CUHK CSE Programming Contest 2013

15:00 – 18:00, 30th May 2013

<table>
<thead>
<tr>
<th>Problem</th>
<th>Execution Time Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>A – Big A</td>
<td>1 second</td>
</tr>
<tr>
<td>B – ASCII Art</td>
<td>1 second</td>
</tr>
<tr>
<td>C – Race</td>
<td>2 seconds</td>
</tr>
<tr>
<td>D – Horse Racing</td>
<td>2 seconds</td>
</tr>
<tr>
<td>E – Examination Hall</td>
<td>2 seconds</td>
</tr>
<tr>
<td>F – Up and Down Numbers</td>
<td>2 seconds</td>
</tr>
<tr>
<td>G – Archery Competition</td>
<td>2 seconds</td>
</tr>
<tr>
<td>H – Archery Competition (hard)</td>
<td>3 seconds</td>
</tr>
</tbody>
</table>

This problem set should contain eight (8) problems on numbered pages. Inform anyone of the contest judges or invigilators immediately if you found anything missing from your problem set.

Do not touch the paper until you are told to do so.
Problem A. Big A

Input file: stdin
Output file: stdout
Time limit: 1 second

Bob decides to send an e-card to Alice to wish her a grade A in the examination. He decides to draw a big ‘A’ but he does not know how large the ‘A’ should be.

Bob wants your help to write a program to create several big ‘A’s in different sizes in the format specified below.

Input
The first line contains a single integer $T$, the number of test cases.
Each test case contains a single integer $n (2 \leq n \leq 10)$, the size of the big ‘A’.

Output
For each test case, output the big ‘A’ according to the size $n$.
For the big ‘A’ of size $n$, the height of the big ‘A’ is $2n$ and the horizontal line of the big ‘A’ is located at line $n + 1$.

For example, when $n = 2$, the output will be as follows:
(one dot ‘.’ represents one space character)

```
...*
...*. *
.*****
*.....*
```

Sample input and output

<table>
<thead>
<tr>
<th></th>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Problem B. ASCII Art

Input file: stdin
Output file: stdout
Time limit: 1 seconds

Bob continues to design his e-card to Alice.

He decides to decorate the card using ‘ASCII Art’. After he has finished some of his newly designed pattern, he thinks that some patterns are in wrong orientations.

Bob wants your help. Please write a program to fix the pattern designed by him. To be specific, he wants to rotate the pattern $90k^\circ$ clockwise, or to reflect the pattern vertically (along $x$-axis) or horizontally (along $y$-axis).

Input
The first line contains a single integer $T$ ($1 \leq T \leq 20$), the number of test cases.
In each test case, the first line contains three integers $h$, $w$ and $n$ ($1 \leq h, w \leq 100$ and $0 \leq n \leq 10$), the height and width of the input pattern, and also the number of transformations.
The next $h$ lines are the input pattern, then the following $n$ lines are the transformation commands.

For rotation commands, the input line will be ‘rotate $d$’, where $d$ is a multiple of $90$.
For reflection commands, the input line will either be ‘reflect x’ or ‘reflect y’.

The pattern will consist of non-space printable characters.

Output
For each test case, the first line contains the case number in the form of ‘Case n:’, then the following lines are the final pattern after all changes.

Sample input and output

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
</table>
| 2
5 4 1
....
.oo
0ooo
.o..
....
rotate 90
5 9 3
....+++
..+++++
...+++++
...+++++
...+++++
...+++++
..+++
reflect x
reflect x
reflect x | Case 1:
..o...
..o...
..o...
Case 2:
..++..+
..++++++
..++++++
..++++++
..++++++
..++++++
..++++++
..++++++
..++++++
..+++++
..++++++
..+++++
..++++++
..+++++
..++++++
..+++++
..++++++
..+++++ |
Problem C. Race

Input file: stdin
Output file: stdout
Time limit: 2 seconds

Bob loves car racing very much, he always keeps track of the latest news of each race.

Today there will be a lot of races being broadcasted. But unfortunately Bob’s TV is out of order! In order to know what happens in each race, he accesses to the most popular forum - Silveren Forum, to figure out what’s happening during the race from the related posts.

During the race, the rankings of cars change very often. When a car surpasses some cars, the users in the forum would leave a comment like this:

"Wow! Car 14 overtakes car 27!"

By gathering all comments in the post, Bob wants to know the final rankings of all the cars in each race. He asks you to help him to analyze the comments.

Input
The first line of the input contains a single integer \( T \) (\( 1 \leq T \leq 20 \)), the number of races today.
In each race, the first line contains two integers \( N \) (\( 1 \leq N \leq 100000 \)) and \( M \) (\( 1 \leq M \leq 200000 \)), the number of cars in that race and the number of comments in the post. The cars are numbered from 1 to \( N \).

