Sabancı University

Faculty of Engineering and Natural Sciences
CS204 Advanced Programming
Fall 2022

Homework 1 – Matrix multiplication

Due: 16/10/2022, Sunday, 23:55

PLEASE NOTE:

Your program should be a robust one such that you have to consider all relevant programmer mistakes and extreme cases; you are expected to take actions accordingly!

You HAVE TO write down the code on your own.
You CAN NOT HELP any friend while coding.
Plagiarism will not be tolerated!!

Introduction

In this homework, you're going to find missing numbers to complete a matrix-vector multiplication operation. Your program will read three mathematical objects: matrix LHS (left-hand side for x), vector RHS (right-hand side for x), and vector RES (result). Matrix LHS and vector RHS will only contain values in the range [1, 9]. These three mathematical objects will satisfy the relation LHS × RHS = RES. However, the LHS matrix you will read will be missing some values. Your job is to find the missing values (which will be in the range [1, 9] inclusive) that will satisfy the mathematical relation LHS × RHS = RES. Here is a visualization of the operation:

$$\overset{\textbf{LHS}}{\begin{array}{|c|c|c|c|}\hline 4 & 5 & X_1 & X_2 \\ \hline X_3 & 5 & 8 & 2 \\ \hline 1 & X_4 & X_5 & X_6 \\ \hline X_7 & X_8 & X_9 & 8 \\ \hline \end{array}} \times \overset{\textbf{RHS}}{\begin{array}{|c|}\hline 4 \\ \hline 8 \\ \hline 9 \\ \hline 8 \\ \hline \end{array}} = \overset{\textbf{RES}}{\begin{array}{|c|}\hline 156 \\ \hline 132 \\ \hline 165 \\ \hline 227 \\ \hline \end{array}}$$

Figure 1: example input with nine missing values. You will read LHS, RHS, and RES, and find the missing values X_1-X_9 in LHS.

You will read LHS, RHS, and RES. LHS will be missing some values (indicated with the Xs). **Your job will be to find the missing values that makes the math correct.**

Since there may exist many possible solutions to each problem instance, there is one more condition for your solution to be accepted: if we take all the possible solutions to a problem instance, and for each instance concatenate its solution's values to form an integer, the solution you find must be the one with the smallest concatenated integer out of all the other solutions. This condition is explained further in the next subsection.

Minimum integer condition

For each problem instance, there will be many possible solutions. It is not enough for your program to find any one of these solutions, there is a specific solution that your program must return. It's the one with the smallest concatenated integer.

To elaborate, if we take each possible solution to a problem and concatenate its values from left to right, starting from the topmost row left-to-right and going downwards, we would form an integer. For example, in Figure 1, we would form this integer by concatenating values in the following order:

$$X_1X_2X_3X_4X_5X_6X_7X_8X_9$$

Given a problem instance, your program should return the solution with the smallest integer out of all the other solutions' concatenated integers.

For the example in Figure 1, there are exactly two possible solutions that satisfy LHS x RHS = RES in which all the missing values are in the range [1, 9] inclusive:

Solution 1: $X_1 = 4$, $X_2 = 8$, $X_3 = 1$, $X_4 = 7$, $X_5 = 9$, $X_6 = 3$, $X_7 = 7$, $X_8 = 9$, $X_9 = 7$
Solution 2: $X_1 = 4$, $X_2 = 8$, $X_3 = 1$, $\mathbf{X_4 = 1}$, $X_5 = 9$, $\mathbf{X_6 = 9}$, $X_7 = 7$, $X_8 = 9$, $X_9 = 7$

They are similar except for the values of $\mathbf{X_4}$ and $\mathbf{X_6}$. If we concatenate each solution's values into integers, we will get

Solution 1 integer = 481793797
Solution 2 integer = 481199797

Since *Solution 2 integer* < *Solution 1 integer*, the correct answer for the above problem is *Solution 2*. That is the answer that your program should return.

Program Flow

Your program will read three matrix files, find the solution, and print out the output to a text file. If at any point your program encounters an error due to a problem in the inputs, it will print out an error string to the output text file and will terminate.

Reading input

Your program starts by asking the user for the names of the files of RHS, LHS, and RES, as well as the name of the output file where it will print out its output, or if it encounters an error, the string “`Error`”.

It will then proceed to read the matrix and vector text files. If any of these files don’t exist, the program will print out the error message “`Error`” to the output file and terminate.

