CSC131: Computational Thinking
Homework Assignment # 5

Due: Beginning of class on Friday April 17th, 2015.

The game of "Jump It" consists of a board with n positive integers in a row except for the first column, which always contains zero. These numbers represent the cost to enter each column. Here is a sample game board where n is 6:

0	3	80	6	57	10

The object of the game is to move from the first column to the last column in the lowest total cost. The number in each column represents the cost to enter that column. You always start the game in the first column and have two types of moves. You can either move to the adjacent column or jump over the adjacent column to land two columns over. The cost of a game is the sum of the costs of the visited columns.

In the board shown above, there are several ways to get to the end. Starting in the first column, our cost so far is 0. We could jump to 80, then jump to 57, then move to 0 for a total of 80 + 57 + 10 = 147. However, a cheaper path would be to move to 3, jump to 6, then jump to 10, for a total cost of 3 + 6 + 10 = 19.

Write a **recursive** solution to this problem that computes the cheapest cost of the game and outputs this value for an arbitrary large game board represented as a list. Because the program can take long time to run on large boards, you can assume that a board size is at most 50 columns. Your program doesn't have to output the actual sequence of jumps, only the cheapest cost of this sequence.

Your program must read input from a text file named **input.dat** and must send output to stdout (the computer's screen). The input file consists of a sequence of lines, where each line corresponds to a board. The numbers on a line are the costs to enter the columns on the board. For example, the above board is represented in the input file as

```
0 3 80 6 57 10
```

Sample input file is as follows:

```
0 3 80 6 57 10
0 98 7 44 25 3 5 85 46 4
0 57 59 83 9 42 70
0 20 49 96 53 7 43 77
0 24 17 15 61 49 61 8 65 43 26 99 7 57 97 50 93 6 82 52
```

The corresponding output is as follows:

```
19
87
138
```

186
330

Please give your program the name **hw5.py**. You may or may not use a class for your solution. If you do not use a class, make sure to write your code in functions. No code should be written outside of a function or a method. Use comments to document and explain your code where needed. Make sure to upload an electronic copy of your source code in a folder named HW\hw5 in your CSC 131 TRACE folder. Also make sure to turn in a **stapled** hard copy of the source code in class on the due date. Write your name and TRACE folder name at the top of your file. **For full credit, your program must follow the problem specifications precisely. A program that does not run will not receive a grade above 60.**