Varsha

(22570063)

Bsc.(Hons.)Computer Science

C++ Practical File (second semester)

KALINDI COLLEGE



**INDEX**

Practical List

1.Write a program to compute the sum of the first n terms of the following series:

S=1-1/2^2+1/3^3-………1/n^n

The number of terms n is to be taken from the user through the command line. If the command line argument is not found then prompt the user to enter the value of n.

2.Write a program to remove the duplicates from an array.

3.Write a program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments.

4.Write a menu driven program to perform string manipulation (without using inbuilt string functions):

a. Show address of each character in string

b. Concatenate two strings.

c. Compare two strings

d. Calculate length of the string (use pointers)

e. Convert all lowercase characters to uppercase

f. Reverse the string

g. Insert a string in another string at a user specified position

5.Write a program to merge two ordered arrays to get a single ordered array.

6.Write a program to search a given element in a set of N numbers.

7.Write a program to calculate GCD of two numbers.

8.Create a Matrix class. Write a menu-driven program to perform following Matrix operations (exceptions should be thrown by the functions if matrices passed to them are incompatible and handled by the main() function):

a. Sum

b. Product

c. Transpose

9.Define a class Person having name as a data member. Inherit two classes Student and Employee from Person. Student has additional attributes as course, marks and year and Employee has department and salary. Write display() method in all the three classes to display the corresponding attributes. Provide the necessary methods to show runtime polymorphism.

10.Create a Triangle class.  Add exception handling statements to ensure the following conditions: all sides are greater than 0 and sum of any two sides is greater than the third side.   The class should also have overloaded functions for calculating the area of a right angled triangle as well as using Heron's formula to calculate the area of any type of triangle.

11. Copy the contents of one text file to another file, after removing all whitespaces.

Ques1.write a program to compute the sum of the first n terms of the following series:

S=1- 1/2^2 +1/3^3 -1/4^4 +1/5^5 …..

The number of terms n is to be taken from the user through the command line.if the command line argument is not found then prompt the user to enter the value of n.

#include<iostream>

#include<cmath>

using namespace std;

int main()

{int i,n;

float sum;

sum =0;

cout << "\n Enter value for n:";

cin >> n;

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

{

if (i%2==0)

sum=sum-1/pow(i,i);

else

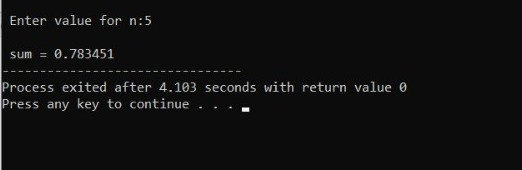
sum=sum+1/pow(i,i);

}

cout<<"\n sum = " << sum;

return 0;

}



Ques2. Write a program to remove duplicates from an array.

#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++;

}

}

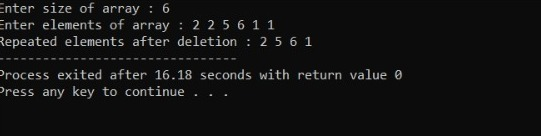
cout << "Repeated elements after deletion : ";

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

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

return 0;

}



Ques 3. Write a program that prints a table indicating the number of occurrences of each alphabet in the text entered as command line arguments .

#include<iostream>

#include<string.h>

using namespace std;

int main(int argc, char\*argv[])

{

int i;

int count=0;

cout << "\n Parameters are \n";

for (int i=0; i<argc; i++)

cout << argv[i] << " ";

for (char k ='a';k<='z';k++)

{

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

{

for (int j=0; argv[i][j]!='\0';j++)

{

if(isupper(argv[i][j])) argv[i][j]=tolower(argv[i][j]);

if(argv[i][j]==k)

count++;

}

}

if(count>0)

cout <<"\n"<<k<<" occurs "<<count<<" times.";

