ET580, Exam1, Spring 2023

Submission detail:

A) Submit **cpp file** for each question with given main function.

B) points distribution: 50% compilation, 50% correctness and logic.

C) Have to use **given main function** to test your answer, **otherwise 0 will be given**.

Q1 - Q5 (20pt each)

Q1: Implement a class called **Point**.
a) A class named **Point**
b) Two **private data members**: x coordinate(double) and y coordinate(double).
c) **Accessors** and **Mutators** function for x, y.
d) implement a member function names ***output*** that will print all information of **Point** object.

Use following main() to test your class.

```
int main(){
  Point a;
  a.setX(1.5);
  a.setY(2.5);
  cout<<a.getX()<<endl;
  cout<<a.getY()<<endl;
  a.output();
}
```

Output from given main: (user input in **bold font**)

```
1.5
2.5
X:1.5 Y:2.5
```

Answer:

Q2: Continue with **Point** class:
a) Copy the previous program to a new file.
b) Write **Constructor with two parameter**, and assign to x, and y.
c) Write **default constructor**, initialize x to 0, y to 0, Implement constructor delegation.
d) implement a non-member function names ***distance*** that return the distance between two **Point** object.

The coordinates of point A are (x_1,y_1) and of B are (x_2,y_2)

$$AB = \sqrt{[x_2 - x_1]^2 + [y_2 - y_1]^2}$$

Use following main() to test your class.

```
int main(){
  Point a(3,4),b;
  cout<<distance(a,b)<<endl; // 5
  Point c(3,0),d(0,4);
  cout<<distance(c,d)<<endl; // 5
}
```

Output from given main: (user input in **bold font**)

```
5
5
```

Answer:

Q3: Continue with **Point** class:
a) Copy the previous program to a new file.
b) Overload the insertion **operator <<** to output x, and y of **Point** objects.
c) Overload the extraction **operator >>** to input x and y of **Point** objects.

Use following main() to test your class.

```
int main(){
  Point a(3,4);
  cout<<a<<endl; // (3,4)
  cout<<"Enter x, y:";
  cin>>a;
  cout<<a<<endl;
}
```

Output from given main: (user input in **bold**)

```
(3,4)
Enter x, y: 6 7
(6,7)
```

Answer:

Q4: Continue with **Point** class:
a) Copy the previous program to a new file.

Implement comparison operator overload for comparing the salary of **Point** object.
a) Overload the operator == to output boolean.

Implement arithmetic operator overload for **Point** object.

b) Overload the operator +, which can add x and y coordinate of two Point Objects, and return a new temporary **Point** object.

c) Overload the operator +, which can add an integer to x and y coordinate of **Point** Objects, and return a new temporary **Point** object.

d) Overload the operator += to update x and y coordinate of **Point,** which can add x and y of a **Point** to x and y coordinate of another **Point**.

e) Overload the operator += to update x and y coordinate of **Point,** which can add an integer to x and y coordinate of **Point**.

Use following main() to test your class.

```
int main(){
  Point a(3,4),b(1,2);
  cout<<a+b<<endl;    // (4,6)
  cout<<a+10<<endl;   // (13,14)
  cout<<a<<endl;      // (3,4)
  cout<<b<<endl;      // (1,2)
  cout<<(a+=b)<<endl; // (4,6)
  cout<<a<<endl;      // (4,6)
  cout<<b<<endl;      // (1,2)
  cout<<(a+=10)<<endl;// (14,16)
  cout<<a<<endl;      // (14,16)
}
```

Output from given main: (user input in **bold**)

```
(4,6)
(13,14)
(3,4)
(1,2)
(4,6)
(4,6)
(1,2)
(14,16)
(14,16)
```

Answer:

Q5: Continue with **Point** class:
a) Copy the previous program to a new file.
b) Overload [] operator, index 0 return x, index 1 return y of **Point** Object. (Only index 0,1 is valid, else terminate program)

Implement prefix and postfix operator overload for **Point** object, ++ operator should increase x and y of **Point** by 1s.

c) Overload the prefix and postfix ++ operators.
Use following main() to test your class.

```
int main(){
  Point a(3,4);
  cout<<a[0]<<endl; // 3
  cout<<a[1]<<endl; // 4
  a[0]=1;
  a[1]=2;
  cout<<a<<endl;    // (1,2)
  cout<<++a<<endl;  // (2,3)
  cout<<a<<endl;    // (2,3)
  cout<<a++<<endl;  // (2,3)
  cout<<a<<endl;    // (3,4)
}
```

Output from given main:

```
3
4
(1,2)
(2,3)
(2,3)
(2,3)
(3,4)
```

Answer: