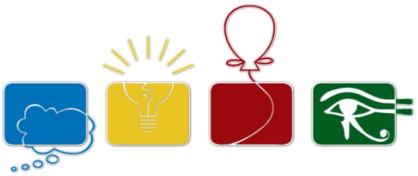

acmASCIS

FCIS Local Contest

Think.. Create.. Solve.. and Have FUN!

Faculty of Computer and Information Sciences
Ain-Shams University
September 2013

color: Pink
solution: koko.cpp
input: stdin
output: stdout

[A] Helping Koko

Once upon a time, there was a little boy named Koko. Little Koko was upset because he was very weak in maths. He asked his teacher for more materials to study from in order to improve himself. His teacher referred him to an online maths course and asked him to watch the lectures and do the homework.

The first lesson in the course was about addition, Koko watched the lecture and was so happy because he finally understood how to add numbers. He was very excited to do the homework!

Indeed, he did the homework and now he is going to upload the answers to the grading system to check them. But Oops!! The website goes down and they announced that it will be back in a week or two.

Koko became disappointed; he wanted to know the results of the homework as soon as possible. So he asked you to help him by writing a program to validate his answers.
Can you help Koko?

Input
The first line of input file denotes T (T <= 200) the number of test cases. Then T test cases follow. Each one has exactly 3 integers: A, B and C. It is guaranteed that every integer in the input file will fit into a 32-bit signed integer.

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is "True" if the addition of A and B equals to C, otherwise it's "False".

Sample input
```
5
1 2 3
3 4 8
2 9 11
1 9 123
2 5 7
```

Sample output
```
Case 1: True
Case 2: False
Case 3: True
Case 4: False
Case 5: True
```

color: Purple
solution: bags.cpp
input: stdin
output: stdout

[B] Bags

You are working in the IT department of one of the largest airlines in the world. Due to the economic crisis, your company's revenue is falling seriously and your company is trying to fight back by providing the best service to customers.

Recently the IT board has approved a new online service to be delivered to customers which helps them to figure out how long a passenger will have to wait for his entire luggage to come out on the luggage belt at the destination airport. Bags come out on the belt one by one and in a uniformly random fashion.

Given the number of bags for a passenger and the total number of bags on the aircraft, you must calculate the expected number of bags the passenger will have to wait (including his own bags) until he is able to pick up his entire luggage off the belt.

Input

The first line on input contains T (0 < T <= 100) the number of test cases, on the next T lines there are two integers N (0 <= N <= 100) and M (N <= M <= 1000) representing the number of bags for a single passenger and the total number of bags in the aircraft respectively.

Output

For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the expected number of bags the passenger has to wait until he is able to pick up all his bags off the belt. Print the result rounded to 6 decimal places.

Sample Input

```
4
1 1
1 2
2 3
100 1000
```

Sample Output

```
Case 1: 1.000000
Case 2: 1.500000
Case 3: 2.666667
Case 4: 991.089109
```

color: Orange
solution: count.cpp
input: stdin
output: stdout

[C] Count the Strings

A subsequence is a sequence that can be derived from another sequence by deleting some (possibly zero) elements without changing the order of the remaining elements.

A suffix of a string X is a string obtained by removing zero or more character(s) from the beginning of that string. Similarly, a prefix of a string X is a string obtained by removing zero or more character(s) from the end of X.

Now, If you want to append 2 strings(A, B), before appending, you have an optional choice to remove any non-empty prefix of B that is a suffix of A at the same time.

For example appending "abbab" and "babaab" has 3 valid results:

- "abbabbabaab" (without using the above optional step)
- "abbababaab" (by removing "b" from "babaab")
- "abbabaab" (by removing "bab" from "babaab")

In this problem, You are given a list of N strings, You have to count the number of distinct strings of length exactly M that can be constructed by selecting a subsequence of strings from the given list and appending this subsequence in order.

