



SSCBS



# Practical File

DSC 04: Object Oriented  
Programming with C++



Prepared by :

**Avinash Shrivastava**

**(22512)**



## Cpp Practicals\Q1\Q1.cpp

```
1  /*
2  1. Write a program to compute the sum of the first n terms of the following series:
3   $S=1-1/2^2 +1/3^3 -... \quad 1/n^n$ 
4  The number of terms n is to be taken from the user through the command line. If the
5  command line argument is not found then prompt the user to enter the value of n.
6  */
7
8  #include <iostream>
9  #include <cmath>
10
11 using namespace std;
12
13 int main( int argc, char * argv[])
14 {
15     int numberOfTerms;
16     if (argc == 1){
17         cout<<"Command line input not passed!"<<endl<<"Please Enter the number of terms ";
18         cin>>numberOfTerms;
19     }
20     else
21     {
22         numberOfTerms = stoi(argv[1]);
23     }
24
25     cout<<"Entered number of terms : "<<numberOfTerms<<endl;
26     float sumOfSeries = 0;
27     for (int i = 1 ; i <= numberOfTerms ; i++)
28     {
29         sumOfSeries += pow(-1,i+1)/pow(i,i);
30     }
31     cout<<"Sum of the series till "<<numberOfTerms<<" terms is "<<sumOfSeries;
32
33     return 0;
34 }
35
36
37
38 /*
39
40 Output :
41 case 1:
42 Command line input not passed!
43 Please Enter the number of terms 6
44 Entered number of terms : 6
45 Sum of the series till 6 terms is 0.783429
46
47 case 2:
48 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q1> ./Q1 '6'
49 Entered number of terms : 6
50 Sum of the series till 6 terms is 0.783429
51 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q1>
52
53 */
54
```

## Cpp Practicals\Q2\Q2.cpp

```
1 // 2. Write a program to remove the duplicates from an array.
2
3 #include <iostream>
4 using namespace std;
5
6 int main()
7 {
8     int arr[5] = {1,1,2,3,2};
9     int uni[5] = {0,0,0,0,0};
10    uni[0] = arr[0];
11    int index = 1;
12    bool found;
13    for (int i = 1; i < 5; i++){
14        found = false;
15        for( int j = 0; j < i; j++ ){
16            if (arr[i] == arr[j]){
17                found = true;
18                break;
19            }
20        }
21        if (!found){
22            uni[index] = arr[i];
23            index++;
24        }
25    }
26    for(int i = 0; i < 5; i++){
27        cout<<uni[i]<<" ";
28    }
29    return 0;
30 }
31
32 /*
33 Output :
34
35 PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q2\" ; if ($?) { g++
36 Q2.cpp -o Q2 } ; if ($?) { .\Q2 }
37 1 2 3 0 0
38 */
```

## Cpp Practicals\Q3\Q3.cpp

```
1  /*
2  3. Write a program that prints a table indicating the number of
3  occurrences of each alphabet in the text entered as command line arguments.
4
5  */
6  #include <iostream>
7  #include <string>
8  using namespace std;
9  int noOfChar(string str, char ch)
10 {
11     int count = 0;
12     for (int i = 0; i < str.length(); i++)
13     {
14         if (ch == str[i])
15         {
16             count++;
17         }
18     }
19
20     return count;
21 }
22
23 int main(int argc, char *argv[])
24 {
25
26     string text = argv[1];
27     string printedChar;
28     cout<<"String : "<<text<<endl;
29     cout<<"| char |occurance |"<<endl;
30     for (int i = 0; i < text.length(); i++)
31     {
32         printedChar += text[i];
33
34         if(noOfChar(printedChar, text[i]) ==1 )
35         {
36             cout<<"|   "<<text[i]<<"   |   "<<noOfChar(text, text[i])<<"   |"<<endl;
37         }
38
39     }
40
41 }
42
43
44 /*
45 Terminal output
46
47 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q3> ./Q3.exe "apple"
48 String : apple
49 | char |occurance |
50 |  a   |    1     |
51 |  p   |    2     |
52 |  l   |    1     |
53 |  e   |    1     |
54 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q3>
55
56 */
```