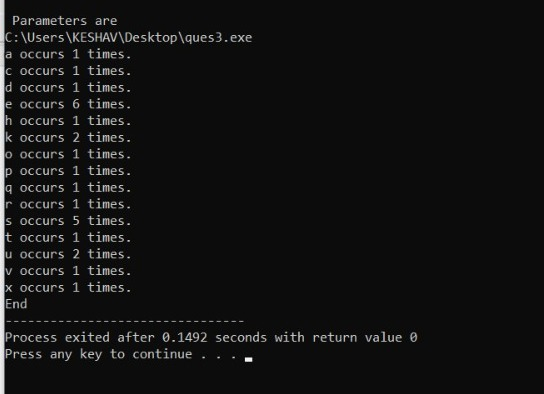
count=0;

}

cout<<"\nEnd";

return 0;

}



Ques 4. Write a menu driven program to perform string manipulation (without using inbuilt string function).

1. Show address of each character in string.
2. Concatenate two strings.
3. Compare two strings.
4. Calculate the length of the strings(use pointers).
5. Convert all lowercase to uppercase.
6. Reverse the string.
7. Insert a string in another string at a user specified position.

#include <iostream>

#include <cstring>

using namespace std;

void show\_address(char\* str) {

std::cout << "The addresses of each character in the string are:\n";

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

std::cout << str[i] << " = " << (void\*)&str[i] << '\n';

}

}

void concatenate\_strings(char\* str1, char\* str2) {

int len1 = strlen(str1);

int len2 = strlen(str2);

char result[len1+len2+1];

int i, j;

for (i = 0; i < len1; i++) {

result[i] = str1[i];

}

for (j = 0; j < len2; j++) {

result[i+j] = str2[j];

}

result[i+j] = '\0';

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

}

bool compare\_strings(char\* str1, char\* str2) {

if (strlen(str1) != strlen(str2)) {

return false;

}

for (int i = 0; i < strlen(str1); i++) {

if (str1[i] != str2[i]) {

return false;

}

}

return true;

}

int length(const char\* str) {

int len = 0;

const char\* ptr = &str[0];

while (\*ptr) {

len++;

ptr++;

}

return len;

}

void to\_uppercase(char\* str) {

int i = 0;

while (str[i] != '\0') {

if (str[i] >= 'a' && str[i] <= 'z') {

str[i] -= 'a' - 'A';

}

i++;

}

cout<<"Upper Case String :"<<str<<endl;

}

void reverse(char\* str) {

int len = strlen(str);

for (int i = 0; i < len/2; i++) {

char temp = str[i];

str[i] = str[len-i-1];

str[len-i-1] = temp;

}

cout << "Reversed string: " << str << endl;

}

void insert(char\* str1, char\* str2, int pos) {

int len1 = strlen(str1);

int len2 = strlen(str2);

char result[len1+len2+1];

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

result[i] = str1[i];

}

for (int j = 0; j < len2; j++) {

result[pos+j] = str2[j];

}

for (int k = pos; k < len1; k++) {

result[k+len2] = str1[k];

}

result[len1+len2] = '\0';

cout << "Result string: " << result << endl;

}

int main() {

while (true) {

cout << "\nMENU" << 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 lowercase characters to uppercase" << endl;

cout << "6. Reverse the string" << endl;

cout << "7. Insert a string in another string at a user specified position" << endl;

cout << "8. Exit" << endl;

int choice;

cout << "Enter your choice: ";

cin >> choice;

if (choice == 1) {

char str[100];

std::cout << "Enter a string: ";

std::cin >> str;

show\_address(str);

return 0;

}

else if (choice == 2) {

char str1[100], str2[100];

cout << "Enter the first string: ";

cin >> str1;

cout << "Enter the second string: ";

cin >> str2;

concatenate\_strings(str1, str2);

}

else if (choice == 3) {

char str1[100], str2[100];

cout << "Enter the first string: ";

cin >> str1;

cout << "Enter the second string: ";

cin >> str2;

if (compare\_strings(str1, str2)) {

cout << "The strings are equal." << endl;

}

else {

cout << "The strings are not equal." << endl;

}

}

else if (choice == 4) {

char str[200];

const char\* ptr = &str[0];

cout << "Enter a string: ";

cin >> str;

int len = length(ptr);

cout << "Length of the string: " << len << endl;

return 0;

