ECOO 2019

Programming Contest Questions

Provincial Competition (Round 1)

March 20-27, 2019

Problem 1: Free Shirts

Throughout the year, there are many programming events that students can attend to meet like-minded individuals, hone their skills and, most importantly, get a free t-shirt.

Ian is an avid attender of these events because he hates doing laundry. Ian only does his laundry when all his shirts are dirty, so this constant influx of shirts allows him to put off laundry for longer periods of time.

Ian starts with **N** clean shirts. Ian wears one clean shirt every day, after which it becomes dirty. If at the beginning of a day (before any events) Ian only has dirty shirts, then he will do the laundry, which makes all his shirts clean again. If Ian goes to an event, then he will receive one clean shirt.

Given the initial number of shirts that Ian has and the schedule of events for the next **D** days, how many times will Ian do the laundry in the next **D** days?

Input Specifications

DATA11.txt (DATA12.txt for the second try) will contain 10 datasets. Each dataset begins with three integers **N**, **M**, **D** ($1 \le \mathbf{N}, \mathbf{M} \le 100$, $1 \le \mathbf{D} \le 1{,}000$), the initial number of shirts that Ian has, the number of events coming up, and the number of days, respectively.

The next line contains **M** integers $\mathbf{A_i}$ ($1 \le \mathbf{A_i} \le \mathbf{D}$), the days on which there are events. There may be multiple events in a single day.

Output Specifications

For each dataset, output the number of times that Ian will do the laundry in the next **D** days.

Sample Input (Two Datasets Shown)

```
1 1 10
10
1 3 10
2 9 5
```

Sample Output

```
9
3
```

Explanation of Sample Datasets

In the first test, Ian does the laundry on days 2, 3, 4, 5, 6, 7, 8, 9, 10.
In the second test, Ian does the laundry on days 2, 4, 7.

Problem 2: L-Systems Go

A Lindenmayer system (L-system) is a parallel rewriting system and a type of formal grammar. Some uses of L-systems include creating visual representations of vegetation, flowers, trees, and grasses.

An L-system consists of an axiom and a set of rules.

- The axiom (**A**) is a string that represents the starting state of the L-system
- The set of rules defines what happens to each letter in an iteration of the system.

Given the axiom, set of rules, and the number of iterations for which to run the system, you are tasked with generating a two-letter code followed by the length of the final state. The two-letter code will be the concatenation of the first letter and last letter of the final state.

Input Specifications

DATA21.txt (DATA22.txt for the second try) will contain 10 datasets. Each dataset begins with three terms **R**, **T, A** ($1 \le \mathbf{R} \le 26$, $1 \le \mathbf{T} \le 30$, $1 \le |\mathbf{A}| \le 5$), the number of rules, the number of iterations, and the axiom. The next **R** lines each contain a character **C** followed by a string **S** ($1 \le |\mathbf{S}| \le 3$) which represents a rule stating that each occurrence of **C** should be replaced with the string **S**. It is guaranteed that each letter in the system will have a corresponding rule.

For the first 4 cases, $\mathbf{T} \le 5$.
For the first 7 cases, $\mathbf{T} \le 12$.

Output Specifications

For each dataset, output a concatenation of first and last letter of the state and the length of the state after **T** iterations.

Sample Input (Two Datasets Shown)

```
3 5 AC
A CAB
B CB
C ACB
4 5 AD
A AC
B ACA
C BD
D B
```

Sample Output

```
CB 288
AB 60
```

Explanation of Sample Datasets

In the first dataset, the first iteration is AC > CABACB.
In the second dataset, the first two iterations are AD > ACB > ACBDACA.

Problem 3: Side Scrolling Simulator

Marr E.O. is currently working on his next video game. He is creating a side-scrolling game and wants you to ensure that the levels that he is creating are possible to complete.

The character in the game, Loo E.G., can only jump a certain height (**J**). Throughout the game there are walls with openings created in them for Loo E.G. to jump through. Loo E.G. can go through these openings as follows: he first jumps vertically up to **J** spaces, then he moves two spaces horizontally through the opening, and finally falls vertically on the other side of the wall.