## Cpp Practicals\Q4\Q4.cpp

```
1  /*
2  4. Write a menu driven program to perform string manipulation (without using inbuilt
   string functions):
3      a. Show address of each character in string
4      b. Concatenate two strings.
5      c. Compare two strings
6      d. Calculate length of the string (use pointers)
7      e. Convert all lowercase characters to uppercase
8      f. Reverse the string
9      g. Insert a string in another string at a user specified position
10
11
12 */
13
14 #include <iostream>
15 #include <string>
16 using namespace std;
17
18 void showAddress(string); //a
19 string concatenate(string, string); //b
20 void compare(string, string); //c
21 int stringLength(string); //d
22 string uppercase(string); //e
23 string reverse(string); //f
24 string insertString(string, string, int); //g
25
26 int main()
27 {
28     char key;
29     while (key != ' ')
30     {
31         cout<<"String Manipulation Program : Press a-g to manipulate strings, press
   spacebar to exit"<<endl;
32         cout<<"      a. Show address of each character in string"<<endl;
33         cout<<"      b. Concatenate two strings. "<<endl;
34         cout<<"      c. Compare two strings "<<endl;
35         cout<<"      d. Calculate length of the string (use pointers) "<<endl;
36         cout<<"      e. Convert all lowercase characters to uppercase "<<endl;
37         cout<<"      f. Reverse the string "<<endl;
38         cout<<"      g. Insert a string in another string at a user specified
   position"<<endl;
39
40         char response;
41         cout<<"Enter your response : ";
42         cin>> response;
43
44         switch (response)
45         {
46             case 'a' :
47             {
48                 string str;
49                 cout<<"Enter a string : ";
50                 cin>>str;
51                 showAddress(str);
52                 break;
53
54             }
```

```

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

```

```

115         cout<<"Enter first string 1 : ";
116         cin.ignore();
117         getline(cin, str1);
118         cout<<"Enter second string 2 : ";
119         getline(cin, str2);
120         cout<<"Enter position where you want to insert string 2 : ";
121         cin>>pos;
122         string newStr = insertString(str1, str2, pos);
123         cout<<newStr<<endl;
124         break;
125     }
126     default:
127     {
128         break;
129     }
130 }
131 }
132 }
133
134 void showAddress(string str)
135 {
136     for(int i = 0; i < str.length(); i++)
137     {
138         cout<<"Position of "<< str[i]<<": "<< (void*) str[i] <<endl;
139     }
140 }
141
142 string concatenate(string str1, string str2)
143 {
144     string conc;
145     conc = str1 + str2;
146     return conc;
147 }
148
149 void compare(string str1, string str2)
150 {
151     if(str1 > str2)
152     {
153         cout<<str1<<" > "<<str2<<endl;
154     }
155     else if (str1 < str2) {
156         cout<<str2<<" < "<<str1<<endl;
157     }
158     else {
159         cout<<str1<<" = "<<str2<<endl;
160     }
161 }
162
163
164
165 int len(string &x)
166 {
167     int count = 0;
168     for (int i : x)
169     {
170         count++;
171     }
172     return(count);
173 }
174

```

```

175 int stringLength(string str)
176 {
177
178     char *sptr;
179     sptr=&str[0];
180
181     int count=0;
182
183     int i=0;
184     while(*sptr!='\0'){
185         sptr++;
186         count++;
187
188     }
189     return count;
190 }
191
192
193 string uppercase(string str)
194 {
195     string str_upper;
196     for(int i = 0; i < str.length(); i++)
197     {
198         char letter = str[i];
199         str_upper += toupper(letter);
200     }
201     return(str_upper);
202 }
203
204 string reverse(string str)
205 {
206     string reversed_str;
207     for(int i = 0; i < str.length(); i++)
208     {
209         char letter = str[i];
210         reversed_str = letter + reversed_str;
211     }
212     return(reversed_str);
213 }
214
215 string insertString(string str1, string str2, int pos)
216 {
217     string newStr;
218     for (int i = 0 ; i < pos; i ++ )
219     {
220         newStr += str1[i];
221     }
222     newStr += str2;
223     for (int i = pos ; i < str1.length(); i ++ )
224     {
225         newStr += str1[i];
226     }
227     return(newStr);
228
229
230 }
231
232
233 /*
234

