Creative Software Programming, Assignment 11-2

Handed out: Nov 25, 2021

Due: 23:59, Nov 30, 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 constructor, destructor, and operators of the following MyVector class.

   B.
```
#ifndef __MY_VECTOR_H__
#define __MY_VECTOR_H__

// my_vector.h - DO NOT modify this class definition
class MyVector {
public:
       // Implement constructor & destructor
       MyVector();
       MyVector(int length);
       ~MyVector();

       // Implement operators
       MyVector operator+(const MyVector& b);
       MyVector operator-(const MyVector& b);
       MyVector operator+(const int b);
       MyVector operator-(const int b);
       friend std::ostream& operator<< (std::ostream& out, MyVector& b);
       friend std::istream& operator>> (std::istream& in, MyVector& b);

       // Add an additional constructor or operator if needed.

private:
       int length;
       double *a;
};

#endif // __MY_VECTOR_H__
```

   C. **DO NOT modify the given my_vector.h except to add an additional constructor or operator**. 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' [length] – Create two MyVector instances (named a, b) of length [length] and fill them with the following user inputs using the overloaded operator >>.

      ii. c= [object] [op] [object] – Apply [op] to two MyVector instances [object] and assign it to object 'c'.

         1. [op] can be '+' or '-', [object] can be 'a' or 'b'. Use the overloaded operators.

         2. There should be **a space** between each [object] or [op].

      iii. c= [object] [op] [integer] – Apply [op] to [object] and [integer] and assign it to object 'c'.

1. [op] can be '+' or '-', [object] can be 'a' or 'b'.
2. There should be **a space** between each [object] or [op].

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

v. 'quit' – Quit the program

F. Output: The output of the operations

G. Files to submit:

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

ii. my_vector.h – DO NOT modify it.

iii. my_vector.cpp – Class MyVector's member function definitions (implementations)

iv. A CMakeLists.txt to generate the executable

```
$ ./MyVector
new 10
enter a
2 3 4 5 6 7 8 9 10 11
enter b
3 4 6 2 7 8 9 3 4 1
print a
2 3 4 5 6 7 8 9 10 11
print b
3 4 6 2 7 8 9 3 4 1
c= a + 3
print a
2 3 4 5 6 7 8 9 10 11
print c
5 6 7 8 9 10 11 12 13 14
c= a + b
print b
3 4 6 2 7 8 9 3 4 1
print c
5 7 10 7 13 15 17 12 14 12
quit
$
```

2. Write a program that works the same as the prob 1 program, using the following class MyVector2 instead of class MyVector.

   A.
   ```
   #ifndef __MY_VECTOR_H__
   #define __MY_VECTOR_H__

   // my_vector2.h - DO NOT modify this class definition
   class MyVector2
   {
   public:
          // Implement constructor & destructor
          MyVector2();
          MyVector2(int length);
          MyVector2(const MyVector2& mv);
          ~MyVector2();

          MyVector2& operator=(const MyVector2& b);

       // Just use the same implementations for these operators
          MyVector2 operator+(const MyVector2& b);
          MyVector2 operator-(const MyVector2& b);
          MyVector2 operator+(const int b);
          MyVector2 operator-(const int b);
          friend std::ostream& operator<< (std::ostream& out, MyVector2& b);
          friend std::istream& operator>> (std::istream& in, MyVector2& b);

   private:
          int length;
          double *a;
   };

   #endif // __MY_VECTOR_H__
   ```

   B. **DO NOT modify the given my_vector2.h.** 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 "MyVector2 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 "MyVector2 d(a+b); MyVector2 c(d);" to call the copy constructor.

F. Files to submit:

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

ii. my_vector2.h – DO NOT modify it.

iii. my_vector2.cpp – Class MyVector's member function definitions (implementations)

iv. A CMakeLists.txt to generate the executable

```
$ ./MyVector
new 10
enter a
2 3 4 5 6 7 8 9 10 11
enter b
3 4 6 2 7 8 9 3 4 1
print a
2 3 4 5 6 7 8 9 10 11
print b
3 4 6 2 7 8 9 3 4 1
c= a + 3
(copy constructor)
print a
2 3 4 5 6 7 8 9 10 11
print c
5 6 7 8 9 10 11 12 13 14
set assign
c= a + b
(assignment operator)
print b
3 4 6 2 7 8 9 3 4 1
print c
5 7 10 7 13 15 17 12 14 12
quit
$
```