}

else if (choice == 5) {

char str[100];

cout << "Enter a string: ";

cin >> str;

to\_uppercase(str);

}

else if (choice == 6) {

char str[100];

cout << "Enter a string: ";

cin >> str;

reverse(str);

}

else if (choice == 7) {

char str1[100], str2[100];

int pos;

cout << "Enter the first string: ";

cin >> str1;

cout << "Enter the second string: ";

cin >> str2;

cout << "Enter the position at which to insert the second string: ";

cin >> pos;

insert(str1, str2, pos);

}

else if (choice == 8) {

break;

}

else {

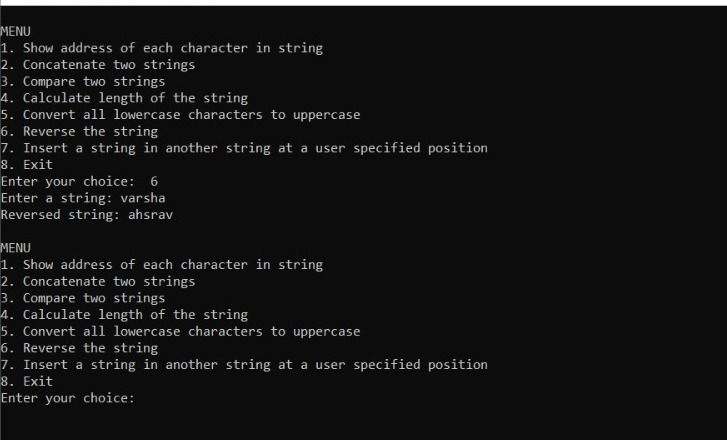
cout << "Invalid choice. Please enter a valid choice." << endl;

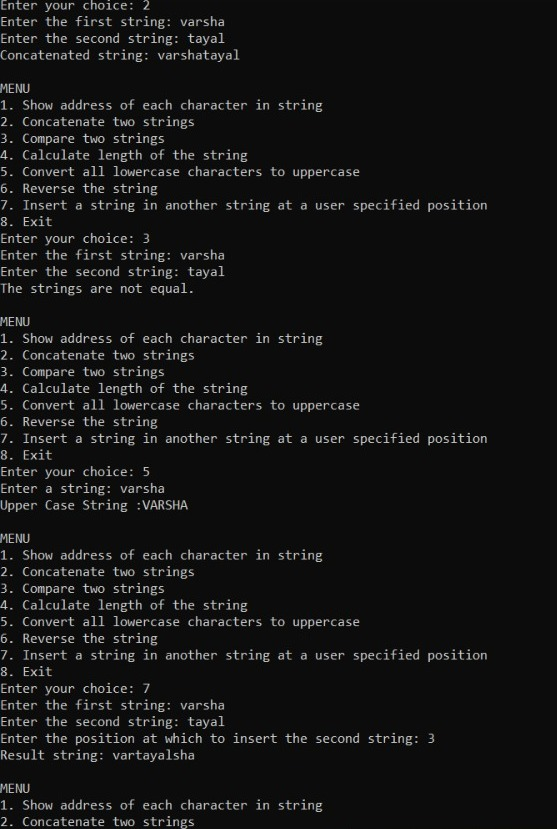
}

}

return 0;

}





Ques 5. Write a program to merge two ordered arrays to get a single ordered array.

#include<iostream>

using namespace std;

int main()

{

int arrOne[50], arrTwo[50], arrMerge[100];

int sizeOne, sizeTwo, sizeMerge, i, j, temp;

cout<<"Enter the Size for First Array: ";

cin>>sizeOne;

cout<<"Enter the Size for Second Array: ";

cin>>sizeTwo;

cout<<"\nEnter "<<sizeOne<<" Elements for First Array: ";

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

cin>>arrOne[i];

cout<<"\nEnter "<<sizeTwo<<" Elements for Second Array: ";

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

cin>>arrTwo[i];

// merging the two arrays

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

{

arrMerge[i] = arrOne[i];

