OOP Lab 6: Operator overloading and friend functions

Date: 23-10-2020

Total marks: 15+15=30

Deadline: same day (Friday, 23-10-2020) before 11:59 pm (sharp)

Submission instructions

1. Submit your file in a .zip or .rar format. Name your file with your Roll number (e.g. BSEF19M009.zip or BSEF19M009.rar)
2. Make a separate folder for each lab task (e.g. task1, task2…). All the headers and .cpp files must be in respective folder with a screenshot of the output of a program of the respective task.
3. Every Cpp file must contain your name and roll number(In comments) at the top of each program
4. Don't enclose your code in comments otherwise it will not be evaluated.

Instructions: (MUST READ)

- **Understanding the problem statement is a part of that question. Do whatever you are asked and what you understand.NO QUERIES ARE ALLOWED as this lab is overly simple**
- **No compensation or makeup lab.**
- **Don't discuss with peers. Changing variable names/ changing for to while loop will not help you in hiding cheating attempt!**
- **Cheating cases will simply be dropped out and their future lab submissions will NOT be graded.**
- **You are not allowed to consult Internet. Plagiarism cases will be strictly dealt.**
- **You must submit the lab solution BEFORE 11:59 pm. Even a few minutes late submissions will not be considered. Make sure to do proper management of time/Internet connectivity/power failure or whatever issue is possible!**

General instructions for all tasks: (marks will be deducted if the instructions are violated)

Note: All the programs should be implemented using class. You can take input in main() function and then call appropriate methods/ member functions of a designed class to set and get values. YOU CAN CREATE A TASK IN SINGLE CPP FILE FOR CLASS DECLARATION AND DEFINITION.

• The attributes of class should be declared as **private** and member functions as **public**.

• You should not initialize the attributes while declaring them in class. The values should be assigned using member functions only. E.g. you cannot declare like:

```
Class Person
{
        Private:
```

int age=25;
}

• The values should be initialized using a constructor. There must be a constructor in your defined class.

• All inputs should be taken in *main()* and all the final results should also be reported/ displayed in the main function.

• All the logic should be implemented in class' member functions. Main() should only input and output relevant values by calling relevant functions of the class.

Task 1:

- Equilateral Triangle has:
  - Three equal sides
  - Three equal angles
- Isosceles Triangle has:
  - Two equal sides
  - Two equal angles

Create two classes **EquilateralTriangle** and **IsoscelesTriangle** with relevant data members (e,g, three sides and three angles for EquilateralTriangle).

Implement a type casting function that converts EquilateralTriangle to IsoscelesTriangle class. In typecasting function you can simply put zero to one of the three sides/angles of EquilateralTriangle class to make it equivalent to IsoscelesTriangle class.

Write a main function that calls the cast operator (overloaded) and then display data members of both the classes. For displaying data members there should be a separate member function in respective classes.

Take all inputs from the user, NO HARDCODING IS ALLOWED.

Task 2:

Create a class Matrix in C++ having a 2-D integer array mat[3][3] as data member (declare it as dynamic). Overload the following binary and unary operators for objects m1 and m2 of this class.

1. m1 + m2
2. m1 - m2
3. m1 * m2 (multiplication of two matrices)
4. m1 * num (num is a scalar)
5. num * m1
6. m1 * vec (vec is a 1-D integer array)
7. vec * m1

8. m1 < m2;m1 > m2;m1 >= m2;m1 <= m2;m1 == m2 (comparison operators that return boolean value)
9. m1^num (power of individual elements of matrix to scalar num)
10. m1/num (division by scalar)

Implement the default and parameterized constructors and accessor methods as needed.

In main function, write a code that tests each of the 10 operations defined above, one by one. Take all inputs from the user, NO HARDCODING IS ALLOWED.