```



```
235 Output :
236
237 String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
238     a.      Show address of each character in string
239     b.      Concatenate two strings.
240     c.      Compare two strings
241     d.      Calculate length of the string (use pointers)
242     e.      Convert all lowercase characters to uppercase
243     f.      Reverse the string
244     g.      Insert a string in another string at a user specified position
245 Enter your response : a
246 Enter a string : Apple
247 Position of A: 0x41
248 Position of p: 0x70
249 Position of p: 0x70
250 Position of l: 0x6c
251 Position of e: 0x65
252 String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
253     a.      Show address of each character in string
254     b.      Concatenate two strings.
255     c.      Compare two strings
256     d.      Calculate length of the string (use pointers)
257     e.      Convert all lowercase characters to uppercase
258     f.      Reverse the string
259     g.      Insert a string in another string at a user specified position
260 Enter your response : b
261 Enter first string : Apple
262 Enter second string : Mango
263 AppleMango
264 String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
265     a.      Show address of each character in string
266     b.      Concatenate two strings.
267     c.      Compare two strings
268     d.      Calculate length of the string (use pointers)
269     e.      Convert all lowercase characters to uppercase
270     f.      Reverse the string
271     g.      Insert a string in another string at a user specified position
272 Enter your response : c
273 Enter first string : Pizza
274 Enter second string : Burger
275 Pizza > Burger
276 String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
277     a.      Show address of each character in string
278     b.      Concatenate two strings.
279     c.      Compare two strings
280     d.      Calculate length of the string (use pointers)
281     e.      Convert all lowercase characters to uppercase
282     f.      Reverse the string
283     g.      Insert a string in another string at a user specified position
284 Enter your response : d
285 Enter a string : Apple
286 5
287 String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
288     a.      Show address of each character in string
289     b.      Concatenate two strings.
290     c.      Compare two strings
291     d.      Calculate length of the string (use pointers)
292     e.      Convert all lowercase characters to uppercase
293     f.      Reverse the string
294     g.      Insert a string in another string at a user specified position
```

```
295 | Enter your response : e
296 | Enter a string : Hello
297 | HELLO
298 | String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
299 |     a.      Show address of each character in string
300 |     b.      Concatenate two strings.
301 |     c.      Compare two strings
302 |     d.      Calculate length of the string (use pointers)
303 |     e.      Convert all lowercase characters to uppercase
304 |     f.      Reverse the string
305 |     g.      Insert a string in another string at a user specified position
306 | Enter your response : f
307 | Enter a string : Avinash
308 | hsanivA
309 | String Manipulation Program : Press a-g to manipulate strings, press spacebar to exit
310 |     a.      Show address of each character in string
311 |     b.      Concatenate two strings.
312 |     c.      Compare two strings
313 |     d.      Calculate length of the string (use pointers)
314 |     e.      Convert all lowercase characters to uppercase
315 |     f.      Reverse the string
316 |     g.      Insert a string in another string at a user specified position
317 | Enter your response : g
318 | Enter first string 1 : Avinash
319 | Enter second string 2 : Shrivastava
320 | Enter position where you want to insert string 2 : 7
321 | AvinashShrivastava
322 |
323 |
324 | */
```

