Sabanci University

Faculty of Engineering and Natural Sciences
CS204 Advanced Programming
Spring 2016

Homework 1 – Subvector Search in Matrix

Due: 17/02/2016, Wednesday, 21:00

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 can NOT collaborate with your friends and discuss solutions. You have to write down the code on your own. Plagiarism will not be tolerated!

Introduction

The aim of this homework is to recall CS201 material and practice on matrices (two dimensional arrays/vectors). You are asked to find series of numbers in a 2D matrix of positive integers via basic search mechanisms and extract information out of it.

In this homework, you are going to implement a program to find out subvectors of integers of a matrix with a particular sum. First, your program is going to ask the file name that holds the matrix values. Then, the user is going to enter a ***sum*** value (a positive integer). Your program should check all rows and columns of the input matrix to find out consecutive elements of which the sum is the entered *sum* value. The user will be able to enter a new *sum* value to be searched until he/she enters an invalid input (anything other than a positive integer is invalid; for examples, 0, negative numbers, real numbers, characters, etc.). For each entered *sum*, your program should search the matrix both horizontally and vertically. For each occurrence of sum, your program should output the coordinates of the series of numbers, which make up the sum, together with their direction. If there is no occurrence, program should display an appropriate message.

Input

The program first prompts for the data file name. Then, it reads the file name from the keyboard and opens it. You should continue to read file name if the file is not opened successfully.

The file contains 2-dimentional matrix with positive integers. You have to consider an appropriate container to hold this 2-dimensional matrix in memory. You have to check any irregularities (anything other than a positive integer, unequal row sizes, empty rows, etc.) in the input file. To clarify further, each line of the given matrix can only contain space (blank) character and digit characters (i.e. the characters between 0 and 9). If there is problem with the file content, then your program must quit immediately with an appropriate error message. Please see the sample runs for correct and incorrect input files.

Program Flow, Outputs and the Search

After the matrix is read, the user is going to enter some *sum* values to be searched in the matrix. These *sum* values should be positive integers. Search results will be displayed right after each input *sum*. Your program should search several *sum* values one after another until an invalid *sum* is entered (*anything other than a positive integer*).

The search mechanism here is to find a consecutive horizontal or vertical series of integers in the matrix of which the sum is the entered *sum* value. The starting position of these series could be any element of the matrix. The number of integers in the series could be one or more than one. While searching a sum, you have to scan all the rows and columns and display the results in the required format (the coordinates of the endpoints of the series and the alignment) as shown in the sample runs. In the matrix, there might be two or more occurrences. In such a case, your program **must** display the results for each occurrence. Moreover, you have to display how many results you found for that particular sum. If the sum is not found, then an appropriate message must be shown.

Sample Runs

Some sample runs are given below, but these are not comprehensive and they do not consider all cases. Matrix files are provided with the homework package.

Sample Run 1
File: matrix1.txt

```
21 99 60 95 63
26 32 17 71 86
 7 64 56 84 36
87 43 98 63 37
70 71 26 62 57
 6 94 80 21 90
21 15  1 25 22
```

Output:

```
Please enter the name of the input file: data.txt
File cannot be opened.
Please enter the name of the input file again: a.txt
File cannot be opened.
Please enter the name of the input file again: matrix1.txt


Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 90
Found one horizontally from (5, 4) to (5, 4).
Found one vertically from (5, 4) to (5, 4).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 127
Found one horizontally from (2, 0) to (2, 2).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
```

```
(anything other than a positive integer to quit) : 60
Found one horizontally from (0, 2) to (0, 2).
Found one vertically from (0, 2) to (0, 2).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 195
Found one horizontally from (5, 1) to (5, 3).
Found one vertically from (0, 1) to (2, 1).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 100
Found one horizontally from (3, 3) to (3, 4).
Found one horizontally from (5, 0) to (5, 1).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 160
Total occurrences: 0

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 230
Found one vertically from (2, 3) to (5, 3).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 147
Found one vertically from (2, 3) to (3, 3).
Found one vertically from (4, 4) to (5, 4).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 190
Found one vertically from (1, 0) to (4, 0).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : abc
End of execution.
```

Sample Run 2
File: matrix2.txt

```
46 91 22
81  a 58
26  f 13
52 84 91
```

Output:

```
Please enter the name of the input file: file.txt
File cannot be opened.
Please enter the name of the input file again: input.txt
File cannot be opened.
Please enter the name of the input file again: matrix2.txt
The matrix in the input file is not in correct matrix format
or contains invalid elements.
End of execution.
```

Sample Run 3

File: matrix3.txt

```
12 87  1 33 89 89 31 78
14 48 23 14 19 62 11 95
16  6 20 69 55  6 52 99
77 99 85 72 36 46 39 60
38 84 60 84  9 31 66 45
25 73 97 71  8 77 28 68
20 69 67 86 85 88 34 23
43 87 94 53 70 71 75 13
```

Output:

```
Please enter the name of the input file: matrix3.txt


Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 212
Found one horizontally from (0, 2) to (0, 5).
Found one horizontally from (2, 4) to (2, 7).
Found one vertically from (1, 4) to (6, 4).
Total occurrences: 3

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 257
Total occurrences: 0

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 167
Found one vertically from (3, 6) to (6, 6).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 342
Found one horizontally from (0, 0) to (0, 6).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 285
Found one vertically from (1, 2) to (5, 2).
Total occurrences: 1
```

```
Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 506
Found one horizontally from (7, 0) to (7, 7).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 245
Found one vertically from (0, 0) to (7, 0).
Total occurrences: 1

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 227
Found one vertically from (3, 3) to (5, 3).
Found one vertically from (0, 6) to (5, 6).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : -1
End of execution.
```