All three files will follow the same format. This format is explained in the next subsection. You must read these files, perform the necessary checks (explained later), and then proceed to find the solution.

Matrix file format

The file starts with a single line containing two numbers separated by a space:

`{num_rows} {num_columns}`

Where
`{num_rows}` = number of rows in the matrix, and
`{num_columns}` = number of columns.

After the first line, the file will contain `{num_rows}` lines exactly, each containing `{num_columns}` numbers. Each line represents a row, and the numbers in it are the values in the row’s columns. The values inside the cells of a matrix are either positive non-zero integers or “-1”, which indicates a missing value. For example, the LHS matrix in Figure 1 will be represented as the following text file:

`lhs.txt`

```
4 4
4 5 -1 -1
-1 5 8 2
1 -1 -1 -1
-1 -1 -1 8
```

And the RHS matrix in Figure 1 will be represented as the following text file:

`rhs.txt`

```
4 1
4
8
9
8
```

While reading each matrix, you must perform the checks listed below. If any check fails, you must print out the string “`Error`” to the output file and terminate the program.

Checks for LHS, RHS, and RES

1) There are exactly {num_rows} number rows in the text file.
2) All the rows have {num_columns} numbers.

Checks for RHS and RES

3) {num_columns} = 1 i.e., there is exactly one number in each row line .

Check for the LHS matrix only

4) All the number values in each row line are either positive integers in the range [1-9] inclusive or -1.

Check for RHS matrix only

5) All the number values in each row line are in the range [1-9] inclusive only.

Check for RES matrix only

6) All the number values in each row line are positive integers >=1.

Output format

While reading the inputs and carrying out the checks mentioned in the previous section, if you encounter any errors, the output file will simply contain the string "Error" as shown in the following output text file:

`output.txt`

```
Error
```

If no errors were found, your program will proceed normally and will find the values. Once you find the values, you will print out the values you find in the output text file. The format of the result will be as follows:

For each missing value, you will print out the following line:
`{row} {column} {value}`
where
`{row}` = the row of the missing value.
`{column}` = the column of the missing value.
`{value}` = the missing value itself.

You *must* print out the values as they occur in the matrix from left to right, from the topmost row to the bottom. In other words, print them in the same order used for forming the concatenated integer. For example, the solution to the problem in Figure 1 will be printed out as follows:

`output.txt`

```
0 2 4
0 3 8
```

```
1 0 1
2 1 1
2 2 9
2 3 9
3 0 7
3 1 9
3 2 7
```

Important: your console outputs should be clear and informative, but they don't have to match the sample runs exactly. However, your outputs to the text file **must match the format above exactly.** **Otherwise, your output will not be accepted.**

Hints

1. You can use `vector<vector<int>>` to represent the matrix.
2. You can use `vector<int>` or `vector<vector<int>>` to represent the vectors.
3. You can use `vector<int>` to represent locations in the matrix.
4. You might be tempted to try every possible integer in the range [1, 9] for every missing value in the LHS matrix. **We strongly advise you against that.** While grading, we will use very large matrices. If your code will try every possible combination for every value, your program will run out of time for these very big ouputs. Try to figure out the faster way to find the right solution.

Sample Runs

Sample 1

Input files:

lhs.txt
```
4 4
4 5 -1 -1
-1 5 8 2
1 -1 -1 -1
-1 -1 -1 8
```

rhs.txt
```
4 1
4
8
9
8
```

res.txt
```
4 1
156
132
165
227
```

Execution (the text in italics is input by the user):

```
Enter LHS matrix filename:inputs/tc1/lhs.txt
Enter RHS matrix filename:inputs/tc1/rhs.txt
Enter RES filename:inputs/tc1/res.txt
Enter output filename:output.txt
```

Output file:

output.txt
```
0 2 4
0 3 8
1 0 1
2 1 1
2 2 9
2 3 9
3 0 7
3 1 9
3 2 7
```

Sample 2

Input files:

lhs.txt

```
3 5
4 5 4 8 -1
5 8 2 1 7
-1 -1 9 -1 8
```

rhs.txt

```
5 1
8
9
8
3
9
```

res.txt

```
3 1
142
194
252
```

Execution (the text in italics is input by the user):

```
Enter LHS matrix filename:inputs/tc2/lhs.txt
Enter RHS matrix filename:inputs/tc2/rhs.txt
Enter RES filename:inputs/tc2/res.txt
Enter output filename:output.txt
```

