SSCBS

Practical File

DSC 04: Object Oriented Programming with C++

Prepared by :

Avinash Shrivastava (22512)

Cpp Practicals\Q1\Q1.cpp

```cpp
/*
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.
*/

#include <iostream>
#include <cmath>

using namespace std;

int main( int argc, char * argv[])
{
    int numberOfTerms;
    if (argc == 1){
        cout<<"Command line input not passed!"<<endl<<"Please Enter the number of terms ";
        cin>>numberOfTerms;
    }
    else
    {
        numberOfTerms = stoi(argv[1]);
    }

    cout<<"Entered number of terms : "<<numberOfTerms<<endl;
    float sumOfSeries = 0;
    for (int i = 1 ; i <= numberOfTerms ; i++)
    {
        sumOfSeries += pow(-1,i+1)/pow(i,i);
    }
    cout<<"Sum of the series till "<<numberOfTerms<<" terms is "<<sumOfSeries;

    return 0;

}


/*

Output :
case 1:
Command line input not passed!
Please Enter the number of terms 6
Entered number of terms : 6
Sum of the series till 6 terms is 0.783429

case 2:
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q1> ./Q1 '6'
Entered number of terms : 6
Sum of the series till 6 terms is 0.783429
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q1>

*/

```

Cpp Practicals\Q2\Q2.cpp

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

#include <iostream>
using namespace std;

int main()
{
    int arr[5] = {1,1,2,3,2};
    int uni[5] = {0,0,0,0,0};
    uni[0] = arr[0];
    int index = 1;
    bool found;
    for (int i = 1; i < 5; i++){
        found = false;
        for( int j = 0; j <i; j++ ){
            if (arr[i] == arr[j]){
                found = true;
                break;
            }
        }
        if (!found){
            uni[index] = arr[i];
            index++;
        }
    }
    for(int i = 0; i <5; i++){
        cout<<uni[i]<<" ";
    }
    return 0;
}

/*
Output :

PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q2\" ; if ($?) { g++ Q2.cpp -o Q2 } ; if ($?) { .\Q2 }
1 2 3 0 0

*/
```

Cpp Practicals\Q3\Q3.cpp

```cpp
/*
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>
using namespace std;
int noOfChar(string str, char ch)
{
    int count = 0;
    for (int i = 0; i < str.length(); i++)
    {
        if (ch == str[i])
        {
            count++;
        }
    }

    return count;
}

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

    string text = argv[1];
    string printedChar;
    cout<<"String : "<<text<<endl;
    cout<<"| char |occurance |"<<endl;
    for (int i = 0; i < text.length(); i++)
    {
        printedChar += text[i];

        if(noOfChar(printedChar, text[i]) ==1 )
        {
            cout<<"|  "<<text[i]<<"   |   "<<noOfChar(text, text[i])<<"      |"<<endl;
        }

    }

}


/*
Terminal output

PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q3> ./Q3.exe "apple"
String : apple
| char |occurance |
|  a   |   1      |
|  p   |   2      |
|  l   |   1      |
|  e   |   1      |
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q3>

*/
```

Cpp Practicals\Q4\Q4.cpp

```cpp
/*
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


*/

#include <iostream>
#include <string>
using namespace std;

void showAddress(string);  //a
string concatenate(string, string);  //b
void compare(string, string); //c
int stringLength(string); //d
string uppercase(string); //e
string reverse(string); //f
string insertString(string, string, int);//g

int main()
{
    char key;
    while (key != ' ')
    {
        cout<<"String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit"<<endl;
         cout<<"       a.   Show address of each character in string"<<endl;
         cout<<"       b.   Concatenate two strings. "<<endl;
         cout<<"       c.   Compare two strings "<<endl;
         cout<<"       d.   Calculate length of the string (use pointers) "<<endl;
         cout<<"       e.   Convert all lowercase characters to uppercase "<<endl;
         cout<<"       f.   Reverse the string "<<endl;
         cout<<"       g.   Insert a string in another string at a user specified position"<<endl;

        char response;
        cout<<"Enter your response : ";
        cin>> response;

        switch (response)
        {
            case 'a' :
            {
                string str;
                cout<<"Enter a string : ";
                cin>>str;
                showAddress(str);
                break;

            }
```

```
55
56              case 'b':
57              {
58                  string str1, str2;
59                  cout<<"Enter first string : ";
60                  cin.ignore();
61                  getline(cin, str1);
62                  cout<<"Enter second string : ";
63                  getline(cin, str2);
64                  string concinated = concatenate(str1, str2);
65                  cout<<concinated<<endl;
66                  break;
67
68              }
69
70              case 'c':
71              {
72                  string str1, str2;
73                  cout<<"Enter first string : ";
74                  cin.ignore();
75                  getline(cin, str1);
76                  cout<<"Enter second string : ";
77                  getline(cin, str2);
78                  compare(str1, str2);
79                  break;
80
81              }
82
83              case 'd':
84              {
85                  string str;
86                  cout<<"Enter a string : ";
87                  cin>>str;
88                  int len = stringLength(str);
89                  cout << len<<endl;
90                  break;
91              }
92
93              case 'e' :
94              {
95                  string str;
96                  cout<<"Enter a string : ";
97                  cin>>str;
98                  string upper_str = uppercase(str);
99                  cout <<upper_str<<endl;
100                 break;
101             }
102             case 'f':
103             {
104                 string str;
105                 cout<<"Enter a string : ";
106                 cin>>str;
107                 string reversed_str = reverse(str);
108                 cout<<reversed_str<<endl;
109                 break;
110             }
111             case 'g':
112             {
113                 string str1, str2;
114                 int pos;
```

```
                cout<<"Enter first string 1 : ";
                cin.ignore();
                getline(cin, str1);
                cout<<"Enter second string 2 : ";
                getline(cin, str2);
                cout<<"Enter position where you want to insert string 2 : ";
                cin>>pos;
                string newStr = insertString(str1, str2, pos);
                cout<<newStr<<endl;
                break;
            }
            default:
            {
                break;
            }
        }
    }
}

void showAddress(string str)
    {
        for(int i = 0; i < str.length(); i++)
        {
            cout<<"Position of "<<  str[i]<<": "<< (void*) str[i] <<endl;
        }
    }

string concatenate(string str1, string str2)
{
    string conc;
    conc = str1 + str2;
    return conc;
}

void compare(string str1, string str2)
{
    if(str1 > str2)
    {
        cout<<str1<<" > "<<str2<<endl;
    }
    else if (str1 < str2) {
        cout<<str2<<" < "<<str1<<endl;
    }
    else {
        cout<<str1<<" = "<<str2<<endl;
    }
}



int len(string &x)
{
    int count = 0;
    for (int i : x)
    {
        count++;
    }
    return(count);
}

```

```
int stringLength(string str)
{

    char *sptr;
    sptr=&str[0];

    int count=0;

    int i=0;
    while(*sptr!='\0'){
            sptr++;
            count++;

    }
    return count;
}


string uppercase(string str)
{
    string str_upper;
    for(int i = 0; i < str.length(); i++)
    {
        char letter = str[i];
        str_upper += toupper(letter);
    }
    return(str_upper);
}

string reverse(string str)
{
    string reversed_str;
    for(int i = 0; i < str.length(); i++)
    {
        char letter = str[i];
        reversed_str = letter + reversed_str;
    }
    return(reversed_str);
}

string insertString(string str1, string str2, int pos)
    {
    string newStr;
    for (int i = 0 ; i < pos; i ++)
    {
        newStr += str1[i];

    }
    newStr += str2;
    for (int i = pos ; i < str1.length(); i ++)
    {
        newStr += str1[i];
    }
    return(newStr);

    }


/*

```

```
Output :

String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : a
Enter a string : Apple
Position of A: 0x41
Position of p: 0x70
Position of p: 0x70
Position of l: 0x6c
Position of e: 0x65
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : b
Enter first string : Apple
Enter second string : Mango
AppleMango
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : c
Enter first string : Pizza
Enter second string : Burger
Pizza > Burger
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : d
Enter a string : Apple
5
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
```

```
Enter your response : e
Enter a string : Hello
HELLO
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : f
Enter a string : Avinash
hsanivA
String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
       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
Enter your response : g
Enter first string 1 : Avinash
Enter second string 2 : Shrivastava
Enter position where you want to insert string 2 : 7
AvinashShrivastava


*/
```

Cpp Practicals\Q5\Q5.cpp

```cpp
/*
5.  Write a program to merge two ordered arrays to get a single ordered array
*/


