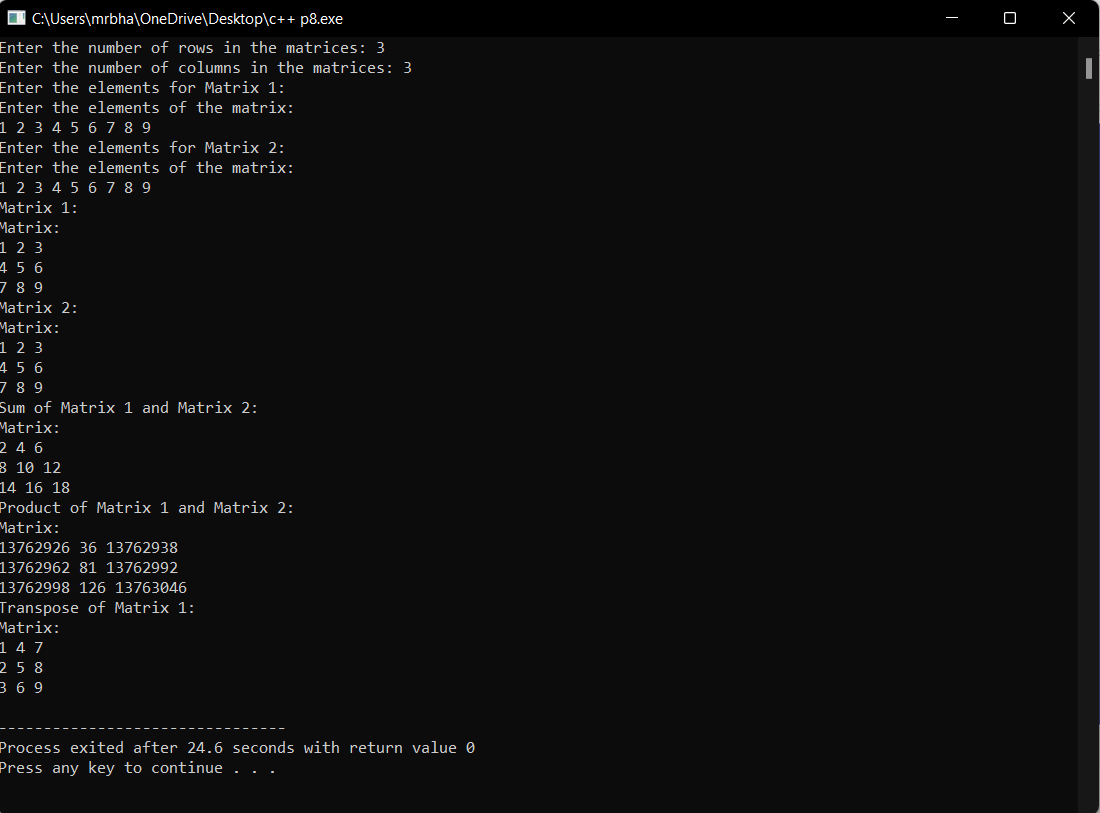
Matrix transposeMatrix = matrix1.transpose();

cout << "Transpose of Matrix 1:" << endl;

transposeMatrix.displayMatrix();

return 0;

}



Question 9

#include <iostream>

#include <string>

using namespace std;

class Person {

protected:

string name;

public:

Person(const string& n) : name(n) {}

virtual void display() {

cout << "Name: " << name << endl;

}

};

class Student : public Person {

private:

string course;

int marks;

int year;

public:

Student(const string& n, const string& c, int m, int y) : Person(n), course(c), marks(m), year(y) {}

void display() override {

cout << "Name: " << name << endl;

cout << "Course: " << course << endl;

cout << "Marks: " << marks << endl;

cout << "Year: " << year << endl;

}

};

class Employee : public Person {

private:

string department;

int salary;

public:

Employee(const string& n, const string& d, int s) : Person(n), department(d), salary(s) {}

void display() override {

cout << "Name: " << name << endl;

cout << "Department: " << department << endl;

cout << "Salary: " << salary << endl;

}

};

int main() {

Person\* person1 = new Person("Hiten Singhla");

Person\* person2 = new Student("Kanishk Chauhan", "Computer Science", 70, 2023);

Person\* person3 = new Employee("Dhruv", "Engineering", 50000);

person1->display();

cout << endl;

person2->display();

cout << endl;

person3->display();

cout << endl;

