창의적 소프트웨어 프로그래밍 Lab 15

Handed out : Thu, Nov 17, 2022

Due : Mon, Nov 21, 2022, 23:59 (NO SCORE for late submissions!)

Submit your file on LMS.

1. Write a program that works as follows:
   A. Implement the operators of the following MyString class.
   B.

```
#ifndef __STRING_H__
#define __STRING_H__

// my_string.h - DO NOT modify this class definition
class MyString
{
public:
    // Implement operators
    MyString& operator=(const MyString& b);
    MyString operator+(const MyString& b);
    MyString operator*(const int b);
    friend    std::ostream&    operator<<(std::ostream&    out,    MyString&
my_string);
    friend    std::istream&    operator>>(std::istream&    in,    MyString&
my_string);

private:
    std::string str;
};

#endif //__STRING_H__
```

   C. DO NOT modify the given my_string.h. Do not add any other member functions or member variables, do not change the member access modifiers (public, private).
   D. This program should take user input repeatedly
   E. Input:
      i. 'new' – Create two MyString instances (named a, b) that is initialized to the following user inputs using the overloaded operator>>.
      ii. [object] + [object] – Print out the concatenated string of two MyString instances [object]. [object] can be 'a' or 'b' using the overloaded operator+.

iii. [object] * [integer] – Print out the string [object] [integer] times. [object] can be 'a' or 'b' using the overloaded operator*.

iv. 'quit' – Quit the program

F. Output: The output of the operations

G. All output should be printed using the overloaded operator<<.

H. Files to submit:

i. main.cpp - main() must be in this file.

ii. my_string.h – DO NOT modify it.

iii. my_string.cpp – Class MyString's member function definitions (implementations)

iv. A CMakeLists.txt to generate the executable

```
$ ./string
new
enter a
Hanyang
enter b
University
a * 3
HanyangHanyangHanyang
a + b
HanyangUniversity
quit
$
```

2. Write a program that works the same as the prob 1 program, using the following class MyString2 instead of class MyString.

   A. The goal is using a copy constructor instead of the assignment operator.

   B.
```
#ifndef __STRING_H__
#define __STRING_H__

// my_string2.h - DO NOT modify this class definition
class MyString2
{
public:
    // Add constructors you need, including copy constructor

    // Incorrect implementation of assignment operator.
    // Do not use the assignment operator.
    // Do not correct this because the goal is to prevent using the
assignment operator.
    MyString2& operator=(const MyString2& b) { return *this; };

    // Just use the same implementations for these operators
    MyString2 operator+(const MyString2& b);
    MyString2 operator*(const int b);
    friend std::ostream& operator<<(std::ostream& out, MyString2&
my_string);
    friend std::istream& operator>>(std::istream& in, MyString2&
my_string);

private:
    std::string str;
};

#endif //__STRING_H__
```

   C. DO NOT modify the given my_string2.h. Do not add any other member functions or member variables, do not change the member access modifiers (public, private). Do not correct the wrong implementation of the assignment operator.

   D. The input, output, and example are the same as prob 1.

   E. Files to submit:

      i. main.cpp - main() must be in this file.

      ii. my_string2.h – DO NOT modify it.

      iii. my_string2.cpp – Class MyString2's member function definitions (implementations)

iv. A CMakeLists.txt to generate the executable