Assignment03 (CLO3)

Object Oriented Programming

Submission Guidelines:

➢ You need to only submit your source file on **google classroom within deadline.**
➢ Late submissions upto **24 hours** are allowed with a penalty of **20%** on google classroom. After that **no** submission will be entertained
➢ Name the source file as "assign03-roll#.h". For example **assig01-i200561.h**
➢ **Don't share your code with anyone.**
➢ **Using ChatGPT for your assignment will not help you to secure good grades in this subject**

myString class

In this assignment, you will create a class myString, which should serve as a competitor to the standard string class of C++. The myString class should have a char array as its data member, where the string data will be stored. To start with, you can have a fixed size of 200 for this char array.
This size will not be able to store anything larger than 200 char data. So if you want to handle any size, then this array should be dynamically allocated, and then its size can be changed whenever required. Creating dynamic array is optional, and carry bonus marks. More on this at the end.

In this assignment, you don't need to implement everything into your class, what a standard string class does. Instead, you just need to implement atleast what is needed to run a demo program associated with this assignment. Actually there are **two** demo programs, one is using standard string class, and other is to be used with your myString class.

You will need to submit a header file with name "myString.h", which should contain the class myString with required functionality.

Make all the necessary functions in your class, so that the myStringdemo.cpp can successfully run, and give results, in the same way as the stringdemo.cpp gives.

Before you submit, write your name, roll number etc at the top of your header file as comments.

Bonus Marks:
Make the char array dynamically allocated, and increase the size of this array if the previous size is not enough. You can take the concept already demonstrated in dynamic arrays. The bonus carries 50% extra weight.