ECOO 2015

Programming Contest Questions

Regional Competition (Round 2)

April 25, 2015

Sheridan | Get Creative

Questions made possible in part through the support of the Sheridan College Faculty of Applied Science and Technology.

Problem 1: The Interlace Cypher

The Interlace Cypher is a method of encrypting text by scrambling the letters in a message. The letters are rearranged so that the original words are interlaced by alternating letters from each word. Then the resulting letter set is broken up into "words" that match the lengths of the words in the original message:

```
the interlace cypher   →   tic hnyetpehr elrace
```

The first three letters of the encrypted message ("tic") come from the first letters of each word. The next three letters "hny" come from the second letters of each word. The letters "etp" are the third letter of each word, but then we run out of letters from the word "the", so the letters are in pairs after that: the letters "eh" are the fourth letters of the second and third words, and so on. The last three letters "ace" are all from the middle word in the original message.

DATA11.txt (DATA12.txt for the second try) will contain 10 test cases. Each test case consists of 2 lines. The first line will contain either the word "encode" or the word "decode" (all lower case). The next line will contain a message consisting entirely of lower case letters and space characters. You should output one line representing the Interlace Cypher encoding or decoding of that message (according to the instruction on the first line).

Note that the sample input below only contains 2 test cases, but the real data files will contain 10.

Sample Input

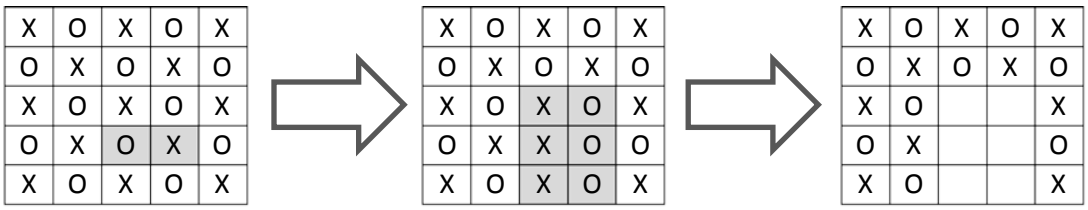
```
encode
whatever you do dont put the blame on you
decode
biotr yy lt nha eeae iaam nhhe
```

Sample Output

```
wyddptbo yho oo uhln oau nte auttm ee ver
blame it on the rain yeah yeah
```

Problem 2: So So

In the game of XOXO (pronounced “So So”) the player starts with a large grid that is filled with square tiles, each marked with either an X or an O. On each turn, the player can swap any two vertically or horizontally adjacent tiles as long as the swap produces a vertical or horizontal line of 3 or more X’s or 3 or more O’s. Any tiles involved in such a row are removed from the board. A single swap may produce multiple lines, in which case all the tiles that are part of one of the new lines are removed.



If you manage to remove all the tiles through repeated swapping like this, you have solved the board and you win the game.

DATA21.txt (DATA22.txt for the second try) will contain 10 test cases. Each test case starts with two integers R and C on a single line separated by a space (R > 0, C > 0, R×C ≤ 30). These numbers indicate the number of rows and columns in each of the boards that follow. The next 5×R lines will contain 5 different starting boards, one after another, each represented as R rows of C characters. Each character will either be a capital X or a capital letter O (not zero) and there will be no horizontal or vertical lines of X's or O's greater than length 2 anywhere on any starting board. Your job is to determine whether or not it is possible to solve each board and then output an S (solvable) or an N (not solvable) for each board. The 5 characters for each test case must appear on a single line with no spaces separating them.

Note that the sample input below contains only 1 test case, but the real data files will contain 10.