The next \( M \) lines are the \( M \) comments describing the race in chronological order (i.e. sorted by time).
Each comment contains two integers \( X \) and \( Y \) (\( 1 \leq X, Y \leq N \)), which states that car \( X \) overtakes car \( Y \). When we say car \( X \) overtakes car \( Y \), it means car \( X \) keeps overtaking other cars until it overtakes car \( Y \). The input guarantees at that moment, car \( Y \) must have a higher ranking than car \( X \).

You may assume that at the beginning of each race, car \( i \) is positioned in front of car \((i + 1)\) for \((1 \leq i \leq N - 1)\).

Output
For each race, output \( N \) lines listing the cars from the highest ranking to the lowest ranking.

Sample input and output

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3 1</td>
<td>1</td>
</tr>
<tr>
<td>3 1</td>
<td>2</td>
</tr>
<tr>
<td>5 3</td>
<td>1</td>
</tr>
<tr>
<td>5 3</td>
<td>4</td>
</tr>
<tr>
<td>4 2</td>
<td>5</td>
</tr>
<tr>
<td>5 2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>
Problem D. Horse Racing

Input file: stdin
Output file: stdout
Time limit: 2 seconds

Once upon a time, in a galaxy far, far away, Chiu and Real were friends. Unfortunately, they argued with each other for an unimportant thing. Since they both owned some horses, they decided to choose the winner by competing in $n$ horse races.

To simplify this problem, each horse has a racing power. In each race, the horse with higher power will win, and the race is a draw if both horses have the same racing power. Each horse will only compete in exactly one race.

To maximize the chance of winning, Real sent out his spy and knew that Chiu would use the horse with power $A[i]$ in the $i$-th race. Real had $n$ horses and the $i$-th horse had racing power $B[i]$. Now he wants you to tell him the maximum number of races that he can win if he uses the horses wisely.

**Input**
The first line contains a single integer $T$ ($1 \leq T \leq 20$), the number of test cases.
In each test case, there will be three lines.
The first line contains an integer $n$ ($1 \leq n \leq 3000$), the number of races.
The second line contains $n$ integers, $A[i]$ ($1 \leq A[i] \leq 10^9$), the power of Chiu’s horse in the $i$-th race.
The last line contains $n$ integers, $B[i]$ ($1 \leq B[i] \leq 10^9$), the powers of Real’s horses.

**Output**
For each test case, output a line containing the maximum number of races that Real can win.

**Sample input and output**

```
3
4
1 3 4 2
8 5 7 6
5
3 6 4 5 8
4 2 6 4 3
4
99 999 9999 99999
11 111 1111 11111
```

```
4
2
3
```
**Problem E. Examination Hall**

**Input file:** stdin  
**Output file:** stdout  
**Time limit:** 2 seconds

The examination week is coming, and everyone starts to panic, including Peter, the one who is responsible for figuring out the seating plan for all grade 1 students.

Since the students come from grade 1, they will be sitting at a 1-dimensional examination hall, i.e. there are a total of $n$ chairs arranged in a straight line. Furthermore, Peter knows the students in grade one is very good at cheating, and two students can exchange their answers without being noticed if there are no more than $k$ chairs separating them. Of course, cheating is undesirable for Peter, so some of the seats must be left empty. Besides, Peter doesn’t want the invigilator to feel lonely, so there must be at least one seat left for students. Peter wants to know how many possible nonempty seating plans are there for a given $n$ and $k$. Can you help him?

Two seating plan is the same if they reserve the same set of seats for students. For example, if there’s a total of 4 chairs, $\{1, 2, 3, 4\}$, with $k = 1$, a possible seating plan is $\{1, 3\}$. And there are exactly 7 seating plans possible, namely $\{1\}, \{2\}, \{3\}, \{4\}, \{1, 3\}, \{1, 4\}, \{2, 4\}$.

**Input**

The first line contains a single integer $T$, the number of test cases.

Each test case consists of one line with two positive integers, $n$ and $k$. $1 \leq n, k \leq 100000$.

**Output**

For each test case, output a single integer, the total number of nonempty seating plans for the given $n$ and $k$, in one line. Since the number can be very large, you should give the answer mod 1000000007.

**Sample input and output**

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>4 1</td>
<td>5</td>
</tr>
<tr>
<td>4 2</td>
<td>6</td>
</tr>
<tr>
<td>6 8</td>
<td>52</td>
</tr>
<tr>
<td>15 6</td>
<td>71317746</td>
</tr>
<tr>
<td>1000 165</td>
<td></td>
</tr>
</tbody>
</table>
Problem F. Up and Down Numbers

Input file: stdin
Output file: stdout
Time limit: 2 second

Dr. Jones is interested in numbers. He especially likes very large numbers.

Today he is investigating a kind of numbers called “Up and Down numbers”. This kind of numbers are composed of digits in the form \( d_1d_2d_3\cdots d_n \), such that \( d_1 < d_2, d_2 > d_3, d_3 < d_4 \) etc. So it goes like “up” and then “down” and then “up” and so on.

This time, he wants to count the total number of “Up and Down numbers” with fixed length \( N \) and ending digit \( K \). The “Up and Down numbers” can have leading zeros.

