

Institute of Space Technology Islamabad

Assignment 02,03



Submitted to:

Ma'am Tayyiba

Submitted by:

Muhammad Bin Nasir

220201012

Question 01

In this task, we implement multiple inheritance. We create two classes, a 'Mammal' and a 'Bird' class. Then we create another class which inherits both the classes, we name the third class 'Organism'. Consider including the following attributes and functionalities to your classes.

classMammal

- voidsetMammalName(string);
- voidshowMammal();

classBird

- voidsetBirdName(string);
- voidshowBird();

classOrganism

- voidsetOrganismName(string);
- voidshowOrganism();
- char* getOrganismName();

Code:

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class Mammal
6  {
7      string MammalName;
8
9      public:
10     Mammal()
11     {
12         MammalName = "Null";
13     }
14     string getMammalName()
15     {
16         return MammalName;
17     }
18     void setMammalName(string Name)
19     {
20         MammalName = Name;
21     }
22     void showMammal()
23     {
24         cout<<"The name of Mammal is "<<MammalName<<endl;
25     }
26 };
27
```

```

class Bird
{
    string BirdName;

public:
    Bird()
    {
        BirdName = "Null";
    }
    string getBirName()
    {
        return BirdName;
    }
    void setBirdName(string Name)
    {
        BirdName = Name;
    }
    void showBird()
    {
        cout<<"The name of bird is "<<BirdName<<endl;
    }
};

```

```

50
51 class Organism:public Bird,public Mammal
52 {
53     string OrganismName;
54
55     public:
56     Organism()
57     {
58         OrganismName = "Null";
59     }
60
61     void setOrganismName(string name)
62     {
63         OrganismName = name;
64     }
65     void showOrganism()
66     {
67         cout<<"The name of organism is "<<OrganismName<<endl;
68     }
69     char* getOrganismName()
70     {
71         char* name;
72         name = &OrganismName[0];
73         return name;
74     }
75 };
76

```

```

77 int main()
78 {
79     Organism org;
80     org.setBirdName("Dove");
81     org.setMammalName("Tiger");
82     org.setOrganismName("Lion");
83     org.showBird();
84     org.showMammal();
85     org.showOrganism();
86     cout<<endl<<org.getOrganismName();
87     return 0;
88 }

```

Output:

```
PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if ($?) { g++ 220201012_Assign02_Problem01.cpp -std=c++11 -o 220201012_Assign02_Problem01.exe }
The name of bird is Dove
The name of Mammal is Tiger
The name of organism is Lion

Lion
PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>
```

Question 02

In this task, we would try to find out areas of various shapes using inheritance. You need to create a class name 'Base'. Define two protected data members of float type for length and width – 'LengthorBase' and 'WidthorHeight'. Now you need to define constructor(s), setter(s) and a getter for the base class. Next, you need to define three child classes, 'Rectangle', 'Square' and 'Triangle'. Every class must contain a constructor that invokes the parent class constructor whenever a child instance is created. Following the same approach, create the particular getter, setter(s) and area functions for each of the class. Area of a class must return a value accordingly. Use a suitable inheritance.

classBase

- Base(float, float){}
- void baseGetter() const{}

classRectangle

- Rectangle(float, float);
- void rectangleGetter(){}
- void rectangleSetter(){}

- voidrectangleSetter(float, float){}
- floatareaOfRectangle(){}

classSquare

- Square(float, float);
- voidsquareGetter(){}
- voidsquareSetter(){}
- voidsquareSetter(float, float){}
- floatareaOfSquare(){}

classTriangle

- Triangle(float, float);
- voidtriangleGetter(){}
- voidtriangleSetter(){}
- voidtriangleSetter(float, float){}
- floatareaOfTriangle(){}

Code:

```

1  #include <iostream>
2  using namespace std;
3
4  class Base
5  {
6      protected:
7          float Length_or_Base;
8          float Width_or_Height;
9
10     public:
11         Base()
12         {
13             cout<<"called"<<endl;
14             Length_or_Base = 0.0;
15             Width_or_Height = 0.0;
16         }
17         Base(float base,float height)
18         {
19             Length_or_Base = base;
20             Width_or_Height = height;
21         }
22         void setLength(float base)
23         {
24             Length_or_Base = base;
25         }
26         void setWidth(float height)
27         {
28             Width_or_Height = height;
29         }
30
31         float getLength()
32         {
33             return Length_or_Base;
34         }
35         float getWidth()
36         {
37             return Width_or_Height;
38         }
39     };
40
41     class Rectangle:public Base
42     {
43     public:
44         Rectangle()
45         {
46             Length_or_Base = 0.0;
47             Width_or_Height = 0.0;
48         }
49         Rectangle(float base,float height)
50         {
51             Length_or_Base = base;
52             Width_or_Height = height;
53         }
54         void rectangleSetter(float base,float height)
55         {
56             Length_or_Base = base;
57             Width_or_Height = height;
58         }
59     };

```