Input

The first line of the input file denotes T (T <= 100) the number of test cases. The first line for each test case will contain two space-separated integers which is N (1 <= N <= 10^4) and M (1 <= M <= 8) then N lines representing the N-list strings will follow with each string in one line. Each string in the given list will have length that is greater than 0 and less than or equal M and each character in it will be either 'a' or 'b'.

Output

For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the required number.

Sample Input

```
2
4 7
abaa
aabba
bbbb
bbbaab
```

```
9 4
a
b
ba
bb
aa
ba
aaa
bbbb
aaaa
```

Sample Output

```
Case 1: 2
Case 2: 11
```

[D] Transformation Matrix

color: Black
solution: matrix.cpp
input: stdin
output: stdout

You are given two matrices **A** and **B**. Each matrix contains only '0' and '1' digits. The j-th character of the i-th element is the value at cell (i, j). Your goal is to transform matrix **A** into matrix **B** using a series of swaps. On each swap, you choose two adjacent (horizontally, vertically or diagonally) cells in matrix **A** and swap their values.

There is a limit to the number of times each cell in matrix **A** can be used. You are given a third matrix **C**. Cell (i, j) in matrix **A** can be used in a maximum of **C**(i, j) swaps. You have to calculate the fewest number of swaps required to achieve your goal, or print -1 if it is impossible.

Input
The first line of input file denotes T (T <= 200) the number of test cases. Then T test cases follow, the first line of each test case contains two separated integers R, C (1 <= R, C <= 20) specifying the number of rows and the number of columns - respectively - of the three matrices (A, B and C) then there is 3*R lines each have C space separated integers specifying the three matrices, the 1st R lines describe A, 2nd R lines describe B and the 3rd R lines describe C, each element of A and B will be either 0 or 1 and each element of C will be between 0 and 9.

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1. and Y is the the fewest number of swaps required to transform A into B or -1 if it is impossible.

Sample Input
```
2
3 3
1 1 0
0 0 0
0 0 1
0 0 0
1 1 0
1 0 0
2 2 2
2 2 2
2 2 2
2 3
0 0 1
1 1 0
0 0 0
1 1 1
0 0 0
1 1 1
```

Sample Output
```
Case 1: 4
Case 2: -1
```

color: Red
solution: convert.cpp
input: stdin
output: stdout

[E] Convert It!

Given a string A, you have to convert it to the word "ACMASCIS" with the minimum cost using 2 types of operations:

1. Pick an arbitrary character from A and convert it to any English letter (with cost = 1)
2. Swap any two characters in A (with cost = 0)

Input

The first line of the input file denotes T (T <= 100) the number of test cases. Then T lines representing each test case follow. In each line the string A will be given. A will contain 8 uppercase English letters.

Output

For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the minimum cost to convert the given string to the word "ACMASCIS"

Sample input

```
2
AACCDDNS
ACMASCIS
```

Sample output

```
Case 1: 3
Case 2: 0
```

[F] Divide, Assign, and Play!

color: White
solution: scoreboards.cpp
input: stdin
output: stdout

There is a bowling lounge that has m different frames (i.e. can have at most m bowling games running at the same time), each of these frames has a scoreboard which displays the unique nicknames and the scores of players currently playing on it. Before the game starts, the nicknames of the players (assigned to this frame) will be displayed in random order with initial scores = 0.

Now, there are n groups of friends who want to play bowling, the i-th group consists of a[i] players. Each group can be divided into any number of separated teams such that each team has >= 1 player.

The manager of the lounge wants to assign these teams to the m games such that:
- Each team must be assigned to exactly one frame
- Each frame must have exactly one team playing on it

The manager is wondering, in how many ways can these groups be divided, assigned and displayed on the scoreboards before starting to play?

Input
The first line of input file denotes T (T <= 100) the number of test cases. Then T test cases follow. Each test case will be presented in 2 lines, the first one contains n, m (1 <= n, m <= 100) the second line contains n integers representing the initial number of players of each group, each a[i] will be greater than zero and not greater than one hundred.