For example, if \( N \) is 3 and \( K \) is 8, then the valid numbers are 098, 198, 298, \ldots, 898. There is no “Up and Down numbers” ending with 0 with length 2.

Input
The first line contains a single integer \( T \) \((1 \leq T \leq 100000)\), the number of test cases. Each test case consists of a single line with two integers \( N \) \((1 \leq N \leq 100000)\) and \( K \) \((0 \leq K \leq 9)\).

Output
For each test case, output the answer module 1000000007 \((10^9 + 7)\).

Sample input and output

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>3 8</td>
<td>0</td>
</tr>
<tr>
<td>2 0</td>
<td>4</td>
</tr>
<tr>
<td>2 4</td>
<td>205</td>
</tr>
<tr>
<td>4 5</td>
<td></td>
</tr>
</tbody>
</table>
Problem G. Archery Competition

Input file: stdin
Output file: stdout
Time limit: 2 seconds

Polary and Peter are having an archery competition. There are n targets in the field. Each target consists of 5 rings and there is a score on each ring. If the arrow hits on a ring with score x, then the shooter will get x scores. Polary and Peter are good archers, when they aim at a specific target, they always hit on one of the rings in the target. In particular, considering the rings from inner-most to outer-most, their arrows will hit them with probability 0.2, 0.3, 0.3, 0.1, 0.1 respectively.

Today, they are going to decide the winner with only one shot each. First, Peter will choose a target and shoots his arrow. After the arrow hits the ring, Peter's score will be recorded. Then it is Polary’s turn to choose a target and shoot. Notice that they can choose the same target. The one with higher score is the winner. If they have the same score, then the winner will be decided by flipping a fair coin.

Both of them will choose the target optimally to maximize the chance they win. Calculate the probability that Polary wins the competition.

Input
The first line contains a single integer T, the number of test cases. In each test case, the first line contains an integer n (1 ≤ n ≤ 100), the number of targets. Each of the next n lines contains 5 integers in the range 1 to 10^6, describing the scores on each ring in the targets. They are listed from the inner-most ring to the outer-most ring.

Output
For each test case, output the probability that Polary wins, rounded to exactly 3 decimal places.

Sample input and output

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.500</td>
</tr>
<tr>
<td>1</td>
<td>0.500</td>
</tr>
<tr>
<td>5 3 2 4 1</td>
<td>0.500</td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4 2 3 5 1</td>
<td></td>
</tr>
<tr>
<td>2 2 2 1 1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5 4 2 3 1</td>
<td></td>
</tr>
<tr>
<td>5 5 2 3 3</td>
<td></td>
</tr>
<tr>
<td>6 2 2 4 3</td>
<td></td>
</tr>
</tbody>
</table>
Problem H. Archery Competition (Hard)

Input file: stdin
Output file: stdout
Time limit: 3 seconds

Polary and Peter are having an archery competition. There are \( n \) targets in the field. Each target consists of 5 rings and there is a score on each ring. If the arrow hits on a ring with score \( x \), then the shooter will get \( x \) scores. Polary and Peter are good archers, when they aim at a specific target, they always hit on one of the rings in the target. In particular, considering the rings from inner-most to outer-most, their arrows will hit them with probability 0.2, 0.3, 0.3, 0.1, 0.1 respectively.

Today, they are going to decide the winner with only one shot each. First, Peter will choose a target and shoots his arrow. After the arrow hits the ring, Peter’s score will be recorded. Then it is Polary’s turn to choose a target and shoot. Notice that they can choose the same target. The one with higher score is the winner. If they have the same score, then the winner will be decided by flipping a fair coin.

Both of them will choose the target optimally to maximize the chance they win. Calculate the probability that Polary wins the competition.

Input
The first line contains a single integer \( T \), the number of test cases. In each test case, the first line contains an integer \( n \) (\( 1 \leq n \leq 100,000 \)), the number of targets.
Each of the next \( n \) lines contains 5 integers in the range 1 to 10\(^6\), describing the scores on each ring in the targets. They are listed from the inner-most ring to the outer-most ring.

Output
For each test case, output the probability that Polary wins, rounded to exactly 3 decimal places.

Sample input and output

<table>
<thead>
<tr>
<th>stdin</th>
<th>stdout</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.500</td>
</tr>
<tr>
<td>1</td>
<td>0.500</td>
</tr>
<tr>
<td>5 3 2 4 1</td>
<td>0.500</td>
</tr>
<tr>
<td>2</td>
<td>0.500</td>
</tr>
<tr>
<td>4 2 3 5 1</td>
<td>0.500</td>
</tr>
<tr>
<td>2 2 2 1 1</td>
<td>0.500</td>
</tr>
<tr>
<td>3</td>
<td>0.500</td>
</tr>
<tr>
<td>5 4 2 3 1</td>
<td>0.500</td>
</tr>
<tr>
<td>5 5 2 3 3</td>
<td>0.500</td>
</tr>
<tr>
<td>6 2 2 4 3</td>
<td>0.500</td>
</tr>
</tbody>
</table>