```

58     void rectangleGetter()
59     {
60         cout<<"Length is "<<Length_or_Base;
61         cout<<"Width is "<<Width_or_Height;
62     }
63     float areaOfRectangle()
64     {
65         return Length_or_Base * Width_or_Height;
66     }
67 };
68
69 class Triangle:public Base
70 {
71     public:
72     Triangle()
73     {
74         Length_or_Base = 0.0;
75         Width_or_Height = 0.0;
76     }
77     Triangle(float base,float height)
78     {
79         Length_or_Base = base;
80         Width_or_Height = height;
81     }
82     void triangleSetter(float base,float height)
83     {
84         Length_or_Base = base;
85         Width_or_Height = height;
86     }

```

```

87     void triangleGetter()
88     {
89         cout<<"Length is "<<Length_or_Base;
90         cout<<"Width is "<<Width_or_Height;
91     }
92     float areaOfTriangle()
93     {
94         return Length_or_Base * Width_or_Height /2;
95     }
96 };
97
98 class Square:public Base
99 {
100     public:
101     Square()
102     {
103         Length_or_Base = 0.0;
104         Width_or_Height = 0.0;
105     }
106     Square(float base,float height)
107     {
108         Length_or_Base = base;
109         Width_or_Height = height;
110     }
111     void squareSetter(float base,float height)
112     {
113         Length_or_Base = base;
114         Width_or_Height = height;
115     }
116     void squareGetter()
117     {
118         cout<<"Length is "<<Length_or_Base;
119         cout<<"Width is "<<Width_or_Height;
120     }
121     float areaOfSquare()
122     {
123         return Length_or_Base * Width_or_Height;
124     }
125 };
126
127 int main()
128 {
129     Square sqr;
130     Triangle trgl;
131     Rectangle rct;
132
133     sqr.setLength(4);
134     sqr.setWidth(4);
135     cout<<endl<<"The length of square = "<<sqr.getLength()<<endl;
136     cout<<"Area of square = "<<sqr.areaOfSquare()<<endl<<endl;
137
138     trgl.setLength(3);
139     trgl.setWidth(5);
140     cout<<"Base of triangle = "<<trgl.getWidth()<<endl;
141     cout<<"Height of triangle = "<<trgl.getLength()<<endl;
142     cout<<"Area of triangle = "<<trgl.areaOfTriangle()<<endl<<endl;
143
144     rct.setLength(3);
145     rct.setWidth(5);
146     cout<<"Length of rectangle = "<<rct.getLength()<<endl;
147     cout<<"Width of rectangle = "<<rct.getWidth()<<endl;
148     cout<<"Area of rectangle = "<<rct.areaOfRectangle()<<endl;
149
150     return 0;
151 }

```


Output:

```
PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if ($?) { g++ 22020
{ .\220201012_Assign_02_03_Q2 }
called
called
called

The length of square = 4
Area of square = 16

Base of triangle = 5
Height of triangle = 3
Area of triangle = 7.5

Length of rectangle = 3
Width of rectangle = 5
Area of rectangle = 15
PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>
```

Question 03

We want to calculate the total marks of each student of a class in Physics, Chemistry and Mathematics and the average marks of the class. The number of students in the class are entered by the user. Create a class named Marks with data members for roll number, name and marks. Create three other classes inheriting the Marks class, namely Physics, Chemistry and Mathematics, which are used to define marks in individual subject of each student. Roll number of each student will be generated automatically.

Code:

```

1  #include <iostream>
2  using namespace std;
3
4  class Marks
5  {
6      protected:
7          int rollNo;
8          string name;
9          float marks;
10
11     public:
12         Marks()
13         {
14             rollNo = RollGenerate();
15             name = "NULL";
16             marks = 0;
17         }
18     private:
19         int RollGenerate()
20         {
21             static int temp0 = 0;
22             return ++temp0;
23         }
24
25 };
26

```