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the number of ways as described above. Since this number can be really huge, you should print it modulo 1000000007. If there is no any valid way, print "Invalid" instead (without the quotes).

Sample Input
```
3
2 3
2 1
2 2
1 1
4 6
1 2 3 4
```

Sample Output
```
Case 1: 6
Case 2: 2
Case 3: 708480
```

color: Silver
solution: perfect.cpp
input: stdin
output: stdout

[G] Perfect Sub-Arrays

A perfect square is a number which is equal to the square of another number. The first few perfect squares are 0, 1, 4, 9, 16, 25, 36.
Similarly there are perfect cubes, perfect fourths, perfect fifths and so on.
Generally, a positive integer n is a perfect power if there exist integers m, k such that k > 0 and n = m^k. In particular, n is said to be a perfect kth power. For example 64 = 64^1 = 8^2 = 4^3 = 2^6, so 64 is a perfect 1st, 2nd, 3rd and 6th power.

In this problem, you will be given an array of length n, and m queries, each is one of the following types:
- "0 i v", meaning replace the ith -one based- value in the array with new value v.
- "1 i j k" meaning print "Yes" if all numbers in the sub-array [i...j] are perfect kth powers and "No" otherwise.

Input
The first line of input file denotes T (T <= 100) the number of test cases. Then T test cases follow. The first line of each test case will contain n(1 <= n <= 100000), m (1 <= m <= 2000). The next line will contain n values representing the array. Next m lines contain one of the above said queries such that (1<=i<=j<=n) and (1 <= k <= 10).
It is guaranteed that every integer in the input will be a positive integer and it will fit into a 32-bit signed integer.

Output
For every case in the input, print one line "Case X:", without the quotes, where X is the case number starting from 1. Then for each query of the 2nd type -in the Xth test case- print "Yes" or "No".

Note: Brute force methods examining every number in the given range will likely exceed the allotted time limit.

Sample Input

```
2
2 5
5 25
1 1 2 2
1 2 2 2
0 1 36
1 1 2 2
1 1 2 3
5 6
64 128 256 512 1024
1 1 5 2
0 2 81
1 1 2 2
1 1 3 2
0 3 10000
1 1 3 2
```

Sample Output

```
Case #1:
No
Yes
Yes
No
Case #2:
No
Yes
Yes
Yes
```

color: Yellow
solution: lcas.cpp
input: stdin
output: stdout

[H] LCAS

A subsequence is a sequence that can be derived from another sequence by deleting some (possibly zero) elements without changing the order of the remaining elements.
For example if A = {1, 5, 3}, then {}, {1}, {5}, {3}, {1, 3}, {1, 5}, {5, 3} and {1, 5, 3} are valid subsequences of A, while {3, 5}, {5, 1}, {3, 1} and {3, 5, 1} are not.

An alternating sequence is a sequence of integers such that any two adjacent integers cannot be both even or both odd. For example: {1}, {2}, {1, 2, 7} and {2, 5, 10} are alternating sequences, while {2, 4}, {1, 3}, {1, 2, 6} and {2, 4, 7, 9} are not.

In this problem, you will be given 2 sequences of integers A and B, Find the length of the longest common alternating subsequence (LCAS) between A and B. In other words, you will have to find the length of longest alternating sequence C such that C is a subsequence of both A and B.

Input
The first line of input file denotes T (T <= 100) the number of test cases. Then T test cases follow. Each test case will be given in 3 lines, the first one have exactly 2 integers N and M (1 <= N, M <= 1000). The second line have N integers representing the sequence A, The third line have M integers representing the sequence B. It is guaranteed that every integer in the input will fit into a 32-bit signed integer.

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1 and Y is the length of the LCAS for the corresponding test case.

Sample Input
```
2
2 3
1 2
1 5 2
5 5
1 2 3 4 5
1 2 3 5 4
```

Sample Output
```
Case 1: 2
Case 2: 4
```

color: Green
solution: mario.cpp
input: stdin
output: stdout

[I] Saving the Princess!