#include <iostream>
using namespace std;


void displayArray(int newarr[],int len);
int main()
{
    int arr1[] = {1,2,3,15,65};
    int arr2[] = {0,11,12,14};
    int len1 = sizeof(arr1)/sizeof(int);
    int len2 = sizeof(arr2)/sizeof(int);

    int newarr[len1+len2];
    for(int i = 0; i < len1; i++ )
    {
        newarr[i] = arr1[i];
    }
    for(int i = 0; i < len2; i++ )
    {
        newarr[i + len1] = arr2[i];
    }


    int n = sizeof(newarr)/sizeof(int);
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++)
        {
            if (newarr[j] > newarr[j+1])
            {
                int temp = newarr[j];
                newarr[j] = newarr[j+1];
                newarr[j+1] = temp;
            }
        }
    }
    cout<<"Orded Array 1 : "<<endl;
    displayArray(arr1,len1);
    cout<<"Orded Array 2 : "<<endl;
    displayArray(arr2,len2);
    cout<<"Orded Merged Array : "<<endl;
    displayArray(newarr,n);

}
void displayArray(int newarr[],int len)
{
    for(int i = 0; i < len; i++ )
    {
        cout<<newarr[i]<<" ";
    }
    cout<<endl;
}
```

```
57
58
59  /*
60
61  Output:
62
63  PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q5\" ; if ($?) { g++
    Q5.cpp -o Q5 } ; if ($?) { .\Q5 }
64  Orded Array 1 :
65  1 2 3 15 65
66  Orded Array 2 :
67  0 11 12 14
68  Orded Merged Array :
69  0 1 2 3 11 12 14 15 65
70  PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q5>
71
72  */
```

Cpp Practicals\Q6\Q6.cpp

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

#include <iostream>
using namespace std;

int main()
{
    int set[] = {1,2,3,5,81,7,8,9};
    int size = sizeof(set)/sizeof(int);
    int search_int;
    cout<<"Enter number to be searched : ";
    cin>>search_int;
    bool found = false;
    int pos;

    for (int i = 0; i<size; i ++ )
    {
        if (search_int == set[i])
        {
            found = true;
            pos = i;
            break;
        }
    }
    if (found)
    {
        cout <<search_int<<" found at "<<pos+1<<" position";
    }
    else
    {
        cout <<search_int<<" is not in the set";
    }
}


/*

Output :
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q5> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q6\" ; if ($?) { g++ Q6.cpp -o Q6 } ; if ($?) { .\Q6 }
Enter number to be searched : 45
45 is not in the set

PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q6> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q6\" ; if ($?) { g++ Q6.cpp -o Q6 } ; if ($?) { .\Q6 }
Enter number to be searched : 1
1 found at 1 position
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q6>
*/
```

Cpp Practicals\Q7\Q7.cpp

```cpp
/*
7.  Write a program to calculate GCD of two numbers.
*/

#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cout<<"Enter num 1 : ";
    cin>>a;
    cout<<"Enter num 2 : ";
    cin>>b;
    if (a<b)
    {
        while (b % a != 0)
        {
            a = b%a;
        }
        cout<<"Required GCD : "<<a;
    }
    else
    {
        while (a % b != 0)
        {
            b = a%b;
        }
        cout<<"Required GCD : "<<b;
    }


}

/*
Output:

PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q7\" ; if ($?) { g++ Q7.cpp -o Q7 } ; if ($?) { .\Q7 }
Enter num 1 : 66
Enter num 2 : 4
Required GCD : 2
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q7>

*/
```

Cpp Practicals\Q8\Q8.cpp

```cpp
/*
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

*/

#include <iostream>
#include <vector>
using namespace std;

class Matrix {
    int row, col;
    vector<vector<int>> arr;

public:
    Matrix() {}
    Matrix(int noOfRow, int noOfCol) : row(noOfRow), col(noOfCol), arr(noOfRow,
vector<int>(noOfCol, 0)) {}

    void inputMatrix() {
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                int element;
                cout << "Enter element at (" << i << "," << j << ") position: ";
                cin >> element;
                arr[i][j] = element;
            }
        }
    }

    void displayMatrix() const {
        for (int i = 0; i < row; i++) {
            for (int j = 0; j < col; j++) {
                cout << arr[i][j] << " ";
            }
            cout << endl;
        }
    }

    int getElement(int i, int j) const {
        return arr[i][j];
    }

    void setElement(int i, int j, int ele) {
        arr[i][j] = ele;
    }

    friend Matrix sum(const Matrix&, const Matrix&);
    friend Matrix product(const Matrix&, const Matrix&);
    friend Matrix transpose(const Matrix&);
};