Sample Input

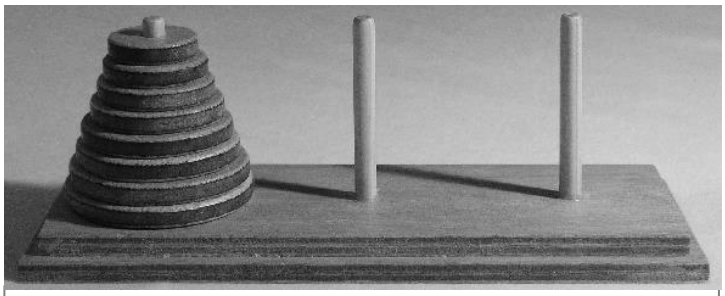
```
4 4
XX0X
00XX
XX00
X0X0
00X0
0XX0
X00X
00XX
00X0
XX0X
0X00
00X0
00XX
XX0X
0XX0
X0XX
X00X
0X00
00X0
XX0X
```

Sample Output

```
NSSSS
```

Problem 3: Lucas' Other Tower

The "Lucas' Tower" problem (also known as the "Tower of Hanoi") is a puzzle that consists of three vertical pins arranged in a row and one or more disks with holes in their centres so that they can slide onto the pins. No two disks are the same size. The classic version of this puzzle starts with all the disks on the leftmost pin in order of size with the smallest on top, as shown below.



Start Position for the Lucas' Tower problem with 8 disks (source: Wikipedia)

The goal is to move the entire stack of disks onto the rightmost pin so that they are stacked in the same order as they were originally, following the rules below:

1. You can only move one disk at a time.
2. You can only move the top disk from one stack onto the top of another stack or onto an empty pin.
3. You can never place a disk on top of a smaller disk.

The "Lucas' Other Tower" problem is less well known. (Actually, we made it up.) In this problem the start position, goal and number of pins are the same. The disks are all of different sizes, but those sizes are integer diameters from 1 to N centimetres, where N is the number of disks in the puzzle. The first two rules are the same as before but there is also a positive non-zero integer T known as the "tolerance factor" which is used to make the third rule more or less restrictive as shown in the rules below:

1. You can only move one disk at a time.
2. You can only move the top disk from one stack onto the top of another stack or onto an empty pin.
3. You can never place a disk on top of another disk that is more than T centimetres larger or smaller.

DATA31.txt (DATA32.txt for the second try) will contain 10 test cases. Each test case consists of two integers N and T on a single line separated by a space ($3 \le N \le 1000000$, $1 \le T \le 10000$). These numbers define an instance of the Lucas' Other Tower problem where N is the number of disks and T is the tolerance factor. Your task is to report the minimum number of moves necessary to solve this instance of the problem.

Note that the sample input below only contains 1 test case, but the real data files will contain 10.

Sample Input

```
4 2
```

Sample Output

```
7
```

Problem 4: Rectangle Roundup

You have been given a set of rectangular and square tiles. Your job is to try and put this set of tiles together to form larger rectangles. The rectangles must be completely filled in. They can't be hollow and they can't contain any holes. You have to use all of the tiles you have been given for each larger rectangle you form.

DATA41.txt (DATA42.txt for the second try) will contain 10 test cases. The first line of each test case will consist of a single integer N on a line by itself ($1 \le N \le 10$). This will be followed by N lines, each containing two integers S_1 and S_2 representing the two side lengths of one of your tiles in centimetres ($1 \le S_1, S_2 \le 5$). The total area of all the tiles combined for each test case will be no more than 30 square centimetres. For each test case you should output a single integer representing the largest perimeter it is possible to create with the tiles when you make rectangles following the rules set out above. If it is not possible to create any rectangles with the tiles you have been given, you should output the words "Not Possible" with both words capitalized.

Note that the sample input below only contains 3 test cases, but the real data files will contain 10.

Sample Input

```
3
3 2
2 2
1 2
4
1 1
1 1
1 1
1 1
3
3 2
1 1
1 3
```

Sample Output

```
16
10
Not Possible
```

Question Development Team
Sam Scott (Sheridan College)
Kevin Forest (Sheridan College)
Greg Reid (St. Francis Xavier Secondary School, Mississauga)
Dino Baron (University of Waterloo)
John Ketelaars (ECOO-CS Communications)
David Stermole (ECOO-CS President)
Thanks also to Lukasz Wawrzyniak (Sheridan College)