Homework Assignment 2

Due Date: 18:30, March 24, 2016

1 Problem 1

Write a program that performs matrix multiplication of two integer matrices.

Note: you must use **vector** to store both of the source and the result matrices. Any use of pointer-based arrays is forbidden.

1.1 Input

The first line of the input defines the number of rows and columns of two matrices (A and B) that are eligible for multiplication and is a space-delimited list of three integers: (1) the number of rows of matrix A, (2) the number of columns of matrix A and also the number of rows of matrix B, and (3) the number of columns of matrix B.

The second line of the input starts the content of the two matrices with one line per row and spaces between each column. Matrix A is firstly given, followed by a blank line and then by the content of matrix B.

1.2 Output

The output shows the content of the matrix (with one line per row and spaces between each column) that stores the result of matrix A multiplying matrix B.

1.3 Sample Input

2 3 2

1 0 2

-1 3 1

3 1

2 1

1 0

1.4 Sample Output

5 1

4 2

2 Problem 2

Complete the application (comprised of main.cpp, oop string.h, and oop string.cpp) that takes a line as input and outputs the input string by removing whitespace that appear at the beginning or end of the string, **reverting** the string, and **converting** all lower case letters to upper case letters. Note: You are permitted to modify **oop_string.cpp** only.

main.cpp

```
#include <iostream>
#include <string>
#include "oop_string.h"

using namespace std;

int main(int argc , char *argv[]){
  string input_line;
  while( getline( cin, input_line).eof() == false ){
    oop::trim(input_line);
    oop::reverse(input_line);
    oop::toUpperCase(input_line);
    cout << input_line << endl;
  }
  return 0;
}
```

oop_string.h

```
#ifndef _OOP_STRING_H
#define _OOP_STRING_H

#include<string>

namespace oop{
  void reverse( std::string &str);
  void toUpperCase( std::string &str);
  void trim( std::string &str);
};

#endif
```

oop_string.cpp

```
#include"oop_string.h"
#include<algorithm>
#include<functional>

namespace oop{
  void reverse( std::string &str){
  }
```

```
  void toUpperCase( std::string &str){
  }

  void trim( std::string &str){
  }
};
```

2.1 Sample Input

```
C++ is the most popular language .
Do you agree that ?
```

2.2 Sample Output

```
. EGAUGNAL RALUPOP TSOM EHT SI ++C
? TAHT EERGA UOY OD
```

3 Problem 3

You are to implement a simplified menu system. You will be given a food and a price of food descriptions. For each order description you are to compute the total price , according to the system. You may use **std::map** to implement the program. You only need to support elementary arithmetic of **integers** in this problem.

3.1 Input

The first line of input contains 2 positive integers: m , the kind of foods in the menu, and n , the number of order descriptions. m lines follow; each contains a word (a string of up to 16 lower-case letters) and a dollar value (a real number between 0 and 1,000,000). Following the order descriptions. Each order description consists of one or more lines of text; for your convenience the text has been converted to lower case and has no characters other than letters, numbers, and spaces. Each order is terminated by a line containing a period.

3.2 Output

For each order, output the corresponding menu computed as the sum of the price for all foods that appear in the description. Words that do not appear in the menu have a value of 0.

3.3 Sample Input

5 2

spaghetti 100

soup 80

rice 50

cake 55

coffee 66

please give my boss spaghetti soup and coffee my colleague need only coffee and i want cake

.

cake and coffee please

.

3.4 Sample Output

367

121

4 Problem 4

Given the preorder and inorder traversals of a binary tree, show the preorder,inorder and postorder traversal of the binary tree. For example, let A B D E F C G be the preorder traversal and D B F E A G C be the inorder traversal of a binary tree, the postorder traversal of the binary tree is D F E B G C A. **In this problem, you need to complete p4.cpp to output the answer .if you realize how it work in the code , TA will give you bonus.**

4.1 Input

The input contains *m* test cases. The first line of input contains a single integer (*m*) representing the number of test cases. Each test case begins with a line containing an integer n, $1 \leq n \leq 26$, indicating the number of nodes in a binary tree. The preorder and inorder traversals of the binary tree are given in the next two lines – the 1st line is the preorder traversal, and the 2nd line is the inorder traversal. The nodes of the binary tree are denoted by upper-case letters.

4.2 Output

Output should contain *m* lines if there are *m* test cases of input data. Each line contains the preorder,inorder and postorder traversals for each test case.

4.3 Sample Input

2

7

A B D E F C G

D B F E A G C

12

A B C D E F G H I J K L

L K J I H G F E D C B A

4.4 Sample Output

A B D E F C G

D B F E A G C

D F E B G C A

A B C D E F G H I J K L

L K J I H G F E D C B A

L K J I H G F E D C B A

5 How to submit the assignment?

1. Name the source code of each problem as following:

_ Problem 1: p1.cpp

_ Problem 2: **oop_string.cpp**

_ Problem 1: p3.cpp

_ Problem 1: p4.cpp

2. Do not rename the files or put them into any directory. Upload them directly to the **e-Campus (E3)** system. You will get no credit if you don't follow the rule. Note that the penalty for late homework is **15% per day**, and late homework will not be accepted after 3 days past the due date. In addition, homework assignments must be individual work. If I detect what I consider to be intentional plagiarism in any assignment, the assignment will receive **zero credit.**