## Cpp Practicals\Q5\Q5.cpp

```
1  /*
2  5. Write a program to merge two ordered arrays to get a single ordered array
3  */
4
5
6  #include <iostream>
7  using namespace std;
8
9
10 void displayArray(int newarr[],int len);
11 int main()
12 {
13     int arr1[] = {1,2,3,15,65};
14     int arr2[] = {0,11,12,14};
15     int len1 = sizeof(arr1)/sizeof(int);
16     int len2 = sizeof(arr2)/sizeof(int);
17
18     int newarr[len1+len2];
19     for(int i = 0; i < len1; i++ )
20     {
21         newarr[i] = arr1[i];
22     }
23     for(int i = 0; i < len2; i++ )
24     {
25         newarr[i + len1] = arr2[i];
26     }
27
28
29     int n = sizeof(newarr)/sizeof(int);
30     for (int i = 0; i < n - 1; i++) {
31         for (int j = 0; j < n - i - 1; j++)
32         {
33             if (newarr[j] > newarr[j+1])
34             {
35                 int temp = newarr[j];
36                 newarr[j] = newarr[j+1];
37                 newarr[j+1] = temp;
38             }
39         }
40     }
41     cout<<"Orded Array 1 : "<<endl;
42     displayArray(arr1,len1);
43     cout<<"Orded Array 2 : "<<endl;
44     displayArray(arr2,len2);
45     cout<<"Orded Merged Array : "<<endl;
46     displayArray(newarr,n);
47
48 }
49 void displayArray(int newarr[],int len)
50 {
51     for(int i = 0; i < len; i++ )
52     {
53         cout<<newarr[i]<<" ";
54     }
55     cout<<endl;
56 }
```

```
57 |
58 |
59 | /*
60 |
61 | Output:
62 |
63 | PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q5\" ; if ($?) { g++
64 | Q5.cpp -o Q5 } ; if ($?) { .\Q5 }
65 |
66 | Orded Array 1 :
67 | 1 2 3 15 65
68 | Orded Array 2 :
69 | 0 11 12 14
70 | Orded Merged Array :
71 | 0 1 2 3 11 12 14 15 65
72 | PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q5>
73 | */
```

## Cpp Practicals\Q6\Q6.cpp

```
1 // 6. Write a program to search a given element in a set of N numbers.
2
3 #include <iostream>
4 using namespace std;
5
6 int main()
7 {
8     int set[] = {1,2,3,5,81,7,8,9};
9     int size = sizeof(set)/sizeof(int);
10    int search_int;
11    cout<<"Enter number to be searched : ";
12    cin>>search_int;
13    bool found = false;
14    int pos;
15
16    for (int i = 0; i<size; i ++ )
17    {
18        if (search_int == set[i])
19        {
20            found = true;
21            pos = i;
22            break;
23        }
24    }
25    if (found)
26    {
27        cout <<search_int<<" found at "<<pos+1<<" position";
28    }
29    else
30    {
31        cout <<search_int<<" is not in the set";
32    }
33 }
34
35
36 /*
37
38 Output :
39 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 }
40 Enter number to be searched : 45
41 45 is not in the set
42
43 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 }
44 Enter number to be searched : 1
45 1 found at 1 position
46 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q6>
47 */
```

## Cpp Practicals\Q7\Q7.cpp

```
1  /*
2  7. Write a program to calculate GCD of two numbers.
3  */
4
5  #include <iostream>
6  using namespace std;
7  int main()
8  {
9      int a,b;
10     cout<<"Enter num 1 : ";
11     cin>>a;
12     cout<<"Enter num 2 : ";
13     cin>>b;
14     if (a<b)
15     {
16         while (b % a != 0)
17         {
18             a = b%a;
19         }
20         cout<<"Required GCD : "<<a;
21     }
22     else
23     {
24         while (a % b != 0)
25         {
26             b = a%b;
27         }
28         cout<<"Required GCD : "<<b;
29     }
30
31
32 }
33
34 /*
35 Output:
36
37 PS C:\Users\hp\Desktop\Cpp> cd "c:\Users\hp\Desktop\Cpp\Cpp Practicals\Q7\" ; if ($?) { g++
38 Q7.cpp -o Q7 } ; if ($?) { .\Q7 }
39 Enter num 1 : 66
40 Enter num 2 : 4
41 Required GCD : 2
42 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q7>
43
44 */
45
46
47
```