```

27 class Physics:public Marks
28 {
29     public:
30     Physics()
31     {}
32     Physics(string Name,float mark)
33     {
34         name = Name;
35         marks = mark;
36     }
37     void setName(string Name)
38     {
39         name = Name;
40     }
41     void setMarks(float mark)
42     {
43         marks = mark;
44     }
45     string getName()
46     {
47         return name;
48     }
49     int getRoll()
50     {
51         return rollNo;
52     }
53     float getMarks()
54     {
55         return marks;
56     }
57 };

```

```

59 class Mathematics:public Marks
60 {
61     public:
62     Mathematics()
63     {}
64     Mathematics(string Name,float mark)
65     {
66         name = Name;
67         marks = mark;
68     }
69     void setName(string Name)
70     {
71         name = Name;
72     }
73     void setMarks(float mark)
74     {
75         marks = mark;
76     }
77     string getName()
78     {
79         return name;
80     }
81     int getRoll()
82     {
83         return rollNo;
84     }
85     float getMarks()
86     {
87         return marks;
88     }
89 };

```

```

91 class Chemistry:public Marks
92 {
93     public:
94     Chemistry()
95     {}
96     Chemistry(string Name,float mark)
97     {
98         name = Name;
99         marks = mark;
100     }
101     void setName(string Name)
102     {
103         name = Name;
104     }
105     void setMarks(float mark)
106     {
107         marks = mark;
108     }
109     string getName()
110     {
111         return name;
112     }
113     int getRoll()
114     {
115         return rollNo;
116     }
117     float getMarks()
118     {
119         return marks;
120     }
121 };

```

```

122
123     int main()
124     {
125         int temp,mathMarks,phyMarks,chemMarks,temp1=0,temp2=0,temp3=0,temp4=0,tempo;
126         string name;
127         Run and Debug (Ctrl+Shift+D) many students are there?"<<endl;
128         cin>>temp;
129
130         Physics* phy = new Physics[temp];
131         Chemistry* chem = new Chemistry[temp];
132         Mathematics* math = new Mathematics[temp];
133
134         for (int i=0;i<temp;i++)
135         {
136             cout<<"What is the name of "<<i+1<<"th student?"<<endl;
137             cin>>name;
138             cout<<"What are the marks in physics?"<<endl;
139             cin>>phyMarks;
140             cout<<"What are the marks in chemistry?"<<endl;
141             cin>>chemMarks;
142             cout<<"What are the marks in Maths?"<<endl;
143             cin>>mathMarks;
144             phy[i] = Physics(name,phyMarks);
145             chem[i] = Chemistry(name,chemMarks);
146             math[i] = Mathematics(name,mathMarks);
147         }
148
149         cout<<endl;
150         cout<<"The total marks of students are: "<<endl;
151         for (int i=0;i<temp;i++)
152         {
153             temp1 += phy[i].getMarks();
154             temp2 += chem[i].getMarks();
155             temp3 += math[i].getMarks();
156             tempo = phy[i].getMarks() + chem[i].getMarks() + math[i].getMarks();
157             cout<<phy[i].getName()<<": "<<tempo<<endl;
158         }
159
160
161         cout<<endl;
162         cout<<"The average in each subject is: "<<endl;
163         cout<<"Physics: "<<temp1/temp<<endl;
164         cout<<"Chemistry: "<<temp2/temp<<endl;
165         cout<<"Mathematics: "<<temp3/temp<<endl;
166
167         return 0;
168     }

```

Output:

```

PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if
f ($?) { .\220201012_Assign02_Problem03 }
How many students are there?
3
What is the name of 1th student?
Ali
What are the marks in physics?
37
What are the marks in chemistry?
34
What are the marks in Maths?
33
What is the name of 2th student?
Ahmad
What are the marks in physics?
41
What are the marks in chemistry?
39
What are the marks in Maths?
43
What is the name of 3th student?
Ashfaq
What are the marks in physics?
46
What are the marks in chemistry?
47
What are the marks in Maths?
46