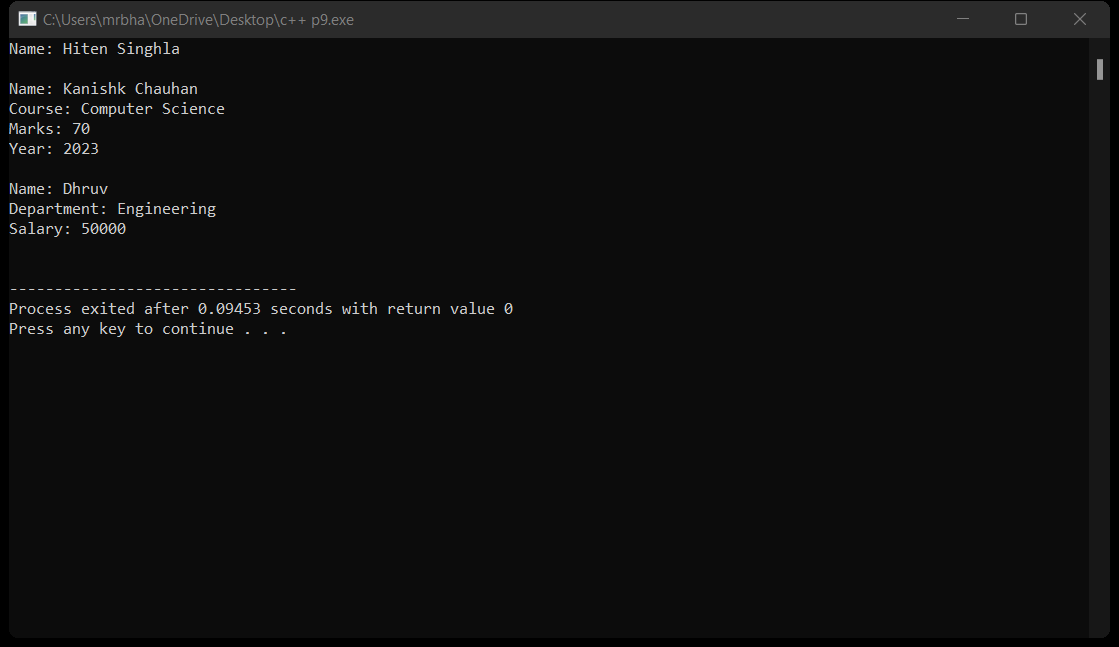
delete person1;

delete person2;

delete person3;

return 0;

}



Question 10

#include <iostream>

#include <cmath>

#include <stdexcept>

using namespace std;

class Triangle {

private:

double side1;

double side2;

double side3;

public:

Triangle(double s1, double s2, double s3) : side1(s1), side2(s2), side3(s3) {

if (side1 <= 0 || side2 <= 0 || side3 <= 0) {

throw invalid\_argument("Invalid side length. Side lengths must be greater than 0.");

}

if (side1 + side2 <= side3 || side1 + side3 <= side2 || side2 + side3 <= side1) {

throw invalid\_argument("Invalid side lengths. The sum of any two sides must be greater than the third side.");

}

}

double calculateArea() {

double s = (side1 + side2 + side3) / 2.0; // Calculate the semi-perimeter

return sqrt(s \* (s - side1) \* (s - side2) \* (s - side3)); // Heron's formula

}

double calculateArea(double base, double height) {

if (side1 == base && side2 == height) {

return (base \* height) / 2.0; // Area of a right-angled triangle

} else {

throw invalid\_argument("Invalid sides for a right-angled triangle.");

}

}

};

int main() {

try {

Triangle triangle1(3, 4, 5); // Right-angled triangle

cout << "Area of right-angled triangle: " << triangle1.calculateArea(4, 3) << endl;

} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

try {

Triangle triangle2(2, 3, 4); // Non-right-angled triangle

cout << "Area of triangle using Heron's formula: " << triangle2.calculateArea() << endl;

} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

try {

Triangle triangle3(0, 2, 3); // Invalid side length (0)

} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

try {

Triangle triangle4(2, 5, 10); // Invalid side lengths (sum of any two sides is not greater than the third side)

