Welcome to Casino Challenge24, where the real high rollers play, where Lady Fortune can smile on anybody she deems worthy!
As you walk in through the crimson velvet curtains, a hall reveals itself, filled with exquisite beauty, vibrating nightlife and the thrill of excitement.

Take a seat, and enjoy the Game!

1. Imageprocessing

In the casino an automatic roulette table is going to be installed, where a camera above the table determines the placed bets based on the image it sees. The table only accepts 20 HUF coins, however there are always people trying to cheat and use other coins, counterfeit money printed on paper, or even show up money on their mobile phones. As head of the engineering team, write an image processing software for the casino that recognises how many real 20 HUF coins have been placed on the table. Remember that it has to recognise a coin independent of its orientation. (I.e. If it is heads up and tails down or vice versa.) It may happen that the guests start to throw money on the table resulting in (not completely) overlapping coins or they might even throw litter on the table because of which the background changes. Also the picture of the camera may not always be perfect, namely the pattern on the money might not be well visible, but the software has to be functional nevertheless. Thanks to the Hungarian money system 20 HUF coins have a unique size, thus they are always identifiable by their dimensions. Two example shots taken by the camera are given to you for calibration where the number of 20 HUF coins on the pictures are given to you. There are ten more pictures on which you have to determine the number of coins and the closer the software guesses, the more points You will get.

Example 1:

Initial situation:

Examle output 1:

2

Example 2:

Initial  situation:

Examle output 2:

1

Input files:
http://ch24.org/static/docs/input1.zip

2. IP Addresses

An IP Address (IPv4) consists of 4 numbers which are all between 0 and 255. In this problem however, we are dealing with "Extended IP Addresses" which consist of K such numbers.

Given a string S containing only digits and a number K, your task is to count how many valid "Extended IP Addresses" can be formed.

Because the number of different "Extended IP Addresses" can be very large, return the result modulo 1000000007.

An Extended IP Address is valid if:

- it consists of exactly K numbers
- each numbers is between 0 and 255, inclusive
- a number cannot have leading zeroes

Examples:

1. S: "1234567", K: 3 -> 1 (valid IP addresses: "123.45.67")
2. S: "100111", K: 3 -> 3 (valid IP addresses: "100.1.11", "100.11.1", "10.0.111")
3. S: "345678", K: 2 -> 0 (it is not possible to form a valid IP Address with two numbers)

Input parameters:

S - the string that should be split into valid "Extended IP Addresses"
K - how many numbers should there be in the IP Addresses

Constraints:

S will contain between 1 and 100, inclusive
K will be between 1 and 100, inclusive

Return value:

int - the number of different "Extended IP Addresses"

Input files:
http://ch24.org/static/docs/input2.zip

3. LazyGuards

You are stuck in a maze and you need to reach the exit.
The maze is given as a 2D array of size M*N.
Starting from cell 'S', moving to adjacent cells you need to find the escape cell 'E'.
Two cells are adjacent if they share a side.
Since it is a maze, there are a few challenges that you need to overcome during your escaping.

The maze contains 2 types of cells:

1. cells in which moving is permitted
   - set of all cells marked as '.' and the special cells 'S' and 'E'
2. cells in which you cannot walk in
   - set of all walls ('#') and the cells where the guards are, marked with 1, 2, 3, or 4
   Which presents the current direction of guarding (1-East, 2-South, 3-West, 4-North).

As a explanation, suppose that in the cell (i,j) there is a guard marked with X direction.
Depending from X he is guarding all the cells (i1, j1) by this rules:

- X=1 -> all (i1, j1) which are on his east side (i1 = i and j1 > j)
- X=2 -> all (i1, j1) which are on his south side (i1 > i and j1 = j)
- X=3 -> all (i1, j1) which are on his west side and (i1 = i and j1 > j)
- X=4 -> all (i1, j1) which are on his north side and (i1 > i and j1 = j)

Also if you are not permitted to be in same cell with the guard because that cell is already occupied by him.
Probably you are already supposing that if you are in one of the guarded cells than your escape is finished unsuccessfully.

Since the guards are lazy and they don't want to move around and change the cells,
It is defined that at the same time when you make a step they change their direction of guarding by this rules:

- 1 (East) -> 2 (South)
- 2 (South) -> 3 (West)
- 3 (West) -> 4 (North)
- 4 (North) -> 1 (East)

Your task is to return the minimum number of steps to reach 'E' starting from 'S'
so that no one from the guards will catch you during your escape.
If it is not possible to escape, then return -1.

Example:

Initial situation:

```
####
#S4#
#.E#
####
```

after step 1 (with X is marked your cell)

```
####
#.1#
#XE#
####
```

after step 2 - you cant go to 'E' because the guard changes his position to 2 and you will be in his guarding cells

```
####
#X2#
#.E#
####
```

after step 3

```
####
#.3#
#XE#
####
```

after step 4 (last step)

```
####
#.4#
#.X#
####
```

So the result here is 4.

Input parameters:

maze - String representation of the maze

Constraints:

- maze - String array with size N - each string will be with size M
- N - less or equal to 50
- M - less or equal to 50
- there will be only one cell marked with 'S' and one with 'E'
- the number of guards will be up to 50

Return value:

int - minimum number of steps for reaching the end

Input files:

http://ch24.org/static/docs/input3.zip

4. Ka-Ching

In the meanwhile a blind beggar is playing with the slot machines and he sacrifices his last coin in the hopes of hitting the jackpot, and fortunately for him, he succeeds. As he is blind, he cannot see how much he has won, so he records the sound with an electronic gadget that uses a software that can count the number of coins fallen based on the sound they make when they hit the ground.

Write this application for the beggar.

Again, You are provided with two example recordings where the number of coins is given to you so that you can calibrate the software, and there are ten more sound recordings where You have to guess the number of coins fallen. The closer You guess, the more points you get.
As an additional information, the slot machines only give back one type of coin, thus here there is no need to distinguish between different types of coins, unlike in the previous task.

You can find the example inputs in here:
http://ch24.org/static/docs/example1.wav
http://ch24.org/static/docs/example2.wav

The format of output is the following:
1
5

Input files:
http://ch24.org/static/docs/input4.zip

5. Crack the ZIP

You are given a multilevel passworded zip file. Crack it!

Input

The zip file itself.

Output

Though you can check the password for each level locally, to earn points upload your passwords as soon as you get them.

Example

If the password for level **01** (i.e. for 01.zip) would be *SeCrEtPaSsWoRd01*, you should upload **01.out** with one line containing the string *SeCrEtPaSsWoRd01*.

Input files:
http://ch24.org/static/docs/input5.zip