## Cpp Practicals\Q8\Q8.cpp

```
1  /*
2  8. Create a Matrix class. Write a menu-driven program to perform following Matrix
3  operations (exceptions should be thrown by the functions if matrices passed to them
4  are incompatible and handled by the main() function):
5  a. Sum
6  b. Product
7  c. Transpose
8
9  */
10
11 #include <iostream>
12 #include <vector>
13 using namespace std;
14
15 class Matrix {
16     int row, col;
17     vector<vector<int>> arr;
18
19 public:
20     Matrix() {}
21     Matrix(int noOfRow, int noOfCol) : row(noOfRow), col(noOfCol), arr(noOfRow,
vector<int>(noOfCol, 0)) {}
22
23     void inputMatrix() {
24         for (int i = 0; i < row; i++) {
25             for (int j = 0; j < col; j++) {
26                 int element;
27                 cout << "Enter element at (" << i << ", " << j << ") position: ";
28                 cin >> element;
29                 arr[i][j] = element;
30             }
31         }
32     }
33
34     void displayMatrix() const {
35         for (int i = 0; i < row; i++) {
36             for (int j = 0; j < col; j++) {
37                 cout << arr[i][j] << " ";
38             }
39             cout << endl;
40         }
41     }
42
43     int getElement(int i, int j) const {
44         return arr[i][j];
45     }
46
47     void setElement(int i, int j, int ele) {
48         arr[i][j] = ele;
49     }
50
51     friend Matrix sum(const Matrix&, const Matrix&);
52     friend Matrix product(const Matrix&, const Matrix&);
53     friend Matrix transpose(const Matrix&);
54 };
55
```

```

56 Matrix sum(const Matrix& A, const Matrix& B) {
57     int row = A.row;
58     int col = A.col;
59     Matrix C(row, col);
60     for (int i = 0; i < row; i++) {
61         for (int j = 0; j < col; j++) {
62             C.setElement(i, j, A.getElement(i, j) + B.getElement(i, j));
63         }
64     }
65     return C;
66 }
67
68 Matrix product(const Matrix& A, const Matrix& B) {
69     int rowA = A.row;
70     int colA = A.col;
71     int rowB = B.row;
72     int colB = B.col;
73
74     if (colA != rowB) {
75         throw "Matrix dimensions are not compatible for multiplication!";
76     }
77
78     Matrix C(rowA, colB);
79
80     for (int i = 0; i < rowA; i++) {
81         for (int j = 0; j < colB; j++) {
82             int sum = 0;
83             for (int k = 0; k < colA; k++) {
84                 sum += A.getElement(i, k) * B.getElement(k, j);
85             }
86             C.setElement(i, j, sum);
87         }
88     }
89
90     return C;
91 }
92
93 Matrix transpose(const Matrix& A) {
94     int row = A.row;
95     int col = A.col;
96     Matrix C(col, row);
97     for (int i = 0; i < col; i++) {
98         for (int j = 0; j < row; j++) {
99             C.setElement(i, j, A.getElement(j, i));
100         }
101     }
102     return C;
103 }
104
105 int main() {
106     int rows, cols;
107
108     cout << "Enter the number of rows in the matrices: ";
109     cin >> rows;
110     cout << "Enter the number of columns in the matrices: ";
111     cin >> cols;
112
113     Matrix A(rows, cols);
114     Matrix B(rows, cols);
115

