HW2_P3 - Find sum of diagonals

Student name :	Carter Hawks
Student email :	ckh170000@utdallas.edu
Class name :	2336.001_F18
Submitted on :	Sep 16, 2018 07:35 am

solution.cpp

```
/* Problem Analysis
Given a 2d array of integers, we must sum up all of the values diagonally

This would be all of the corners, the middle (if there is one),
    and every value located diagonally adjacent to those corners.
For example:
X = element to be summed 0 = not summed
X 0 0 0 X
0 X 0 X 0
0 0 X 0 0
0 X 0 X 0
X 0 0 0 X

*/

/* Problem Design
1. First, we can start by calculating the sum of the numbers in the
    \ diagonal. Each of these coordinate pairs will have the same
    number for x as they do for y (0, 0)(1, 1)(2, 2)(3, 3) etc
2. Iterate through each element in the matrix and check if its x
    coordinate is equal to its y coordinate. If so, add it to the sum.
3. Next, we need to calculate the sum of the numbers in the / diagonal.
    We have to remember to exclude any number already calculated in
    the first diagonal (this would only be the center in an odd-length
    matrix, so ((N/2)-1, (N/2)-1) )
4. Iterate through each element in the matrix and check if its
    coordinate pair is equal to (N-1 - y, y). If so, add it to the
    current sum count variable.
5. Return the calculated sum.

*/

#include<stdlib.h>
#include<iostream>
using namespace std;

int findSumOfDiagonals(int *A, int N){
    int array [N][N]; // create array that doesnt have a pointer on it because im weak

    // create more usable array? ??
    for(int i = 0; i < N; i++){
        for(int j = 0; j < N; j++){
            array[i][j] = *(A + (i * N + j));
        }
    }
```

```
    int sum = 0;
    // calculate diagonals
    // \ diagonal, the easy one.
    // i == j means PROFIT
    // for an array N=5, (0,0)(1,1)(2,2)(3,3)(4,4)
    for (int i = 0; i < N; i++){
        for (int j = 0; j < N; j++){
            if(i == j){
                sum = sum + array[i][j];
            }
        }
    }

    // / diagonal, the hard one.
    // j == (N-1)-i means PROFIT-ER
    for (int i = 0; i < N; i++){
        for (int j = 0; j < N; j++){
            if(i == j){
                // do nothing please, you're in the middle.
            } else {
                // for an array N=5, (0,4)(1,3)(2,2)(3,1)(4,0)
                // ignore (2,2) with i==j comparison, because that is handled in the \ diagonal
                if((N-1) - i == j){
                    sum = sum + array[i][j];
                }
            }
        }
    }

    return sum;
}

//Your program will be evaluated by this main method and several test cases.
int main(){
    int i,j,N,*A;
    cin >> N;
    A = (int *) malloc(N *N *sizeof(int));
    for(i = 0; i < N; i++)
        for(j = 0; j < N; j++)
            cin >> A[i * N + j];
    cout << findSumOfDiagonals(A,N);
    return 0;
}
```

Name

Custom test case

Input

2 1 2 2 1

Output (Lines:2)

6

Expected Output (Lines:0)

Status

NA

Name
Custom test case

Input
5 1 2 3 4 6 5 6 7 8 5 9 8 7 6 4 5 4 3 2 3 9 6 5 3 1

Output (Lines:2)
44

Expected Output (Lines:0)

Status
NA

Name
Custom test case

Input
4 1 2 3 4 5 6 7 8 9 8 7 6 5 4 3 2

Output (Lines:2)
40

Expected Output (Lines:0)

Status
NA

Name
Default

Input
3 5 8 1 2 3 4 2 4 9

Output (Lines:2)
20

Expected Output (Lines:1)
20

Status
Pass
