Object Oriented Programming

CS 244 - 040

Department of Physical and Computer Sciences

Medgar Evers College

Exam 4 - Take Home

Name: _____

Directions: Read the questions carefully. Write legibly to earn credit. Good Luck!

Section	Max Points	Points Earned
1	16	
2	4	
Total	20	

Section 1: Class

Create the header file named "Indexer.h" that contains the following classes:

The header file will consist of two classes, **Index** and **BoundedIndex**. The class **Index** counts up from 0 onward, or from another start point. While the class **BoundedIndex** count up from 0 to an upper bound or from another start point.

The class **Index** should have the following features:

- □ Private int field named *counter*.
- □ Public default constructor that assigns 0 to *counter*.
- □ Public overloaded constructor that takes a int as a parameter named *start*. It assigns *start* to *counter* if it is greater than or equal to 0; otherwise, it assigns 0 to *counter*.
- □ Public copy constructor.
- □ Public overloaded assignment operator.
- □ Public empty destructor.
- □ Public constant get method for *counter*.
- □ Protected set method for *counter*.
- □ Public void method named `Reset()` that takes no parameters. It assigns 0 to *counter*.
- □ Public void method named `Increment()` that takes no parameters. It increments *counter* by 1.
- □ Public string constant method named `ToString()` that takes no parameters. It returns a string of *counter*.
- □ Friend overloaded ostream operator. It displays a **Index** object in the same format as `ToString()`.

The class **BoundedIndex** should have the following features:

- □ Publicly inherits **Index**.
- □ Private int field named *max*.
- □ Public default constructor that assigns 0 to *counter* and 10 to *max*.
- □ Public overloaded constructor that takes a int as a parameter named *max*. It assigns *max* to the *max* field if it is greater than 0; otherwise, it assigns 10 to *max*. Likewise, it assigns 0 to *counter*.
- □ Public overloaded constructor that takes two ints as parameters named *max* and *start* respectively. It assigns *max* to the *max* field if it is greater than 0; otherwise, it assigns 10 to *max*. And it assigns *start* to *counter* if it is greater than or equal to 0; otherwise, it assigns 0 to *counter*.
- □ Public copy constructor.
- □ Public overloaded assignment operator.
- □ Public empty destructor.
- □ Public constant get method for *max*.
- □ Public set method for *max*. It assigns the parameter to *max* only if it is greater than 0.
- □ Public bool method named `IsDone()` that takes no parameters. It returns true if *counter* equals *max*.
- □ Public overridden`Increment()` method. It increments *counter* by 1 only if it is less than *max*.
- □ Public overridden `ToString()` method. It returns a string of *counter* comma *max* enclosed in square braces.
- □ Friend overloaded ostream operator. It displays a **BoundedIndex** object in the same format as `ToString()`.

Section 2: Extra Credit

Define the class **AlternatingBoundedIndex** that has the following features:

- ☐ Publicly inherits **BoundedIndex**.
- ☐ Private int field named *min*.
- ☐ Public default constructor that assigns 0 to *counter*, 10 to *max* and −10 to *min*.
- ☐ Public overloaded constructor that takes two ints as a parameter named x and y respectively. It assigns the maximum of x and y to *max*, the minimum of the values to *min* and the midpoint of the values to *counter*.
- ☐ Public overloaded constructor that takes three ints as parameters named x, y and *start* respectively. It assigns the maximum of x and y to *max*, the minimum of the values to *min* and *start* to *counter* if it is between *max* and *min* inclusively; otherwise, it assigns the midpoint of the values to *counter*.
- ☐ Public copy constructor.
- ☐ Public overloaded assignment operator.
- ☐ Public empty destructor.
- ☐ Public constant get method for *min*.
- ☐ Public overridden set method for *max*. It assigns the parameter to *max* only if it is greater than *min* and assigns the midpoint of *max* and *min* to *counter*.
- ☐ Public set method for *min*. It assigns the parameter to *min* only if it is less than *max* and assigns the midpoint of *max* and *min* to *counter*.
- ☐ Public overridden `Reset()` method. It assigns the midpoint of *max* and *min* to *counter*.
- ☐ Public overridden `IsDone()` method. It returns true if *counter* equals *max* or *min*.
- ☐ Public void method named `Decrement()` that takes no parameters. It decrements *counter* by 1 only if it is greater than *min*.
- ☐ Public overridden `ToString()` method. It returns a string of *min* comma *counter* comma *max* enclosed in square braces.
- ☐ Friend overloaded ostream operator. It displays a **AlternatingBoundedIndex** object in the same format as `ToString()`.