}

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

{

arrMerge[i] = arrTwo[j]; i++;

}

sizeMerge = i;

// sorting the merged array in ascending order

for(j=0; j<(sizeMerge-1); j++)

{

for(i=0; i<(sizeMerge-1); i++)

{

if(arrMerge[i]>arrMerge[i+1])

{

temp = arrMerge[i];

arrMerge[i] = arrMerge[i+1];

arrMerge[i+1] = temp;

}

}

}

cout<<"\nThe New Array (Merged Array):\n";

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

{

if(i==(sizeMerge-1))

cout<<arrMerge[i];

else

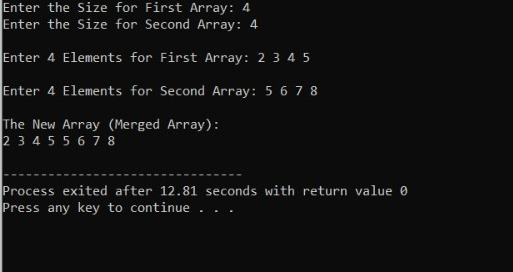
cout<<arrMerge[i]<<" ";

}

cout<<endl;

return 0;

}



Ques6. Write a program to search a given element In a set of N numbers using binary search

1. With recursion (ii) without recursion

#include<iostream>

using namespace std;

int main()

{

int arr[10], i, num, n, cnt=0, pos;

cout<<"\n Enter Array Size : ";

cin>>n;

cout<<"\n Enter Array Elements : \n";

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

{

cout<<" ";

cin>>arr[i];

}

cout<<"\n Enter Element to be Searched : ";

cin>>num;

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

{

if(arr[i]==num)

{

cnt=1;

pos=i+1;

break;

}

}

if(cnt==0)

{

cout<<"\n Element Not Found..!!";

}

else

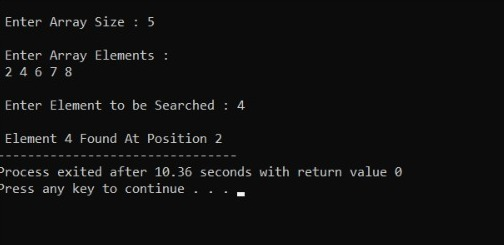
{

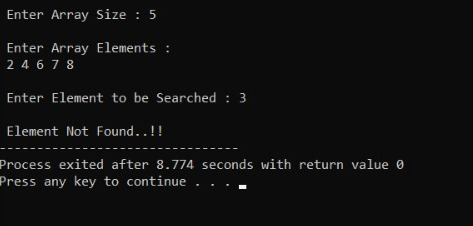
cout<<"\n Element "<<num<<" Found At Position "<<pos;

}

return 0;

}





Ques 7. Write a program to calculate GCD of two numbers (i)with recursion

(ii)without recursion

#include <iostream>

using namespace std;

int main() {

int n1, n2, hcf;

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) {

hcf = i;

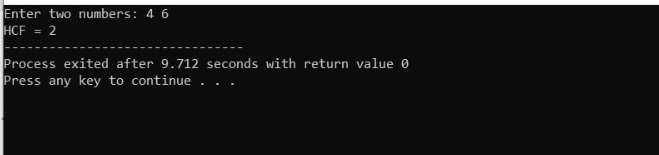
}

}

cout << "HCF = " << hcf;

return 0;

}



Ques8. Create a matrix class .write a menu driven program to perform following matrix operations (exception should be thrown by the functions if matrices passed to them are incompatible and handled by the main() function):

1. Sum
2. Product
3. Transpose

#include <iostream>

#include <vector>

using namespace std;

class Matrix {

private:

vector<vector<int>> data;

int rows;

int cols;

public:

Matrix(int rows, int cols) : rows(rows), cols(cols) {

data.resize(rows, vector<int>(cols, 0));

}

void setElement(int row, int col, int value) {

data[row][col] = value;

}

int getElement(int row, int col) const {

return data[row][col];

}

int getRows() const {

return rows;

}

int getCols() const {

return cols;

