Creative Software Programming, Assignment 11-1

Handed out: Nov 24, 2021

Due: 23:59, Nov 24, 2021 **(NO SCORE for late submissions!)**

- *Only files submitted by **git push to this course project at https://hconnect.hanyang.ac.kr** (<Year>_<Course no.>_<Class code>/<Year>_<Course no.>_<Student ID>.git) will be scored.*
- *Place your files under the directory structure **<Assignment name>/<Problem no.>/<your files>** just like the following example.*
  - ***DO NOT push CMake intermediate output files** (CMakeCache.txt, cmake_install.cmake, Makefile, CMakeFiles/*).*
  - ***DO NOT use cmake_minimum_required()** command in CMakeLists.txt.*

```
+ 2020_ITE0000_2019000001
  + 2-1/
    + 1/
      - 1.cpp
      - CMakeLists.txt
    + 2/
      - 2.cpp
      - CMakeLists.txt
    + …
```

- *The submission time is determined not when the commit is made **but when the git push is made**.*
- *Your files must be committed to the **master branch**. Otherwise, it will not be scored.*
- *Your program should output correct results even for inputs other than those used in the example.*
- *Basically, assignments are scored based on the output results. If it is not possible to check whether a requirement is implemented because the output is not correct, no score is given for the requirement, even if it is implemented internally. However, even if the output result is correct, no score is given for a requirement if the internal implementation does not satisfy the requirement.*

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 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. c= [object] + [object] – Concatenated string of two MyString instances [object] and assign it to object 'c'.

         1. [object] can be 'a' or 'b'. Use the overloaded operator+().

         2. There should be **a space** between the '+', 'c=' and each [object].

      iii. c= [object] * [integer] - Concatenate the string [object] [integer] times and assign it to object 'c'.

         1. [object] can be 'a' or 'b'. Use the overloaded operator*().

         2. There should be **a space** between the '+', 'c=' and each [object].

      iv. print [object] – Print out the string [object]. [object] can be 'a', 'b' or 'c'.

v. 'quit' – Quit the program

F. Output: The output of the operations

G. Given an arithmetic command (ex. "a + b"), you must assign the result of the operation into a new instance variable 'c'. DO NOT use assignment operator elsewhere.

i. For example, DO "**c=a+b;**". DO NOT something like "c=a; c+b;"

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

I. 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
c= a * 3
print c
HanyangHanyangHanyang
print a
Hanyang
print b
University
c= a + b
print a
Hanyang
print b
University
print c
HanyangUniversity
c= a + a
print c
```

```
HanyangHanyang
quit
$
```

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

A.

```
#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
    // Add an assignment operator

    // 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__
```

B. DO NOT modify the given my_string2.h **except constructors and an assignment operator** (as instructed in the comment). Do not add any other member functions or member variables, do not change the member access modifiers (public, private).

C. Input: Same as the prob1 except following additional input:

   i. set [option]: Set the option for setting values to the object 'c'.

      1. If [option] is set to 'copy', in the following '+' or '*' operations, the object 'c' should be constructed and initialized by a copy constructor. The code that prints the text "(copy constructor)" should be inside the copy constructor.

      2. If [option] is 'assign', in the following '+' or '*' operations, the object 'c' should be first constructed by a default constructor and then assigned by an assignment operator. The code that prints the text "(assignment operator)" should be inside the assignment operator.

3. The default option is 'copy'.

D. Output: The output of the operations, with "(copy constructor)" or "(assignment operator)" text for operations.

E. Note that "MyString2 c(a+b);" does not call the copy constructor due to return value optimization (copy elision) which is used by g++ by default. You need to do something like "MyString2 d(a+b); MyString2 c(d);" to call the copy constructor.

F. Files to submit:

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

ii. my_string2.h – DO NOT modify it except constructors and an assignment operator.

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

iv. A CMakeLists.txt to generate the executable

```
$ ./string2
new
enter a
Hanyang
enter b
University
c= a * 3
(copy constructor)
print c
HanyangHanyangHanyang
print a
Hanyang
print b
University
set assign
c= a + b
(assignment operator)
print a
Hanyang
print b
University
```

```
print c
HanyangUniversity
set copy
c= a + a
print c
(copy constructor)
HanyangHanyang
quit
$
```