As you may know - from your old childhood days - Super Mario is an atari platform game in which its main character "Super Mario" has one goal, to save the princess! Despite being called "Super" Mario, this time he needs your help.

Super Mario is moving on NxM grid. Other than the two cells containing Mario and the princess, each cell in the grid may be either free or has a monster. Some of the free cells may contain doors or keys, there are ten different types of doors and consequently ten different types of keys.
Each key can open only doors of the same type. Mario can move between adjacent free cells vertically or horizontally. Mario will die if he moved to a cell having a monster. If a cell contains a door, Mario can go there only if he can open that door which requires holding the appropriate key. The grid can contain more doors and/or keys of the same type. each key cell has an infinite supply of a particular key type. If Mario stepped through a key cell, he must take exactly one key before leaving it, but unfortunately if he was holding the same type of that key before stepping through this cell, the two keys will disappear. Once super Mario leaved a door cell this door will be closed again.

Super Mario needs to save the princess by arriving to her cell, can you help super Mario by determining either the minimum number of moves to save her if he can or stating that he can't save her.

Input
The first line of input file denotes T (T <= 100) the number of test cases. Then T test cases follows describing T grids Each grid begins with a line containing two integer numbers R and C (1 <= R, C <= 50) specifying the grid size. Then there are R lines each containing C characters. Each character is one of the following:

- '.' indicating a free cell
- '#' indicating a monster
- '*' indicating super Mario
- '$' indicating the princess
- 'A'-'J' indicating a door cell (corresponding to the 10 door types with 'A' as the first type and 'B' as second type and so on)
- 'a'-'j' indicating a key cell (corresponding to the 10 key types with 'a' is the key type that can open a door with type 'A', 'b' is the key type that can open a door with type 'B' and so on).

Note that:

- There will be only one '*' and one '$' in gird.
- It is allowed to have keys without corresponding doors and vice versa

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1. and Y is the minimum number of moves Mario needs to save the princess or -1 if he can't save her.

Sample Input

```
2
2 9
$..A....*
.B.a.....
10 10
a........F
..........
..........
..........
####*####g
....A.....
b....i....
..........
B........B
...b....#$
```

Sample Output

```
Case 1: 10
Case 2: 26
```

color: Blue
solution: gardens.cpp
input: stdin
output: stdout

[J] Gardens

Two neighbors decided to build circular-shaped gardens around their houses. They asked you to help them in finding out whether the two gardens will intersect or not. If there will be an intersection, they needed to know the distance between the intersection points.
Note that one garden can't be completely contained in the other garden.

Input
The first line on input contains T (0 < T <= 100) the number of test cases, Each of the next T lines consists of 6 floating point numbers, the first 3 numbers x0, y0, r0 represent center coordinates and radius of the first garden respectively, and the second 3 numbers x1, y1, r1 represent center coordinates and radius of the second garden.

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the distance between the intersection points.
if there is no intersection, print "No intersection." instead.

Sample Input
2
0.0 0.0 2.5 2.0 0.0 3.0
-6.0 -7.0 1.0 8.0 2.0 1.0

Sample Output
Case 1: 4.96078
Case 2: No intersection.

[K] Power Sum

color: Gold
solution: power.cpp
input: stdin
output: stdout

You are given 2 integers n and k. Print the value of the sum 1^k + 2^k + 3^k + … + n^k modulo 1000000007.

Input
The first line of input file denotes T (T <= 100) the number of test cases. Then T test cases follow. Each one contains exactly 2 integers: n (1 <= n <= 10^9) and k (1 <= k <= 50).

Output
For every case in the input, print one line "Case X: Y", without the quotes, where X is the case number starting from 1, and Y is the required sum.

Sample input

```
4
5 1
4 2
13 5
123456789 1
```

Sample output

```
Case 1: 15
Case 2: 30
Case 3: 1002001
Case 4: 383478132
```