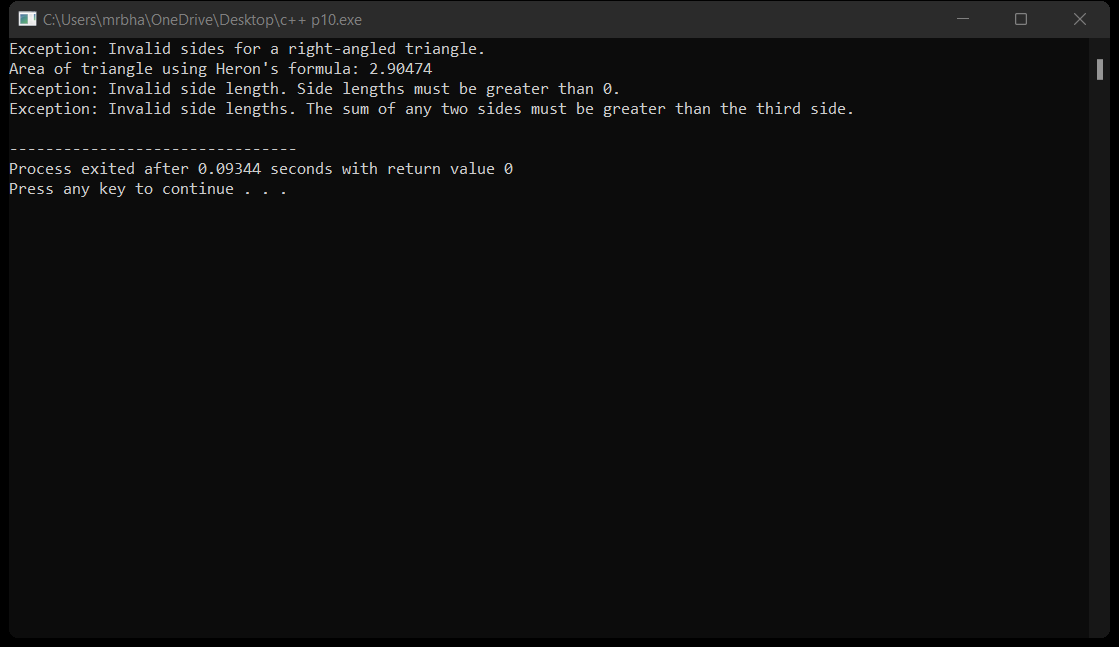
} catch (const exception& e) {

cout << "Exception: " << e.what() << endl;

}

return 0;

}



Question 11

#include <iostream>

#include <fstream>

using namespace std;

class Student {

private:

int rollNo;

string name;

int classYear;

double totalMarks;

public:

Student() {}

Student(int roll, const string& n, int year, double marks)

: rollNo(roll), name(n), classYear(year), totalMarks(marks) {}

int getRollNo() const {

return rollNo;

}

string getName() const {

return name;

}

int getClassYear() const {

return classYear;

}

double getTotalMarks() const {

return totalMarks;

}

void display() {

cout << "Roll No: " << rollNo << endl;

cout << "Name: " << name << endl;

cout << "Class Year: " << classYear << endl;

cout << "Total Marks: " << totalMarks << endl;

cout << endl;

}

};

int main() {

Student students[3];

for (int i = 0; i < 3; i++) {

int rollNo, classYear;

string name;

double totalMarks;

cout << "Enter details for Student " << i + 1 << ":" << endl;

cout << "Roll No: ";

cin >> rollNo;

cout << "Name: ";

cin.ignore();

getline(cin, name);

cout << "Class Year: ";

cin >> classYear;

cout << "Total Marks: ";

cin >> totalMarks;

students[i] = Student(rollNo, name, classYear, totalMarks);

cout << endl;

}

ofstream outFile("students.txt", ios::out);

if (!outFile) {

cerr << "Failed to open file." << endl;

return 1;

}

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

outFile << "Roll No: " << students[i].getRollNo() << endl;

outFile << "Name: " << students[i].getName() << endl;

outFile << "Class Year: " << students[i].getClassYear() << endl;

outFile << "Total Marks: " << students[i].getTotalMarks() << endl;

outFile << endl;

}

outFile.close();

ifstream inFile("students.txt", ios::in);

if (!inFile) {

cerr << "Failed to open file." << endl;

return 1;

}

cout << "Student Records:" << endl;

string line;

while (getline(inFile, line)) {

cout << line << endl;

}

inFile.close();

return 0;

}

A screenshot of a computer

Description automatically generated