ECOO 2017

Programming Contest Questions

Final Competition (Round 3)

May 13, 2017

Problem 1: Baker Brie

Author: Andrew Seidel

Baker Brie is holding a celebration for being in business for 13 years, and having opened its 130th franchise. *Baker Brie* wants to congratulate franchises that have performed well throughout the years. *Baker Brie* also wants to congratulate everyone for performing well on certain days of the year!

Baker Brie wants to offer congratulations as follows:

- If, in a single day, all franchises combined sell an amount of baked goods that is equivalent to a multiple of a baker's dozen (13), then all franchises will receive a bonus.
- If an individual franchise, throughout its entire existence, has sold an amount of baked goods that is equivalent to a multiple of a baker's dozen (13), then that franchise will receive a bonus.

Input Specifications

DATA11.txt (DATA12.txt for the second try) will contain 10 data sets. On the first line of each dataset there will be the values **F** and **D** separated by a space where **F** ($4 \le F \le 130$) represents the number of franchises that Baker Brie has, and **D** ($2 \le D \le 4745$) represents the number of days of information. On the next **D** lines, there will be **F** integers separated by spaces (each in the range 1 through 13000), such that the i^{th} integer on line *j* represents the number of baked goods sold by franchise *i* on day *j*.

Output Specifications

You must determine, both for each day (across all franchises) and for each franchise (across all days), whether or not the number of baked goods sold is a multiple of 13. If it is, you need to track how many baker's dozens were sold. Report the total number of baker's dozens as a single integer on its own line.

Sample Input (2 test cases shown)

```
4 5
4 3 2 4
3 3 2 1
8 2 4 1
2 2 4 3
9 3 2 3
4 2
4 4 4 1
1 1 3 4
```

Sample Output

```
4
1
```

Explanation of Sample Output

In the first case, the first franchise sold a total of 26 baked goods (which is 2 baker's dozens), the second franchise sold a total of 13 baked goods (which is 1 baker's dozen), and finally, all franchises together sold 13 baked goods on the first day (which is 1 baker's dozen). This totals to 4 baker's dozen.

For the second dataset, no franchises made enough baked goods on their own, but there was a single baker's dozen created among them all on the first day. This totals to 1 baker's dozen.

Problem 2: Family Trees

Author: Reyno Tilikaynen

Family trees are trees that show relationships between family members. They begin with a root ancestor and show that ancestor's children, then every child's children, and so on. For example, Bob, the root ancestor, can have two daughters Alice and Eve. Alice can then have two children Jenna and Brian, and Eve can have one daughter Sarah. To help find people in the tree, we give each family member a family ID, formatted as a series of integers separated by dots (e.g. 0.1, 0.2.3, 0.5.1.7 and so on).

A family ID is either:

- 0, which represents the root ancestor, or
- X.y, where X is a valid family ID, and where this ID represents the y^{th} child of X.

For the example above, the family IDs are:

```
Bob:       0
     Alice      0.1
          Jenna      0.1.1
          Brian      0.1.2
     Eve        0.2
          Sarah      0.2.1
```

Family IDs can give you an idea of how big a family is. For example, if you know that someone has the ID 0.2.3, then you know there are family members with IDs 0, 0.1, 0.2, 0.2.1, and 0.2.2. Given a list of family IDs, figure out the smallest possible size of the family.

Input Specifications

DATA21.txt (DATA22.txt for the second try) will contain 10 test cases. Each test case starts with an integer **N** (1 ≤ **N** ≤ 100,000). The next **N** lines each contain a family ID. For 50% of cases, **N** ≤ 100, and the total input size will not exceed 2,000 characters.

Output Specifications

For each test case, your program should output the minimum size of the family, modulo 1,000,000,007.[1]

Sample Input (2 families shown)

```
1
0.2.3
3
0
0.2.1
0.1.1
```

Sample Output

```
6
5
```

[1] This means that if the size of the family is 999999999999 you should output 999993006, the remainder after dividing 999999999999 by 1000000007.

Problem 3: Region Selection

Author: Reyno Tilikaynen

The semi-final of the OCEE provincial competition happens at two different locations in Ontario. Ontario is a big place, so the locations need to be carefully chosen to accommodate the participants as best as possible.

In particular, a school's travel cost to the competition is equal to the square of the distance between the school and the nearest semi-final location. An optimal location selection would minimize the sum of these squared distances for every school.

Given the locations of all participating schools, can you determine the optimal placements of the two semi-finals?

Input Specifications

DATA31.txt (DATA32.txt for the second try) will contain 10 test cases. Each case begins with an integer **N** which represents the number of schools competing ($1 \leq$ **N** ≤ 100). **N** lines follow, each containing two integers **X** and **Y**, representing the location of a school ($1 \leq$ **X, Y** $\leq 1{,}000$). No two schools will be at the same location.

For 40% of the cases, **N**, **X, Y** ≤ 40.

Output Specifications

For each test case, output the minimum total sum of every school's travel costs, rounded to the nearest integer. For this question, there are part marks available if you get close to the optimal answer.

Sample Input (2 test cases shown)

```
3
1 1
2 2
3 3
6
1 1
2 1
3 1
1 4
2 4
3 4
```

Sample Output

```
1
4
```

Problem 4: Ice Cream Beach

Author: Reyno Tilikaynen

Harnessing your entrepreneurial spirit, you have decided to found a start-up. Your start-up, the first of its kind, will sell ice cream at the local beach. Every day, the beach gets **N** visitors which sit at specific locations on the beach. The visitors would all love to buy ice cream, but they don't like walking around in the sun. A visitor's reluctance to buy ice cream is calculated by multiplying their distance from the nearest ice cream stand by their reluctance factor: a number unique to each visitor.

You have access to **M** ice cream stands, which you can place anywhere on the beach. Moreover, you know where every visitor will sit and you know their reluctance factor. Since you don't want your start-up to sink, you'd like to carefully position your stands to minimize the total reluctance of all the visitors. Can you figure out the minimum total reluctance that you can achieve?

Input Specifications

DATA41.txt (DATA42.txt for the second try) will contain 10 test cases. Each test case starts with two integers **N, M** ($1 \le$ **N** ≤ 4000, $1 \le$ **M** ≤ 20) where **N** is the number of visitors and **M** is the number of ice cream stands. The next **N** lines describe the visitors coming to the beach. Each line contains two integers **X** and **F**, where **X** represents the location of the visitor on the beach and **F** represents the reluctance factor of that visitor ($1 \le$ **X, F** $\le 10^6$). Visitors will be listed in ascending order of **X** and no two visitors will be at the same location. Since the beach is long and thin, it can be thought of as being one dimensional.

For 30% of the cases, **N** ≤ 10. For 60% of the cases, **N** ≤ 100.

Output Specifications

For each test case, your program should output the minimum total reluctance of all the visitors, modulo 1,000,000,007.[2]

Sample Input (3 test cases shown)

```
2 1
10 10
20 10
2 2
10 10
20 10
4 2
1 10000
100 10
150 10
200 10
```

Sample Output

```
100
0
1000
```

See next page for an explanation of this output.

[2] This means that if the minimum total reluctance is 999999999999 you should output 999993006, the remainder after dividing 999999999999 by 1000000007.

Explanation of Sample Output

In the first test case, it's best to put your only ice cream stand anywhere between positions 10 and 20. If we were to put it at position 15, then each visitor would have a reluctance of 5*10 = 50, for a total reluctance of 100.

In the second test case, you can perfectly accommodate both guests by putting your two carts at locations 10 and 20.

In the last test case, it's best to put one cart at location 1 to appease the incredibly reluctant visitor and put your second cart at location 150.