The total marks of students are:
Ali: 104
Ahmad: 123
Ashfaq: 139

The average in each subject is:
Physics: 41
Chemistry: 40
Mathematics: 40
PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>

```

Question 04

Consider a class Computer having Two fields (i.e. companyName, price) and A single function named show() A class named Desktop inherits Computer class and adds fields representing color, monitor size, and processor type and Override function named show() to display values of its all attributes A class named Laptop inherits Computer class and adds fields representing color, size, weight, and processor type and Override function named show() to display values of its all

attributes In Main() instantiate objects of derived classes to access respective show() functions to see the polymorphic behavior

Code:

```
1  #include <iostream>
2  using namespace std;
3
4  class Computer
5  {
6      string companyName;
7      int price;
8
9      public:
10     Computer()
11     {
12         companyName = "NULL";
13         price = 0;
14     }
15     Computer(string name,int Price)
16     {
17         companyName = name;
18         price = Price;
19     }
20     void setCompany(string name)
21     {
22         companyName = name;
23     }
24     void setPrice(int pri)
25     {
26         price = pri;
27     }
28     string getCompany()
29     {
30         return companyName;
31     }
32     int getPrice()
33     {
34         return price;
35     }
36     void show()
37     {
38         cout<<"Company: "<<companyName<<endl;
39         cout<<"Price: "<<price<<endl;
40     }
41 };
42
43 class Desktop:public Computer
44 {
45     string color;
46     float monitorSize;
47     string processorType;
48
49     public:
50     Desktop()
51     {
52         color = "NULL";
53         monitorSize = 0;
54         processorType = "NULL";
55     }
56     Desktop(string name,int Price,string Color,int size,string type):Computer(name,Price),
57     color(Color),monitorSize(size),processorType(type){}
58     void setColor(string Color)
59     {
```

```

59         color = Color;
60     }
61     void setType(string type)
62     {
63         processorType = type;
64     }
65     void setSize(int size)
66     {
67         monitorSize = size;
68     }
69     string getColor()
70     {
71         return color;
72     }
73     int getSize()
74     {
75         return monitorSize;
76     }
77     string getType()
78     {
79         return processorType;
80     }
81
82     void show()
83     {
84         Computer::show();
85         cout<<"Color: "<<color<<endl;
86         cout<<"Monitor Size: "<<monitorSize<<endl;
87         cout<<"Processor Type: "<<processorType<<endl;
88     }
89 };
90
91
92 class Laptop:public Computer
93 {
94     string color;
95     float size;
96     float weight;
97     string processorType;
98
99     public:
100     Laptop()
101     {
102         color = "NULL";
103         size = 0;
104         weight = 0;
105         processorType = "NULL";
106     }
107     Laptop(string name,int Price,string Color,float Size,float Weight,string type):Computer(name,Price),
108     color(Color),size(Size),processorType(type),weight(Weight){}
109     void setColor(string Color)
110     {
111         color = Color;
112     }
113     void setType(string type)
114     {
115         processorType = type;
116     }
117     void setSize(float Size)
118     {
119         size = Size;
120     }
121     void show() const

```



```

121  void setWeight(float Weight)
122  {
123      weight = Weight;
124  }
125  string getColor()
126  {
127      return color;
128  }
129  int getSize()
130  {
131      return size;
132  }
133  string getType()
134  {
135      return processorType;
136  }
137
138  void show()
139  {
140      Computer::show();
141      cout<<"Color: "<<color<<endl;
142      cout<<"Size: "<<size<<endl;
143      cout<<"Weight: "<<weight<<endl;
144      cout<<"Processor Type: "<<processorType<<endl;
145  }
146  };
147
148  int main()
149  {
150      Laptop lap("Dell",150000,"Black",17,160,"Ryzen 7 5700U");
151      Desktop dsk("Alienware",275000,"White",32,"Ryzen 7 5800X3D");
152      lap.show();
153      cout<<endl;
154      dsk.show();
155      cout<<endl;
156
157      return 0;
158  }

```

Output:

```

PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if ($?) { g++ 2
{ .\220201012_Assign_02_03_Q4 }
Company: Dell
Price: 150000
Color: Black
Size: 17
Weight: 160
Processor Type: Ryzen 7 5700U

Company: Alienware
Price: 275000
Color: White
Monitor Size: 32
Processor Type: Ryzen 7 5800X3D

PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>

```


Question 05

Write a class Distance that holds distances or measurements expressed in feet and inches. This class has two private data members:

- feet: An integer that holds the feet.
- inches: An integer that holds the inches.
- Write a constructor with default parameters that initializes each data member of the class. If inches are greater than equal to 12 then they must be appropriately converted to corresponding feet.
- Generate appropriate getter-setter functions for the data members.

1. void setFeet(int f) and int getFeet()const
2. void setInches(int i) It should ensure proper conversion to feet.
3. int getInches() const

- Define an operator '+' that overloads the standard '+' math operator and allows one Distance object to be added to another. Distance operator+ (const Distance &obj).

- Define an operator -function that overloads the standard '-' math operator and allows subtracting one Distance object from another. Distance operator-(const Distance &obj)

- Define an operator= function that overloads the = operator and assign one Distance object to another. const Distance operator=(const Distance &obj)

Code:

```

1  #include <iostream>
2  using namespace std;
3
4  class Distance
5  {
6      int feet,inches;
7
8      public:
9      Distance()
10     {
11         feet = 0;
12         inches = 0;
13     }
14     Distance(int a, int b)
15     {
16         while (b>=12)
17         {
18             b %= 12;
19             a += 1;
20         }
21         feet = a;
22         inches = b;
23     }
24     void setFeet(int f)
25     {
26         feet = f;
27     }
28     int getFeet()
29     {
30         return feet;
31     },
32     void setInches(int i)
33     {
34         while (i>=12)
35         {
36             i %= 12;
37             feet += 1;
38         }
39         inches = i;
40     }
41     int getInches()
42     {
43         return inches;
44     }
45
46     Distance operator + (const Distance &obj)
47     {
48         Distance d;
49         d.feet = feet + obj.feet;
50         d.inches = inches + obj.inches;
51         while (d.inches>=12)
52         {
53             d.inches %= 12;
54             d.feet += 1;
55         }
56         return d;
57     }
58

```

```

59     Distance operator - (const Distance &obj)
60     {
61         Distance d;
62         if (inches < obj.inches)
63         {
64             d.inches = obj.inches - inches;
65             d.feet = feet - obj.feet - 1;
66         }
67         else
68         {
69             d.inches = inches - obj.inches;
70             d.feet = feet - obj.feet;
71         }
72         return d;
73     }
74
75     Distance operator = (const Distance &obj)
76     {
77
78         feet = obj.feet;
79         inches = obj.inches;
80
81     }
82
83     return *this;
84 };
85
86 int main()
87 {
88     Distance d1(6,11);
89     cout<<"Distance 1 is:"<<endl<<d1.getFeet()<<" feet and "
90         <<d1.getInches()<<" inches."<<endl<<endl;
91
92     Distance d2(5,13);
93     cout<<"Distance 2 is:"<<endl<<d2.getFeet()<<" feet and "
94         <<d2.getInches()<<" inches."<<endl<<endl;
95
96     Distance d3;
97     d3.setFeet(7);
98     d3.setInches(14);
99     cout<<"Distance 3 is:"<<endl<<d3.getFeet()<<" feet and "
100         <<d3.getInches()<<" inches."<<endl<<endl;
101
102
103     Distance d4;
104     d4 = d2 + d3;
105     cout<<"Distance 4 = Distance 2 + Distance 3:"<<endl<<d4.getFeet()<<" feet and "
106         <<d4.getInches()<<" inches."<<endl<<endl;
107
108     Distance d5;
109     d5 = d1 - d2;
110     cout<<"Distance 5 = Distance 1 - Distance 2:"<<endl<<d5.getFeet()<<" feet and "
111         <<d5.getInches()<<" inches."<<endl<<endl;
112
113     Distance d6;
114     d6 = d1;
115     cout<<"Distance 6 = Distance 1 (assignment operator):"<<endl<<d6.getFeet()<<" feet and "
116         <<d6.getInches()<<" inches."<<endl<<endl;
117
118
119     return 0;
120
121 }

```

Output:

```
PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if ($?) { g++ 2
{ .\220201012_Assign_02_03_Q5 }
Distance 1 is:
6 feet and 11 inches.

Distance 2 is:
6 feet and 1 inches.

Distance 3 is:
8 feet and 2 inches.

Distance 4 = Distance 2 + Distance 3:
14 feet and 3 inches.

Distance 5 = Distance 1 - Distance 2:
0 feet and 10 inches.

Distance 6 = Distance 1 (assignment operator):
6 feet and 11 inches.

PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>
```

Question 06

You are supposed to construct a class named Matrix that shall contains private data member:

- matrix:integer type array of size 3 by 3;

Define a constructor that should have default parameters that can set all elements to 0. Moreover, define a function called setMatrixValues(int matrixArray[3][3]) that can assign values of matrixArray (provided to the function in argument) to the matrix array (data member of the class Matrix) of the class. Along with it, define a function called displayMatrix() that can display all values of the matrix. Moreover, you are instructed to construct a program that can perform operator overloading for the following operators + (plus), -(minus), and == (equal).

Code:

```
1  #include <iostream>
2  using namespace std;
3
4  class Matrix
5  {
6      int matrix[3][3];
7
8      public:
9      Matrix()
10     {
11         for (int i=0;i<3;i++)
12         {
13             for (int j=0;j<3;j++)
14             {
15                 matrix[i][j] = 0;
16             }
17         }
18     }
19
20     void setMatrixValues(int matrixArray[3][3])
21     {
22         for (int i=0;i<3;i++)
23         {
24             for (int j=0;j<3;j++)
25             {
26                 matrix[i][j] = matrixArray[i][j];
27             }
28         }
29     }
30
31     void display()
32     {
33         for (int i=0;i<3;i++)
34         {
35             cout<<"[ ";
36             for (int j=0;j<3;j++)
37             {
38                 cout<<matrix[i][j]<<" ";
39             }
40             cout<<"]"<<endl;
41         }
42         cout<<endl<<endl;
43     }
44
45     Matrix operator + (Matrix &M)
46     {
47         Matrix temp;
48         for (int i=0;i<3;i++)
49         {
50             for (int j=0;j<3;j++)
51             {
52                 temp.matrix[i][j] = matrix[i][j] + M.matrix[i][j];
53             }
54         }
55         return temp;
56     }
57 }
```

```

58     Matrix operator - (Matrix &M)
59     {
60         Matrix temp;
61         for (int i=0;i<3;i++)
62         {
63             for (int j=0;j<3;j++)
64             {
65                 temp.matrix[i][j] = matrix[i][j] - M.matrix[i][j];
66             }
67         }
68         return temp;
69     }
70
71     bool operator == (Matrix &M)
72     {
73         int count = 0;
74         for (int i=0;i<3;i++)
75         {
76             for (int j=0;j<3;j++)
77             {
78                 if (matrix[i][j] != M.matrix[i][j])
79                 {
80                     count += 1;
81                 }
82             }
83         }
84         if (count==0)
85         {
86             return true;
87         }
88         else
89         {
90             return false;
91         }
92     }
93 };
94
95
96 int main()
97 {
98     Matrix m1,m2,m3,m4;
99     bool temp,temp1;
100
101     int mat[3][3];
102     for (int i=0;i<3;i++)
103     {
104         for (int j=0;j<3;j++)
105         {
106             mat[i][j] = i*j;
107         }
108     }
109
110     m1.setMatrixValues(mat);
111     m2.setMatrixValues(mat);
112     cout<<"Matrix 1 is:"<<endl;
113     m1.display();
114     cout<<"Matrix 2 is:"<<endl;
115     m2.display();
116     m3 = m1 + m2;
117     cout<<"Matrix 3 = Matrix 1 + Matrix 2:"<<endl;
118     m3.display();

```

```

119     m4 = m1 - m2;
120     cout<<"Matrix 4 = Matrix 1 - Matrix 2:"<<endl;
121     m4.display();
122     temp = m1==m2;
123     temp1 = m1==m3;
124     if (m1==m2)
125     {
126         cout<<"Matrix 1 and Matrix 2 are equal."<<endl;
127     }
128     else
129     {
130         cout<<"Matrix 1 and Matrix 2 are not equal."<<endl;
131     }
132     if (m1==m3)
133     {
134         cout<<"Matrix 1 and Matrix 3 are equal."<<endl;
135     }
136     else
137     {
138         cout<<"Matrix 1 and Matrix 3 are not equal."<<endl;
139     }
140
141     return 0;
142 }

```

Output:

```

PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assigment 2,3\" ; if ($?) { g++ 2202
{ .\220201012_Assign_02_03_Q6 }
Matrix 1 is:
[ 0 0 0 ]
[ 0 1 2 ]
[ 0 2 4 ]

Matrix 2 is:
[ 0 0 0 ]
[ 0 1 2 ]
[ 0 2 4 ]

Matrix 3 = Matrix 1 + Matrix 2:
[ 0 0 0 ]
[ 0 2 4 ]
[ 0 4 8 ]