```



```

116     cout << "Enter the elements of the first matrix:" << endl;
117     A.inputMatrix();
118     cout << "Enter the elements of the second matrix:" << endl;
119     B.inputMatrix();
120
121     int choice;
122     cout << "Select an operation:" << endl;
123     cout << "1. Sum" << endl;
124     cout << "2. Product" << endl;
125     cout << "3. Transpose" << endl;
126     cout << "Enter your choice (1-3): ";
127     cin >> choice;
128
129     Matrix result;
130     try {
131         switch (choice) {
132             case 1:
133                 result = sum(A, B);
134                 cout << "Sum of the matrices:" << endl;
135                 result.displayMatrix();
136                 break;
137             case 2:
138                 result = product(A, B);
139                 cout << "Product of the matrices:" << endl;
140                 result.displayMatrix();
141                 break;
142             case 3:
143                 result = transpose(A);
144                 cout << "Transpose of the matrix:" << endl;
145                 result.displayMatrix();
146                 break;
147             default:
148                 cout << "Invalid choice!" << endl;
149         }
150     } catch (const char* errorMessage) {
151         cout << "Error: " << errorMessage << endl;
152     }
153
154     return 0;
155 }
156
157
158 /*
159 Output:
160
161 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 }
162 Enter the number of rows in the matrices: 2
163 Enter the number of columns in the matrices: 2
164 Enter the elements of the first matrix:
165 Enter element at (0,0) position: 1
166 Enter element at (0,1) position: 1
167 Enter element at (1,0) position: 1
168 Enter element at (1,1) position: 1
169 Enter the elements of the second matrix:
170 Enter element at (0,0) position: 2
171 Enter element at (0,1) position: 2
172 Enter element at (1,0) position: 2
173 Enter element at (1,1) position: 2
174 Select an operation:

```

```
175 | 1. Sum
176 | 2. Product
177 | 3. Transpose
178 | Enter your choice (1-3): 1
179 | Sum of the matrices:
180 | 3 3
181 | 3 3
182 |
183 | Select an operation:
184 | 1. Sum
185 | 2. Product
186 | 3. Transpose
187 | Enter your choice (1-3): 2
188 | Product of the matrices:
189 | 4 4
190 | 4 4
191 |
192 | Select an operation:
193 | 1. Sum
194 | 2. Product
195 | 3. Transpose
196 | Enter your choice (1-3): 3
197 | Transpose of the matrix:
198 | 1 1
199 | 1 1
200 | */
201 |
```

## Cpp Practicals\Q9\Q9.cpp

```
1 // 9. Define a class Person having name as a data member.
2 // Inherit two classes Student and Employee from Person.
3 // Student has additional attributes as course, marks and year and
4 // Employee has department and salary. Write display() method in
5 // all the three classes to display the corresponding attributes.
6 // Provide the necessary methods to show runtime polymorphism.
7
8 #include <iostream>
9 using namespace std;
10
11 class Person
12 { protected:
13     string name;
14
15     public:
16     Person(string Pname)
17     {
18         name = Pname;
19     }
20     virtual void display(void)
21     {
22         cout<<"Name : "<<name<<endl;
23     }
24 };
25
26
27 class Student : public Person
28 {
29     string course;
30     float marks;
31     int year;
32
33     public:
34     Student(string name, string Mcourse, float mark, int yrs) : Person(name)
35     {
36         course = Mcourse;
37         marks = mark;
38         year = yrs;
39     }
40     void display(void)
41     {
42         cout<<"Name : "<<name<<endl;
43         cout<<"Course : "<<course<<endl;
44         cout<<"Marks : "<<marks<<endl;
45         cout<<"Year : "<<year<<endl;
46     }
47 };
48
49
50 class Employee : public Person
51 {
52     string department;
53     float salary;
54
55     public:
56     Employee(string Ename, string dept, float sal) : Person(Ename)
```

```

57     {
58         department = dept;
59         salary = sal;
60     }
61     void display(void)
62     {
63         cout<<"Name : "<<name<<endl;
64         cout<<"Department : "<<department<<endl;
65         cout<<"Salary : "<<salary<<endl;
66     }
67 };
68
69 int main()
70 {
71     Person * perPtr;
72     Student S1("Ravi","CS",123,2023);
73     Employee E1("Anshu", "Tech",900000);
74     perPtr = &E1;
75     cout<<"Employee's details : "<<endl;
76     perPtr->display();
77     perPtr = &S1;
78     cout<<endl;
79     cout<<"Student's details : "<<endl;
80     perPtr->display();
81
82     return 0;
83 }
84
85
86
87 /*
88 Output :
89
90 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 }
91 Employee's details :
92 Name : Anshu
93 Department : Tech
94 Salary : 900000
95
96 Student's details :
97 Name : Ravi
98 Course : CS
99 Marks : 123
100 Year : 2023
101 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q9>
102
103 */