Output file:

output.txt

```
0 4 1
2 0 3
2 1 6
2 3 10
```

Sample 3

lhs.txt

```
5 3
4 5 -1
-1 -1 5
5 -1 -1
6 3 -1
-1 2 5
```

rhs.txt

```
3 1
4
8
9
```

res.txt

```
5 1
92
85
164
66
93
```

Execution (the text in italics is input by the user):

```
Enter LHS matrix filename:inputs/tc3/lhs.txt
Enter RHS matrix filename:inputs/tc3/rhs.txt
Enter RES filename:inputs/tc3/res.txt
Enter output filename:output.txt
```

Output file:

output.txt

```
0 2 4
1 0 2
1 1 4
2 1 9
2 2 8
3 2 2
4 0 8
```

Sample 4

lhs.txt

```
4 4
4 5 -1 -1
-1 5 8 10
1 -1 -1 -1
-1 -1 -1
```

rhs.txt

```
4 2
4 1
8 2
9 11
```

res.txt

```
4 1
156
132
165
227
```

Execution (the text in italics is input by the user):

```
Enter LHS matrix filename:inputs/tc4/lhs.txt
Enter RHS matrix filename:inputs/tc4/rhs.txt
Enter RES filename:inputs/tc4/res.txt
Enter output filename:output.txt
Error!
```

Output file:

output.txt

```
Error
```

Note:
Multiple errors exist in these inputs:

1. LHS matrix has a missing number from the fourth column.
2. There are 3 rows in the RHS matrix instead of 4.
3. There are two columns in the RHS matrix even though it should only have 1.

Notice that we only need to report an error once. one error, not both of them.

Sample 5

Input files:

lhs.txt
```
4 4
4 5 -1 -1
-1 5 8 2
1 -1 -1 -1
-1 -1 -1 8
```
rhs.txt
```
4 1
4
8
9
8
```
res.txt
```
4 1
156
132
165
227
```

Execution (the text in italics is input by the user):

```
Enter LHS matrix filename:inputs/tc1/wrong_filename.txt
Enter RHS matrix filename:inputs/tc1/rhs.txt
Enter RES filename:inputs/tc1/res.txt
Enter output filename:output.txt
```

Output file:

output.txt
```
Error
```

Note:
The filename used for the LHS matrix is wrong (file doesn’t exist). Therefore, an error was reported.

Submission rules

In order to get full credit, your programs must be efficient and well presented, presence of any redundant computation or bad indentation, missing comments, or irrelevant comments are going to decrease your grades. You also have to use understandable identifier names and informative prompts. Modularity is also important; you have to use functions wherever needed and appropriate.

When we grade your homework we pay attention to these issues. Moreover, in order to observe the real performance of your codes, we will run your programs in Release mode and we will test your programs with very large test cases.

What and where to submit (PLEASE READ, IMPORTANT)

You must write your solution in C++. It'd be a good idea to write your name and last name in the program (as a comment line of course). Submission guidelines are below. Some parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade.

Name your cpp file that contains your `main` function as follows:

```
SUCourseUserName_YourLastname_YourName_HWnumber.cpp
```

Your SUCourse user name is your SUNet username that is used for your sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse username is cago, your first name is Çağlayan, and your last name is özbugsizkodyazaroğlu, then the file name must be:

```
cago_Ozbugsızkodyazaroglu_Caglayan_1.cpp
```

If your solution contains other code files, you don't have to change the other files' names. Only the file with the main function needs to have the naming convention above.

You shouldn't add any other files besides your code to the submission folder. In other words, don't add the example inputs and outputs to the submission.

Place all of your code files inside a folder named with the same naming convention shown above (without the .cpp extension, of course). So, the same student above would place all of his code inside a folder named:

```
cago_Ozbugsızkodyazaroglu_Caglayan_1
```

Compress this folder using a compression program such as WinZip or WinRAR. Please use "zip" compression. "rar", "7z" or any other compression mechanisms are NOT allowed. Our homework processing system only works with zip files. Therefore, make sure that the resulting compressed file has a zip extension. Check that your compressed file opens up correctly and it contains all your code files. You will receive no credits if your compressed folder does not expand or it does not contain the correct files. The name of the zip file follows the same convention. The zip file for the homework submission by the student mentioned above would be:

```
cago_Ozbugsızkodyazaroglu_Caglayan_1
```

Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.). Successful submission is one of the requirements of the homework. If for some reason you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!

Amro