```

```
Matrix sum(const Matrix& A, const Matrix& B) {
    int row = A.row;
    int col = A.col;
    Matrix C(row, col);
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            C.setElement(i, j, A.getElement(i, j) + B.getElement(i, j));
        }
    }
    return C;
}

Matrix product(const Matrix& A, const Matrix& B) {
    int rowA = A.row;
    int colA = A.col;
    int rowB = B.row;
    int colB = B.col;

    if (colA != rowB) {
        throw "Matrix dimensions are not compatible for multiplication!";
    }

    Matrix C(rowA, colB);

    for (int i = 0; i < rowA; i++) {
        for (int j = 0; j < colB; j++) {
            int sum = 0;
            for (int k = 0; k < colA; k++) {
                sum += A.getElement(i, k) * B.getElement(k, j);
            }
            C.setElement(i, j, sum);
        }
    }

    return C;
}

Matrix transpose(const Matrix& A) {
    int row = A.row;
    int col = A.col;
    Matrix C(col, row);
    for (int i = 0; i < col; i++) {
        for (int j = 0; j < row; j++) {
            C.setElement(i, j, A.getElement(j, i));
        }
    }
    return C;
}

int main() {
    int rows, cols;

    cout << "Enter the number of rows in the matrices: ";
    cin >> rows;
    cout << "Enter the number of columns in the matrices: ";
    cin >> cols;

    Matrix A(rows, cols);
    Matrix B(rows, cols);

```

```
    cout << "Enter the elements of the first matrix:" << endl;
    A.inputMatrix();
    cout << "Enter the elements of the second matrix:" << endl;
    B.inputMatrix();

    int choice;
    cout << "Select an operation:" << endl;
    cout << "1. Sum" << endl;
    cout << "2. Product" << endl;
    cout << "3. Transpose" << endl;
    cout << "Enter your choice (1-3): ";
    cin >> choice;

    Matrix result;
    try {
        switch (choice) {
            case 1:
                result = sum(A, B);
                cout << "Sum of the matrices:" << endl;
                result.displayMatrix();
                break;
            case 2:
                result = product(A, B);
                cout << "Product of the matrices:" << endl;
                result.displayMatrix();
                break;
            case 3:
                result = transpose(A);
                cout << "Transpose of the matrix:" << endl;
                result.displayMatrix();
                break;
            default:
                cout << "Invalid choice!" << endl;
        }
    } catch (const char* errorMessage) {
        cout << "Error: " << errorMessage << endl;
    }

    return 0;
}


/*
Output:

PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q8> cd "c:\Users\hp\Desktop\Cpp\Cpp
Practicals\Q8\" ; if ($?) { g++ Q8.cpp -o Q8 } ; if ($?) { .\Q8 }
Enter the number of rows in the matrices: 2
Enter the number of columns in the matrices: 2
Enter the elements of the first matrix:
Enter element at (0,0) position: 1
Enter element at (0,1) position: 1
Enter element at (1,0) position: 1
Enter element at (1,1) position: 1
Enter the elements of the second matrix:
Enter element at (0,0) position: 2
Enter element at (0,1) position: 2
Enter element at (1,0) position: 2
Enter element at (1,1) position: 2
Select an operation:
```

```
1. Sum
2. Product
3. Transpose
Enter your choice (1-3): 1
Sum of the matrices:
3 3
3 3

Select an operation:
1. Sum
2. Product
3. Transpose
Enter your choice (1-3): 2
Product of the matrices:
4 4
4 4

Select an operation:
1. Sum
2. Product
3. Transpose
Enter your choice (1-3): 3
Transpose of the matrix:
1 1
1 1
*/

