National University of Computer & Emerging Sciences

Karachi Campus

Artificial Intelligence

Programming Assignment #1

Instructions ***Each Problem of 15 points***

Due Date: The assignment is due by February 25, 2014 till midnight.

Submission: The assignment has to be submitted via slate website submission. You must submit the source code files with proper naming convention for example (Assignment No. 1 Problem No. 1) you should give CS401-K1xxxxx-A1P1.cpp. You should copies all questions for the assignment in a single folder (named as your id e.g. K1xxxxx) and zipped it before uploading to slate.

Sample Input and Output files: The sample input and output files are available from course website at slate. Make sure that you have tested your programs against all the available input files and EXACT output file is produced.

Problem 1 United Federation of Planets (UFP)

The Zozo Planets is planning to host Year 2050 meeting of UFP; there are several planets to attend this meeting. Zozo scientists have designed a space ship for providing travel to all the participants from space station Salyut 2000. The space ship has very limited capacity; it can only hold 2 species at max. All the participants have to reach Salyut 2000 at 00:00, Jan 01, 2050. From there the space ship will carry them to the actual event venue. The space ship is quite extraordinary in itself. The speed of the ship is actual the height of the tallest of the passenger (species) in it. The Zozo scientist hire you as a computer programmer to design a computer program that calculate the optimal time to transfer all the participants from Salyut to the event venue. An optimal solution to the problem is the sequence of space ship -trips that will bring the participants from the salyut 2000 to the meeting venue in a minimum amount of total time. The program must compute the minimum time required for all the participants reaching for the meeting together with the total number of trips, and the list of passenger in each trip. In addition, the program should print out the sequence of trips that lead to the optimal solution.

Example: if there were four species from four different planets like:

Alastria – A Delta Quadrant planet with a binary star about 40,000 light-years from Sikaris, accessed via the spatial trajector

Brax – Planet in the Gamma Quadrant whose inhabitants call Q the "God of Lies".

Cait - Planet in the 15 Lyncis star system; the homeworld of the ailuroid (cat-like) Caitian species

Delta IV - Homeworld of the Deltan species

We call them ABCD, having height 10, 5, 1, 2 then there will be a solution for the transportation trip that will take 19 units of time in total, and consists of following trips.

1. C and D will go first it will take 2 unit of time.
2. C will come back; it will take 1 unit of time.
3. B and C will go in the second trip, it will take 5 unit of time.
4. C will come back; it will take again 1 unit of time.
5. A and C will go in the third trip, it will take 10 unit of time.

There are 15 planets that are participating in the meeting. Each one is represented by English alphabet, staring from A to J. The height of each species 10,5,1,2,6,7,3,4,4,and 5 respectively.

Input/Output

The input contains several testcases. Each is specified by 1 to 10 unsigned integers that represent the number of individuals from each planet. The input is terminated by 0 for all.
For each test case, print the best possible description of the trips, as provided in the example. The first line of the output contains the optimal unit of the time, then from the next line each trip is describe with the individual and the time it takes.

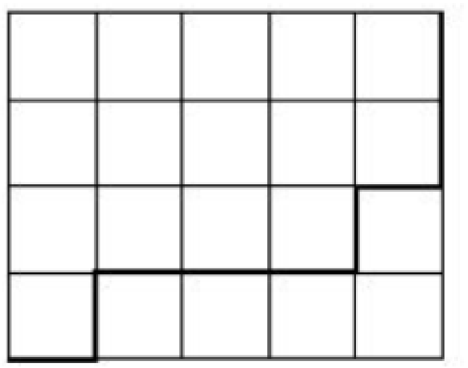
Example

A1P1in1.txt	A1P1out1.txt

1 0 0 0 0 0 0 0 1 0
1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
	10
	19
	0

Problem 2   Raju’s Patterns

Raju in his Mathematics class is bored again. Obviously, He knows all the stuff that his teacher is delivering in this lecture. On old mathematics note book of square lines, Raju start drawing lines, suddenly he has got this idea. Raju asked himself, that if he considered a small portion of the page say a rectangular grid of size n*m and If he sketch a line from the pencil starting from the lower left corner of the grid, and move the pencil to the upper right corner, taking care that it stays on the lines and moves only to the right or up. The complete procedure produces the following result:



Seeing the pattern Raju fascinated. The next question triggered in his mind is exactly how many such patterns he can have in a grid on n*m. You are required to write a computer program that when given a grid of size n*m calculates how many such pattern can be found.

Input/Output

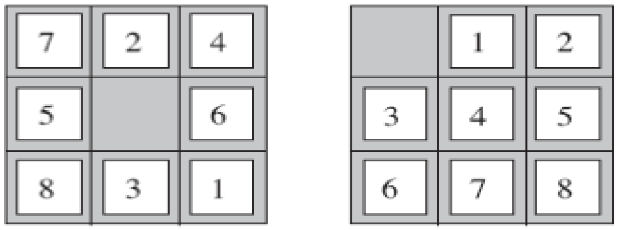
The input contains several testcases. Each is specified by two unsigned 32-bit integers n and m, denoting the size of the rectangle. As you can observe, the number of lines of the corresponding grid is one more in each dimension. Input is terminated by n=m=0.
For each test case output on a line the number of different art works that can be generated using the procedure described above. That is, how many paths are there on a grid where each step of the path consists of moving one unit to the right or one unit up? You may safely assume that this number fits into a 32-bit unsigned integer.

Example:

A1P2in1.txt 5 4 1 1 0 0	A1P2out1.txt 126 2

Problem 3   8-Puzzle

The 8-puzzle is a smaller version of the slightly better known 15-puzzle. The puzzle consists of an area divided into a grid, 3 by 3. On each grid square is a tile, expect for one square which remains empty. A tile that is next to the empty grid square can be moved into the empty space, leaving its previous position empty in turn. Tiles are numbered, 1 thru 8 for the 8-puzzle, so that each tile can be uniquely identified. The aim of the puzzle is to achieve a given configuration of tiles from a given (different-initial) configuration by sliding the individual tiles around the grid as described above. For example: The following diagram shows an initial and goal state of the problem.



Start State   Goal State

Considering the above start state, there are four possibilities, (1) we can move a 6 to the left (2) we can move 5 to right, (3) we can move 3 up and (4) we can move 2 down. These are the four possible operators <UP, DOWN, LEFT and RIGHT>. In this problem you need to implement basic operators action processing for 8-puzzle. The next programming assignment is based on the code that you developed in this. For example if the following board position are given to you as initial position, and the goal position.

```
   1  3    1  2  3
4  2  5    4  5  6
7  8  6    7  8

initial     goal
```

There is a move sequence that achieve this goal in three steps

```
   1  3        1     3        1  2  3        1  2  3        1  2  3
4  2  5   =>   4  2  5   =>   4     5   =>   4  5      =>   4  5  6
7  8  6        7  8  6        7  8  6        7  8  6        7  8

initial        1 left          2 up          5 left          goal
```

All you need to implement the operators move for the game. That is <UP, DOWN, LEFT and RIGHT>.

Input/Output

The input contains a 3 x 3 board, and from the next line there are move sequences per line. The output file gives the final board position after applying the given move. If the move is not possible just write “Invalid Move”

Example

A1P3in1.txt	A1P3out1.txt
0 1 3 4 2 5 7 8 6 1 left 2 up 5 left 5 left	1 0 3 4 2 5 7 8 6 1 2 3 4 0 5 7 8 6 1 2 3 4 5 0 7 8 6 Invalid Move

<The end>