Marr E.O. thinks he created the walls such that it is possible for Loo E.G. to complete the level, but he wants you to check. Marr E.O. provides his levels in a text format, where each symbol has a meaning:

- A dot '.' means there is nothing there, and Loo E.G. can move freely through that space.
- A hash '#' represents the ground (last row of the input).
- An at-symbol '@' represents the wall, which cannot be passed through or stood upon.
- The letters 'L' and 'G' are used to represent where Loo E.G. start and end respectively. The start point is guaranteed to be left of the end point. Both points are guaranteed one space above the ground.

Input Specifications

DATA31.txt (DATA32.txt for the second try) will contain 10 datasets. Each dataset begins with a line containing three integers **J**, **W**, **H** ($1 \le J \le 10$, $5 \le W \le 100$, $2 \le H \le 10$), representing the jump height of Loo E.G. and the width and height of the level. **H** lines follow, each containing **W** ASCII characters representing the level (as shown above).

Output Specifications

For each dataset, output 'CLEAR' if Loo E.G. is able to complete the level, or a single integer **N** that represents the first column that Loo E.G. is unable to reach.

Sample Input (Two Datasets Shown)

```
2 10 5
..........
.......@..
.......@..
L......@.G
##########
1 10 5
..........
..@..@@...
..@.......
L.....@..G
##########
```

Sample Output

```
8
CLEAR
```

ECOO 2018 Question Development Team

John Ketelaars ..... ECOO-CS Communications
Stella Lau ..... University of Cambridge
Davin Beharry ..... Ryerson University
Andrew Seidel ..... John Fraser Secondary School, Mississauga
David Stermole ..... ECOO-CS President
Reyno Tilikaynen ..... University of Waterloo
Andy Huang ..... University of Waterloo

Problem 4: Bus Route

Every morning, Lisa drives a public bus from one terminal stop to the other. Through months of careful observation, she has noticed that the order in which buses complete the route isn't necessarily the order in which they started. It is possible for the first bus to arrive at the first stop to not be the first bus to leave the last stop due to delays along the route. To leave work as early as possible, Lisa would like to figure out which bus will finish the route first on a given morning.

The route has **N** bus stops (labelled 1 to **N**) and **M** daily passengers. Each passenger gets on a bus at some stop and gets off at a later stop. Each morning, the buses (labelled 1, 2, 3, …) start at the first stop in 10-minute intervals, i.e., the **k**+1st bus starts 10 minutes after the **k**th bus.

A bus takes one minute to travel from one stop to the next. Each bus can carry 40 people. Passengers at a bus stop board the buses in the order that they arrive at the stop. A bus will only stop at a bus stop if it is not full and there is a passenger waiting at the stop, or if someone on the bus wants to get off at the stop. If a bus stops, it first lets off any passengers that want to get off the bus, then picks up as many passengers that it can. Stopping delays a bus by one minute, regardless of how many people it picks up or drops off. If multiple buses arrive at a stop at the same time, they would stop in increasing order of their indices (i.e., the bus with the lowest index stops first).

Can you help Lisa by figuring out which bus she should drive to complete the route first?

Input Specifications

DATA41.txt (DATA42.txt for the second try) will contain 10 datasets. Each dataset begins with two integers **N**, **M** ($1 \le N, M \le 500{,}000$). The next **M** lines each contain two integers **S**, **F** ($1 \le S < F \le N$), which represent a passenger who wants to travel between stop **S** and stop **F**. If multiple people start at the same stop, buses pick them up in the order that they are listed.

For the first 4 datasets, **N**, **M** ≤ 1,000.

Output Specifications

For each dataset, output "Bus #X", where X is the index of the bus that will be the first to reach the last stop. If two buses tie for first, output the one with the lower index.

Sample Input (Two Datasets Shown)

```
5 3
1 2
2 3
4 5
13 6
1 7
2 8
3 9
4 10
5 11
6 12
```

Sample Output

```
Bus #1
Bus #2
```