창의적 소프트웨어 프로그래밍 Lab 16

Handed out : Fri, Nov 18, 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 constructor, destructor, and operators of the following MyVector class.

```
#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 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);

private:
	int length;
	double *a;
};

#endif // __MY_VECTOR_H__
```

   B.

   C. DO NOT modify the given my_vector.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' [length] – Create two MyVector instances (named a, b) of length [length] and fill them with the following user inputs.

ii. [object] [op] [object] – Apply [op] to two MyVector instances [object]. [op] can be '+' or '-', [object] can be 'a' or 'b'.

iii. [object] [op] [integer] – Apply [op] to [object] and [integer]. [op] can be '+' or '-', [object] can be 'a' or 'b'.

iv. '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
a + 3
5 6 7 8 9 10 11 12 13 14
a + b
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. The goal is using a copy constructor instead of the assignment operator.

```
#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();

    // 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.
    MyVector2& operator=(const MyVector2& b) { return *this; };

    // 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.

   C. 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). 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_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