Matrix 4 = Matrix 1 - Matrix 2:
[ 0 0 0 ]
[ 0 0 0 ]
[ 0 0 0 ]

Matrix 1 and Matrix 2 are equal.
Matrix 1 and Matrix 3 are not equal.
PS D:\University\Assignments\Semester 2\OOP\Assigment 2,3>

```

Question 07

We want to create a class of Product that contains multiple private data members such as

- quantity: An integer that holds a count value.

- objCount: A static integer that holds that count of objects.
- serialNo: An integer that holds the serial number of objects of a specific product (assume the single object of Product class).
- Define a constructor that can accept two arguments i.e., totalQuantityOfProduct and serialNumberOfProduct; and assign it to the respective data members of the class. Moreover, the static member be shall be initialized from the outside of the class with zero by using scope resolution operator.
- Define operator = that add the value of quantity to the left hand operand. i.e. c2=c1 (the quantity of object c2 shall be incremented with the quantity of c1).
- Define unary operator -that inverts the value of quantity for product class and should allow the statements like c1 -= 4;

Code:


```

1  #include <iostream>
2  using namespace std;
3
4  class Product
5  {
6      int quantity;
7      static int objCount;
8      int serialNo;
9
10     public:
11     Product()
12     {
13         quantity = 0;
14         serialNo = 0;
15     }
16     Product(int totalQuantityOfProduct,int serialNumberOfProduct)
17     {
18         quantity = totalQuantityOfProduct;
19         serialNo = serialNumberOfProduct;
20     }
21
22     void setQuantity(int quan)
23     {
24         quantity = quan;
25     }
26     void setSerial(int serial)
27     {
28         serialNo = serial;
29     }
30
31     int getQuantity()
32     {
33         return quantity;
34     }
35     int getSerial()
36     {
37         return serialNo;
38     }
39
40     Product operator +=(Product &obj)
41     {
42         quantity += obj.quantity;
43         return *this;
44     }
45     Product operator -= (Product &obj)
46     {
47         quantity -= obj.quantity;
48         return *this;
49     }
50     Product operator -= (int num)
51     {
52         quantity -= num;
53         return *this;
54     }
55 };
56

```

```

59 int main()
60 {
61     Product prd1(10,15), prd2, prd3, prd4;
62     prd2.setQuantity(20);
63     prd2.setSerial(101);
64
65     cout<<"Product 1:"<<endl;
66     cout<<"Quantity: "<<prd1.getQuantity()<<endl;
67     cout<<"Serial no: "<<prd1.getSerial()<<endl<<endl;
68
69     cout<<"Product 2:"<<endl;
70     cout<<"Quantity: "<<prd2.getQuantity()<<endl;
71     cout<<"Serial no: "<<prd2.getSerial()<<endl<<endl;
72
73     prd2 = prd1;
74
75     cout<<"Product 2 after adding values of Product 1:"<<endl;
76     cout<<"Quantity: "<<prd2.getQuantity()<<endl;
77     cout<<"Serial No: "<<prd2.getSerial()<<endl<<endl;
78
79     prd2 -= prd1;
80
81     cout<<"Product 2 after subtracting values of Product 1:"<<endl;
82     cout<<"Quantity: "<<prd2.getQuantity()<<endl;
83     cout<<"Serial No: "<<prd2.getSerial()<<endl<<endl;
84
85     prd2 -= 4;
86
87     cout<<"PRproduct 2 after subtracting 4 from the quantity:"<<endl;
88     cout<<"Quantity: "<<prd2.getQuantity()<<endl;
89     cout<<"Serial No: "<<prd2.getSerial()<<endl<<endl;
90
91     return 0;
92 }
93

```

Output:

```

PS C:\Users\M> cd "d:\University\Assignments\Semester 2\OOP\Assignment 2,3\" ; if ($?) { g++ 2
{ .\220201012_Assign_02_03_Q7 }
Product 1:
Quantity: 10
Serial no: 15

Product 2:
Quantity: 20
Serial no: 101

Product 2 after adding values of Product 1:
Quantity: 30
Serial No: 101

Product 2 after subtracting values of Product 1:
Quantity: 20
Serial No: 101

PRproduct 2 after subtracting 4 from the quantity:
Quantity: 16
Serial No: 101

PS D:\University\Assignments\Semester 2\OOP\Assignment 2,3>

```