}

Matrix operator+(const Matrix& other) const {

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

throw runtime\_error("Incompatible matrices for addition");

}

Matrix result(rows, cols);

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

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

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

}

}

return result;

}

Matrix operator\*(const Matrix& other) const {

if (cols != other.rows) {

throw runtime\_error("Incompatible matrices for multiplication");

}

Matrix result(rows, other.cols);

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

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

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

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

}

}

}

return result;

}

Matrix transpose() const {

Matrix result(cols, rows);

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

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

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

}

}

return result;

}

};

void printMatrix(const Matrix& matrix) {

for (int i = 0; i < matrix.getRows(); i++) {

for (int j = 0; j < matrix.getCols(); j++) {

cout << matrix.getElement(i, j) << " ";

}

cout << endl;

}

}

int main() {

int rows1, cols1;

cout << "Enter the dimensions of the first matrix: ";

cin >> rows1 >> cols1;

Matrix matrix1(rows1, cols1);

cout << "Enter the elements of the first matrix:\n";

for (int i = 0; i < rows1; i++) {

for (int j = 0; j < cols1; j++) {

int value;

cin >> value;

matrix1.setElement(i, j, value);

}

}

int rows2, cols2;

cout << "Enter the dimensions of the second matrix: ";

cin >> rows2 >> cols2;

Matrix matrix2(rows2, cols2);

cout << "Enter the elements of the second matrix:\n";

for (int i = 0; i < rows2; i++) {

for (int j = 0; j < cols2; j++) {

int value;

cin >> value;

matrix2.setElement(i, j, value);

}

}

try {

cout << "Matrix 1:\n";

printMatrix(matrix1);

cout << "\nMatrix 2:\n";

printMatrix(matrix2);

cout << "\nMatrix Sum:\n";

Matrix sum = matrix1 + matrix2;

printMatrix(sum);

cout << "\nMatrix Product:\n";

Matrix product = matrix1 \* matrix2;

printMatrix(product);

cout << "\nMatrix 1 Transpose:\n";

Matrix transpose1 = matrix1.transpose();

printMatrix(transpose1);

cout << "\nMatrix 2 Transpose:\n";

Matrix transpose2 = matrix2.transpose();

printMatrix(transpose2);

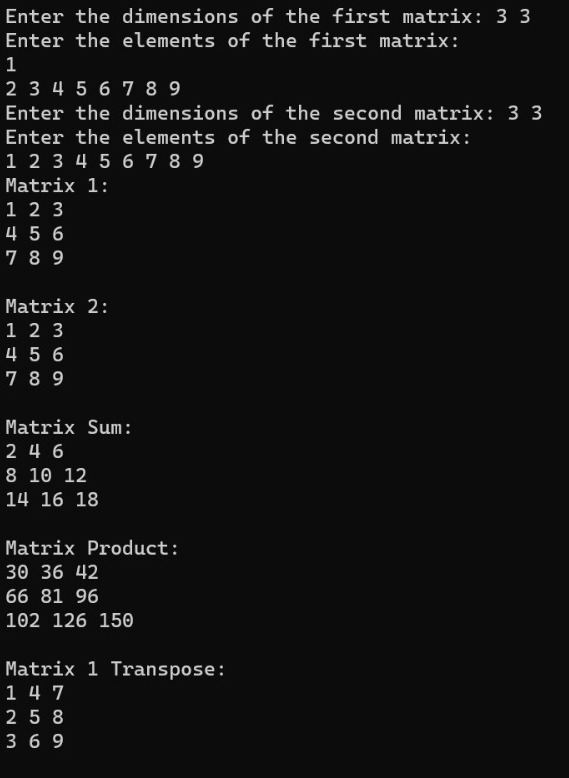
} catch (const exception& e) {

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

}

return 0;

}



9.Define a class Person having name as a data member. Inherit two classes Student and Employee from Person. Student has additional attributes as course, marks and year and Employee has department and salary. Write display() method in all the three classes to display the corresponding attributes. Provide the necessary methods to show runtime polymorphism.