```

Cpp Practicals\Q9\Q9.cpp

```cpp
// 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 <iostream>
using namespace std;

class Person
{   protected:
    string name;

    public:
    Person(string Pname)
    {
        name = Pname;
    }
        virtual void display(void)
        {
            cout<<"Name : "<<name<<endl;
        }
};


class Student : public Person
{
    string course;
    float marks;
    int year;

    public:
    Student(string name, string Mcourse, float mark, int yrs) : Person(name)
    {
        course = Mcourse;
        marks = mark;
        year = yrs;
    }
        void display(void)
        {
            cout<<"Name : "<<name<<endl;
            cout<<"Course : "<<course<<endl;
            cout<<"Marks : "<<marks<<endl;
            cout<<"Year : "<<year<<endl;
        }

};

class Employee : public Person
{
    string department;
    float salary;

    public:
        Employee(string Ename, string dept, float sal) : Person(Ename)
```

```cpp
        {
            department = dept;
            salary = sal;
        }
        void display(void)
        {
            cout<<"Name : "<<name<<endl;
            cout<<"Department : "<<department<<endl;
            cout<<"Salary : "<<salary<<endl;
        }
};

int main()
{
    Person * perPtr;
    Student S1("Ravi","CS",123,2023);
    Employee E1("Anshu", "Tech",900000);
    perPtr = &E1;
    cout<<"Employee's details :"<<endl;
    perPtr->display();
    perPtr = &S1;
    cout<<endl;
    cout<<"Student's details :"<<endl;
    perPtr->display();

    return 0;
}



/*
Output :

PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q8> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q9\" ; if ($?) { g++ Q9.cpp -o Q9 } ; if ($?) { .\Q9 }
Employee's details :
Name : Anshu
Department : Tech
Salary : 900000

Student's details :
Name : Ravi
Course : CS
Marks : 123
Year : 2023
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q9>

*/
```

Cpp Practicals\Q10\Q10.cpp

```cpp

/*
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>
#include <cstring>
using namespace std;
class Error {
    int err_code;
    string err_desc;

public:
    Error(int c, string errMsg)
    {
        err_code = c;
        err_desc = errMsg;
    }

    void err_display(void)
    {
        cout << "Error Code: " << err_code << endl << "Error Description: " << err_desc << endl;
    }
};

class Triangle
{
    float side1, side2, side3;
    public :
    Triangle(){}
    Triangle(float a, float b, float c)
    {
        try
        {
            if(a<= 0 || b <=0 || c <= 0)
            {
                throw Error(001,"Sides cannot be negative or 0!");
            }
            if(a >= b + c || b >= a + c || c >= a + b)
            {
                throw Error(002,"Either of side exceeds the sum of other two sides!");
            }
            side1 = a;
            side2 = b;
            side3 = c;

        }
        catch (Error e)
        {
            e.err_display();
```

```
        }


    }
    float area()
    {
        float s = (side1 + side2 + side3)/2;
        float area = sqrt(s*(s-side1)*(s-side2)*(s-side3));
        return area;
    }
    float area(float base, float height)
    {
        try
        {
             float area = (base * height)/2;
             if( area == 0)
             {
                throw Error(003, "Invalid Base or Height of Right triangle");
             }
             return area;

        }
        catch( Error e)
        {
            e.err_display();
        }
    }
};



int main()
{
    Triangle DEF(0,3,4);
    Triangle ABC(3, 4, 5);
    float area = ABC.area();
    cout<<"Area of general Trianle ABC is "<< area<<endl;
    Triangle PQR;
    float rArea = PQR.area(4,6);
    cout<<"Area of Right angled Trianle ABC is "<< rArea<<endl;


}

/*
Error Code: 1
Error Description: Sides cannot be negative or 0!
Area of general Trianle ABC is 6
Area of Right angled Trianle ABC is 12
PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q10>
*/
```

Cpp Practicals\Q11\Q11.cpp

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

#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    ifstream file("textfile.txt");
    ofstream fileCopy("copiedfile.txt");
    string line;
    while (file)
    {
        getline(file, line);
        string copyLine;
        for(int i = 0; i <line.length(); i++)
        {
            if(line[i] != ' ')
            {
                copyLine += line[i];
            }
        }
        fileCopy<<copyLine<<endl;
    }
}


```

Cpp Practicals\Q11\textfile.txt

```
This document is meant to be copied removing blanks spaces;
This is part of Q11
```

Cpp Practicals\Q11\copiedfile.txt

```
Thisdocumentismeanttobecopiedremovingblanksspaces;
ThisispartofQ11
```

Cpp Practicals\Thank you.txt

Thank you for spending time and going through these programs!

You can find all these practicals on my Github profile at this link :
https://github.com/AvinashShrivastav/Cpp/tree/main/Cpp%20Practicals