IT230 Lab 2
v1.1
Score: _____/10

The purpose of this lab is to give you more practice writing C++ code independently.

You will use the same repository as lab 1. You will add a new class, header, and new code in your `main.cpp` file to test the new code.

Specifications

Translate the MyOrderedList2 class from C# into a class in C++. Name this class MyOrderedList. Ensure that all functionality mirrors the original class.

To complete this lab, you will need to implement class inheritance in C++. Here are some links which will help you learn this quickly:

- https://www.w3schools.com/cpp/cpp_inheritance.asp
- https://www.tutorialspoint.com/cplusplus/cpp_inheritance.htm
- https://www.programiz.com/cpp-programming/function-overriding

Since C++ does not have a `CompareTo()` function like C#, only allow MyOrderedList to work with numeric types. This will enable the use of standard comparison operators (<, >, etc.).

Use the following concept in your MyOrderedList header file:

```
template <typename T>
    concept Numeric = std::is_arithmetic_v<T>;
```

This concept will enforce a constraint on the types allowed in MyOrderedList.

Important: Do not allow the `insert()` method to be called with an index argument. If it is, throw a `std::logic_error` exception.

Additionally, change certain members (such as `size` and `capacity`) from `private` to `protected`, to allow derived classes access to them.

If you get stuck, **read the Tips section** to see if there is information to help you.

Constraints

- All methods defined in the source file must have their declarations in the header file (.h).
- Use camelCase for naming variables and methods in C++. Classes should be named using PascalCase. Refer to the Tips section for more guidance.
- If a method does not modify the state of the object, mark it as `const`.

Submission

The submission for this lab is the commit ID you want to be graded. You will submit the commit ID via the Canvas assignment. If you need help finding the commit ID, ask for help.

10 points will be assigned by correctness and your observable understanding as reflected in your code.

Remember to write efficient code. Optimize your code wherever possible.

Tips

- I highly recommend making two functions in the same file as your `main()` function. Move all of the tests for MyList into one function and write tests for your MyOrderedList in the other.
  - In C++, your `main()` should either be at the bottom of your source file or you should put all function prototypes at the top of the file.
- The new header file should list all methods you plan to override. Mark overridden methods by adding the `override` keyword after the parameter list (and after `const`, if applicable).
  - The `override` keyword is only required in the header file, not in the source file.
  - Ensure that the methods you override are marked `virtual` in their base class's header file.
- Your new header file should be where your new class definition is. This is where you will show inheritance.
  - Remember to include MyList (which is a template class).
- If you encounter errors like "`size` was not declared in this scope," it’s likely because you are trying to access a `protected` member from the base class in the derived class. To resolve this, use the `this->` pointer to access such members in the derived class implementation, like this:

  ```
  this->size;
  ```
- If you want to test your code using random numbers, this snippet of code should help. It generates a random number between 1 and 10.

  ```
  rand() % 10 + 1
  ```
- When implementing your binary search, make sure the "right boundary" variable does not underflow its bounds. Ensure the condition in your `while` loop includes a check to prevent the right boundary from being `string::npos`.