#include<iosteam>

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;

double salary;

public:

Employee(const string& n, const string& d, double 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\* personPtr;

Student student("varsha tayal", "Computer Science", 90, 2023);

Employee employee("seema negi", "computer science", 5000.00);

personPtr = &student;

personPtr->display();

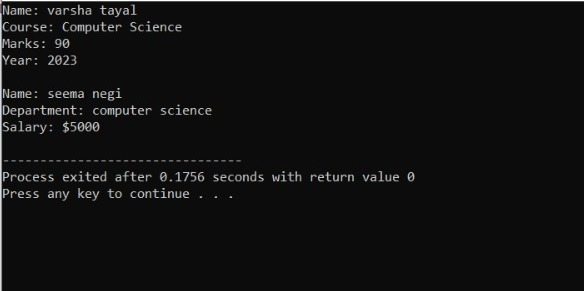
cout << endl;

personPtr = &employee;

personPtr->display();

return 0;

}



10. Create a Triangle class.  Add exception handling statements to ensure the following conditions: all sides are greater than 0 and sum of any two sides is greater than the third side.   The class should also have overloaded functions for calculating the area of a right angled triangle as well as using Heron's formula to calculate the area of any type of triangle.

#include <iostream>

#include <cmath>

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 runtime\_error("Invalid side length. All sides must be greater than 0.");

}

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

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

}

}

double calculateArea() const {

double s = (side1 + side2 + side3) / 2.0; // Semi-perimeter

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

}

double calculateArea(double base, double height) const {

return 0.5 \* base \* height; // Area of a right-angled triangle

}

};

int main() {

try {

double side1, side2, side3;

cout << "Enter the lengths of the sides of the triangle: ";

cin >> side1 >> side2 >> side3;

Triangle triangle(side1, side2, side3);

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

if (side1 > 0 && side2 > 0 && side3 > 0 && (side1 \* side1 + side2 \* side2 == side3 \* side3 || side1 \* side1 + side3 \* side3 == side2 \* side2 || side2 \* side2 + side3 \* side3 == side1 \* side1)) {

cout << "Enter the base and height of the right-angled triangle: ";

double base, height;

cin >> base >> height;

cout << "Area of the right-angled triangle: " << triangle.calculateArea(base, height) << endl;

} else {

cout << "The given triangle is not a right-angled triangle, so cannot calculate the area using the right-angled triangle formula." << endl;

}

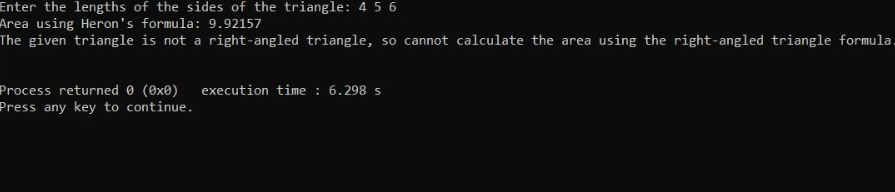
} catch (const exception& e) {

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

}

return 0;

}



11. Copy the contents of one text file to another file, after removing all whitespaces.

#include <iostream>

#include <fstream>

#include <string>

#include <cctype>

using namespace std;

// Function to remove whitespace characters from a string

string removeWhitespace(const string& str) {

string result;

for (char ch : str) {

if (!isspace(ch)) {

result += ch;

}

}

return result;

}

int main() {

string sourceFile, destinationFile;

cout << "Enter the name of the source file: ";

cin >> sourceFile;

cout << "Enter the name of the destination file: ";

cin >> destinationFile;

ifstream inputFile(sourceFile);

if (!inputFile) {

cout << "Error opening source file: " << sourceFile << endl;

return 1;

}

ofstream outputFile(destinationFile);

if (!outputFile) {

cout << "Error opening destination file: " << destinationFile << endl;

return 1;

}

string line;

while (getline(inputFile, line)) {

string lineWithoutWhitespace = removeWhitespace(line);

outputFile << lineWithoutWhitespace << endl;

}

cout << "File copied successfully!" << endl;

inputFile.close();

outputFile.close();

return 0;

}