Sample Run 4
File: matrix4.txt

```
74 36  1 59 97
65 77 72 26
 1 54 19
```

Output:

```
Please enter the name of the input file: matrix4.txt
The matrix in the input file is not in correct matrix format
or contains invalid elements.
End of execution.
```

Sample Run 5
File: matrix5.txt

```
52 65  5  5
37  6 74 10
31 68 55 41
75 58  2 71

72 49 11 34
45 91 14  3
```

Output:

```
Please enter the name of the input file: matrix5.txt
The matrix in the input file is not in correct matrix format
or contains invalid elements.
End of execution.
```

Sample run 6
File: matrix6.txt

```
 0 46 94
28  0  8
81 14 40
```

Output:

```
Please enter the name of the input file: matrix6.txt
The matrix in the input file is not in correct matrix format
or contains invalid elements.
End of execution.
```

Sample run 7
File: matrix7.txt

```
22 75 -3 89
-1 93 72 18
76 23 56 82
```

Output:

```
Please enter the name of the input file: matrix7.txt
The matrix in the input file is not in correct matrix format
or contains invalid elements.
End of execution.
```

Sample run 8
File: matrix8.txt

```
26 36 64 44 57 79
28  3 57 66 58 82
69 89 61 65 82 20
83 98 68 67 64  9
57 44 77 65 97 98
87 18 73 14 73 93
63 43 65 81  6 84
```

Output:

```
Please enter the name of the input file: matrix8.txt


Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 65
Found one horizontally from (2, 3) to (2, 3).
Found one horizontally from (4, 3) to (4, 3).
Found one horizontally from (6, 2) to (6, 2).
Found one vertically from (6, 2) to (6, 2).
Found one vertically from (2, 3) to (2, 3).
Found one vertically from (4, 3) to (4, 3).
Total occurrences: 6

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 66
Found one horizontally from (1, 3) to (1, 3).
```

```
Found one vertically from (1, 3) to (1, 3).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 67
Found one horizontally from (3, 3) to (3, 3).
Found one vertically from (3, 3) to (3, 3).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 68
Found one horizontally from (3, 2) to (3, 2).
Found one vertically from (3, 2) to (3, 2).
Total occurrences: 2

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : 73
Found one horizontally from (3, 4) to (3, 5).
Found one horizontally from (5, 2) to (5, 2).
Found one horizontally from (5, 4) to (5, 4).
Found one vertically from (5, 2) to (5, 2).
Found one vertically from (5, 4) to (5, 4).
Total occurrences: 5

Please enter a sum value that will be searched in the matrix
(anything other than a positive integer to quit) : no more
End of execution.
```

Some Important Rules

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

Moreover, in order to observe the real performance of your codes, we may run your programs in *Release* mode and **we may test your programs with very large test cases**. Of course, your program should work in *Debug* mode as well.

What and where to submit (PLEASE READ, IMPORTANT)

You should prepare (or at least test) your program using MS Visual Studio 2012 C++. We will use the standard C++ compiler and libraries of the abovementioned platform while testing your homework. It'd be a good idea to write your name and last name in the program (as a comment line of course).

Submissions 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

solution, project, cpp file that contains your main program using the following convention (the necessary file extensions such as .sln, .cpp, etc. are to be added to it):

"SUCourseUserName_YourLastname_YourName_HWnumber"

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

Cago_Ozbugsizkodyazaroglu_Caglayan_hw1

In some homework assignments, you may need to have more than one .cpp or .h files to submit. In this case add informative phrases after the hw number. However, do not add any other character or phrase to the file names.

Now let us explain which files will be included in the submitted package. Visual Studio 2012 will create two *debug* folders, one for the solution and the other one for the project. You should delete these two *debug* folders. Moreover, if you have run your program in release mode, Visual Studio may create *release* folders; you should delete these as well. Apart from these, Visual Studio 2012 creates a file extension of *.sdf* ; you will also delete this file. The remaining content of your solution folder is to be submitted after compression. Compress your solution and project folders using WINZIP or WINRAR programs. Please use "zip" compression. "rar" or another compression mechanism is NOT allowed. Our homework processing system works only 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 of the solution, project and source code files that belong to the latest version of your homework. Especially double-check that the zip file contains your cpp and (if any) header files that you wrote for the homework.

Moreover, we strongly recommend you to check whether your zip file will open up and run correctly. To do so, extract the content of your zip file to another location. Then, open your solution by clicking the file that has a file extension of .sln. Clean, build and run the solution; if there is no problem, you could submit your zip file. Please note that the deleted files/folders may be regenerated after you build and run your program; this is normal, but do not include them in the submitted zip file.

You will receive no credits if your compressed zip file does not expand or it does not contain the correct files. The naming convention of the zip file is the same. The name of the zip file should be as follows:

SUCourseUserName_YourLastname_YourName_HWnumber.zip

For example zubzipler_Zipleroglu_Zubeyir_hw1.zip is a valid name, but

Hw1_hoz_HasanOz.zip, HasanOzHoz.zip

are **NOT** valid names.

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!
Albert Levi, Ömer Mert Candan