```

## Cpp Practicals\Q10\Q10.cpp

```
1
2  /*
3  10. Create a Triangle class. Add exception handling
4  statements to ensure the following conditions: all
5  sides are greater than 0 and sum of any two sides is greater than the
6  third side. The class should also have overloaded functions for calculating
7  the area of a right angled triangle as well as using Heron's formula to calculate
8  the area of any type of triangle.
9  */
10 #include <iostream>
11 #include <cmath>
12 #include <cstring>
13 using namespace std;
14 class Error {
15     int err_code;
16     string err_desc;
17
18 public:
19     Error(int c, string errMsg)
20     {
21         err_code = c;
22         err_desc = errMsg;
23     }
24
25     void err_display(void)
26     {
27         cout << "Error Code: " << err_code << endl << "Error Description: " << err_desc <<
endl;
28     }
29 };
30
31 class Triangle
32 {
33     float side1, side2, side3;
34 public :
35     Triangle(){}
36     Triangle(float a, float b, float c)
37     {
38         try
39         {
40             if(a<= 0 || b <=0 || c <= 0)
41             {
42                 throw Error(001,"Sides cannot be negative or 0!");
43             }
44             if(a >= b + c || b >= a + c || c >= a + b)
45             {
46                 throw Error(002,"Either of side exceeds the sum of other two sides!");
47             }
48             side1 = a;
49             side2 = b;
50             side3 = c;
51
52         }
53         catch (Error e)
54         {
55             e.err_display();
56         }
57     }
58 }
```

```

56     }
57
58
59 }
60 float area()
61 {
62     float s = (side1 + side2 + side3)/2;
63     float area = sqrt(s*(s-side1)*(s-side2)*(s-side3));
64     return area;
65 }
66 float area(float base, float height)
67 {
68     try
69     {
70         float area = (base * height)/2;
71         if( area == 0)
72         {
73             throw Error(003, "Invalid Base or Height of Right triangle");
74         }
75         return area;
76     }
77     catch( Error e)
78     {
79         e.err_display();
80     }
81 }
82 };
83
84
85
86
87
88 int main()
89 {
90     Triangle DEF(0,3,4);
91     Triangle ABC(3, 4, 5);
92     float area = ABC.area();
93     cout<<"Area of general Trianle ABC is "<< area<<endl;
94     Triangle PQR;
95     float rArea = PQR.area(4,6);
96     cout<<"Area of Right angled Trianle ABC is "<< rArea<<endl;
97
98
99 }
100
101 /*
102 Error Code: 1
103 Error Description: Sides cannot be negative or 0!
104 Area of general Trianle ABC is 6
105 Area of Right angled Trianle ABC is 12
106 PS C:\Users\hp\Desktop\Cpp\Cpp Practicals\Q10>
107 */

```

## Cpp Practicals\Q11\Q11.cpp

```
1 // 11. Copy the contents of one text file to another file, after removing all whitespaces.
2
3 #include <iostream>
4 #include <fstream>
5 using namespace std;
6
7 int main()
8 {
9     ifstream file("textfile.txt");
10    ofstream fileCopy("copiedfile.txt");
11    string line;
12    while (file)
13    {
14        getline(file, line);
15        string copyLine;
16        for(int i = 0; i < line.length(); i++)
17        {
18            if(line[i] != ' ')
19            {
20                copyLine += line[i];
21            }
22        }
23        fileCopy<<copyLine<<endl;
24    }
25 }
26
27
28
```

## **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

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

## **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>