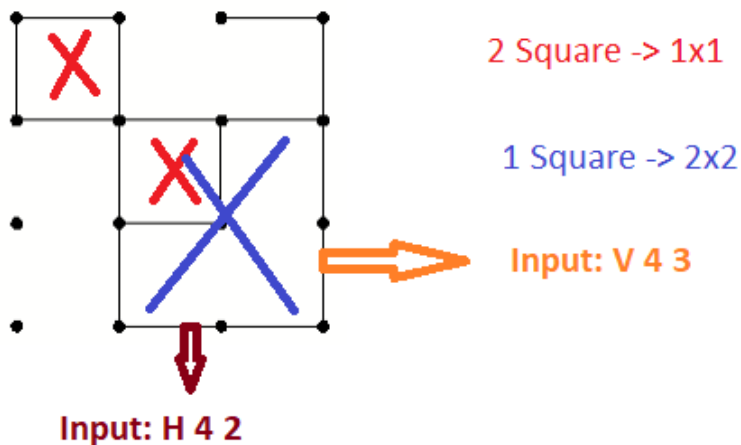


BOARD GAME - SQUARE COUNTING

***Summary of Project:**

In this assignment, we were asked to get inputs from an input file given to us and to create an output file based on these inputs. In the given input file, the dimensions of the board and how many lines were created were written first. In the next sections, the moves made by the person playing the game were written. The game board is created according to the number of points given.

The main purpose of the task was to take all coordinates properly and process them in the game, and then calculate the number of squares formed on the game board separately for each size.



(My illustration drawing)

***What is the problem that you will deal with?**

Actually homework was easy, but my extra time is gone because of my mistake.

My mistake was that I received the vertical inputs in the opposite direction of the order specified in the

homework. So first I have to get the j values, but instead I got the i values. So it took me a long time to understand this error and all of my input files that I created for testing purposes all gave false results.

$H\ i\ j$ indicates a horizontal line in row i which connects the dot in column j to the one to its right in column $j + 1$
or
 $V\ i\ j$ indicates a vertical line in column i which connects the dot in row j to the one below in row $j + 1$

$V\ 2\ 1$
 $V\ 2\ 2$
 $V\ 2\ 3$
 $V\ 3\ 2$

Apart from that, I thought about how to count the squares.

Apart from that, I did not encounter any errors or any problems that forced me.

***What model / method / algorithm that you will apply the problem?**

We didn't actually need to install a detailed algorithm. I created a new project and after thinking about 5-10

minutes, I started writing my code. I did not make any preliminary preparations before that.

What I intended was to create two 2D matrices for vertical and horizontal lines and set the initial values of these matrices to 0.

```
void resetRowAndColMatrix(int rowMatrix[SIZEMAX][SIZEMAX], int colMatrix[SIZEMAX][SIZEMAX], int sizeOfGameBoard) {
    int i, j;
    for (i = 0; i < sizeOfGameBoard; i++) {
        for (j = 0; j < sizeOfGameBoard; j++) {
            rowMatrix[i][j] = 0;
            colMatrix[i][j] = 0;
        }
    }
}
```

Then I determined which coordinates there are lines and if the line is vertical, I set the value of that coordinate of my vertical matrix to 1, if the line is horizontal, I did the same operations for that coordinate of my horizontal matrix.

```
for (i = 0; i < lineNumber; i++) {
    inputDataFile >> inputChar;
    if (inputChar == 'H') {
        inputDataFile >> tempI;
        inputDataFile >> tempJ;
        setRow(rowMatrix, tempI - 1, tempJ - 1);
    }
    if (inputChar == 'V') {
        inputDataFile >> tempJ;
        inputDataFile >> tempI;
        setCol(colMatrix, tempI - 1, tempJ - 1);
    }
}
```

I run the function upside down for vertical lines. In other words, i values and j values were replaced.

```
void setRow(int rowMatrix[SIZEMAX][SIZEMAX], int tempI, int tempJ) {  
    rowMatrix[tempI][tempJ] = 1;  
}  
void setCol(int colMatrix[SIZEMAX][SIZEMAX], int tempI, int tempJ) {  
    colMatrix[tempI][tempJ] = 1;  
}
```

The most challenging part of me was how to count the squares. I used a nested loop for this and it was a bit difficult to construct this loop.

```
for (k = 0; k < squareSize; k++) {  
    for (i = 0; i < dotNumber; i++) {  
        for (j = 0; j < dotNumber; j++) {  
            flagOfSquare = 1;  
            for (counterN = 0; counterN < squareSize; counterN++) {  
                if (rowMatrix[i][j + counterN] != 1 || rowMatrix[i + squareSize][j + counterN] != 1 || colMatrix[i + counterN][j] != 1 || colMatrix[i + counterN][j + squareSize] != 1) {  
                    flagOfSquare = 0;  
                    break;  
                }  
            }  
            if (flagOfSquare == 1) {  
                squareCounter[squareSize]++;  
            }  
        }  
    }  
    squareSize++;  
    if (squareSize == dotNumber)  
        break;  
}
```

I used 4 loops intertwined. I run the outermost loop separately for each frame size. All transactions were like this.

*** Application: Some print screen of your program etc.**

C:\Users\Casper\Desktop\FinalProject_Cpp\Debug\FinalProject_Cpp.exe

All transactions have been successfully completed. Please check the output file!

Press any key to continue . . .

it doesn't write anything on the console, it just seems to see if the process has been successfully completed.

input.txt - Not D

Dosya Düzen Biçi

```
4
16
H 1 1
H 1 3
H 2 1
H 2 2
H 2 3
H 3 2
H 4 2
H 4 3
V 1 1
V 2 1
V 2 2
V 2 3
V 3 2
V 4 1
V 4 2
V 4 3
2
3
H 1 1
H 2 1
V 2 1
```



output.txt - Not Defteri

Dosya Düzen Biçim Görünüm Yardım

Problem #1

2 square(s) of size 1.

1 square(s) of size 2.

Problem #2

No completed squares can be found!

(Controls of input and output files given in the homework file.)

7

37

H 1 1

H 4 3

H 5 3

H 6 3

V 3 4

V 3 5

V 4 4

H 4 4

V 5 4

V 5 5

H 6 4

V 1 1

V 2 1

H 2 1

V 1 2

V 1 3

V 1 4

V 1 5

H 1 2

H 1 3

H 1 4

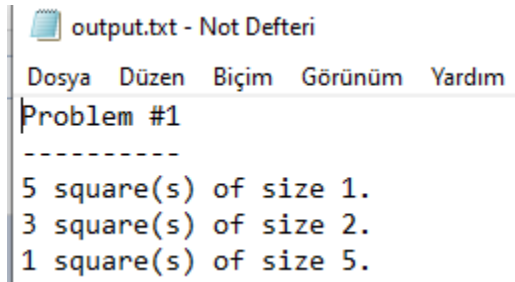
H 1 5

H 6 1

H 6 2

H 6 5

```
.. - .  
H 1 5  
H 6 1  
H 6 2  
H 6 5  
V 6 1  
V 6 2  
V 6 3  
V 6 4  
V 6 5  
H 5 5  
H 4 5  
H 3 1  
H 3 2  
V 3 1  
V 3 2  
V 4 5
```



```
output.txt - Not Defteri  
Dosya Düzen Biçim Görünüm Yardım  
Problem #1  
-----  
5 square(s) of size 1.  
3 square(s) of size 2.  
1 square(s) of size 5.
```

(Input file I created and control of its results.)

-THE END-

