Editor's Choice Fall '21

November 13, 2021
No Time Limit
8 Questions
Problem Packet by CALICO

Table of Contents

Problem 1: Not So Self-Driving
5 Points

Problem ID: `car`

Rank: 1

Introduction

Dwayne and Johnson are your partners in a group project for your autonomous-vehicles engineering course. However, none of you were paying attention in class, so you forget about the project until the night before it's due! Knowing there would be no time to train a decent self-driving algorithm in just a few hours, you, Dwayne, and Johnson all decide to fake a self-driving car with a few sensors and a bunch of if-statements.

Your task is to create an obstacle-avoidance system that will output instructions based on information about obstacle distance and the car's speed.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A single line containing two numbers separated by a colon. The numbers will be as follows:
  - A non-negative decimal value `v`, denoting the vehicle's current speed (in m/s).
  - A positive decimal value `x`, denoting the obstacle's distance from the front of the car (in m).
  - Both values will be rounded to two decimal places.

Example Input:

```
5
23.15:98.34
2.40:17.33
6.79:5.01
0.00:1.53
113.56:113.56
```

Program Output

For each test case, your program should output instructions to the car based on the following criteria:

- If the car is projected to collide with the obstacle in one second or less at the current speed, your program should output `SWERVE`
- Otherwise, if the car is projected to collide with the obstacle in five seconds or less at the current speed, your program should output `BRAKE`
- Otherwise, your program should output `SAFE`

Example Output:

```
BRAKE
SAFE
SWERVE
SAFE
SWERVE
```

Problem Constraints

`T` ≤ 1000

0 ≤ `v` ≤ 200

1 ≤ `x` ≤ 400

Assume the obstacle is directly in front of the car.

Problem Author: Chris Liu

Problem 2: All About That Base
7 Points

Problem ID: `dna`

Rank: 1

Introduction

You are working as an unpaid lab intern in the genetic engineering division of Aperture Laboratories. Due to your diligent work ethic, you have completed all of your assigned work before your lunch break! However, your evil boss decides to assign you busy work, forcing you to manually analyze DNA strands until your shift is over! DNA is composed of two polynucleotide chains, each of which contains nitrogenous bases of type A, T, C, or G. Each base has its respective pair, with A pairing with T and C pairing with G. You don't want to spend the rest of the day staring at DNA strands, so you need to come up with a solution quick!

Your task is to create a DNA conversion program that will convert each base in a polynucleotide chain to its respective base pair.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of number of test cases that follow. Each test case will have the following input:

- A single line containing a string of DNA bases of length `n`, each of type A, T, C, or G

Example Input:

```
5
AAAATA
CGATAGAAGCTA
C
ATCGTAGCTGGGGAT
GGGGGGGGG
```

Program Output

For each test case, your program should output the respective polynucleotide chain that pairs with the input chain.

- Your output should correspond to the following pairs: A to T and C to G.

Example Output:

```
TTTTAT
GCTATCTTCGAT
G
TAGCATCGACCCCTA
CCCCCCCCC
```

Problem Constraints

`T` ≤ 1000

1 ≤ `n` ≤ 500

The input chain will be entirely capitalized.

Problem Author: Chris Liu

Problem 3: The Case of the Missing Book
11 Points

Problem ID: `library`

Rank: 2

Introduction

You're living a glamorous life as your school library's only T.A. One day, a freshman English class comes in to return some books from their Shakespeare unit—but some of their returns are all mixed up! Students are turning in others' books—and to make matters worse, your school has yet to electronically track their inventory, so you have to keep track of all these returns by hand. After all the students have left the library, you're notified that one student was absent from class—meaning one book is still yet to be returned. Can you find out whose book is missing from today's records alone?

Your task is to create a program that will output the owner of the absent student's book.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A first line consisting of two values separated by a space. The values are as follows:
  - The name of the absent student.
  - A non-negative integer `n` denoting the number of returns that happened that day.
- The next `n` lines will consist of a single transaction record in the following format:

```
<NAME> TURNS IN <OWNER>'s BOOK
```

- A blank line separating individual test cases.

Example Input:

```
3
David 1
Alexa TURNS IN David's BOOK

Javid 5
Preston TURNS IN Walter's BOOK
Jon TURNS IN Quincy's BOOK
Walter TURNS IN Javid's BOOK
Quincy TURNS IN Jon's BOOK
Gary TURNS IN Gary's BOOK

Susan 2
Abcde TURNS IN Felicia's BOOK
Felicia TURNS IN Abcde's BOOK
```

Program Output

For each test case, your program should output the owner of the absent student's book in the following format: `<ABSENT STUDENT> HAS <OWNER>'s BOOK`

Example Output:

```
David HAS Alexa's BOOK
Javid HAS Preston's BOOK
Susan HAS Susan's BOOK
```

Problem Constraints

`T` ≤ 100

0 ≤ `n` < 100

It is possible for students to turn in their own book.

Assume all names listed plus the absent student are the only students in the class.

Assume all students' names are unique and consist of only one word.

Problem Author: Chris Liu

Problem 4: Frames and Phrases
11 Points

Problem ID: `frame`

Rank: 2

Introduction

Sometimes you hear a phrase so good, you just want to frame it and hang it up in your room. With the power of computer programming, we can do that digitally!

Your task is to create a program that will take in a phrase and output the phrase inside a frame of asterisks.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A single line containing a phrase or word of length `n` characters.

Example Input:

```
4
Live Laugh Love
Bless This Mess
i have high blood pressure
A
```

Program Output

For each test case, your program should output the input phrase surrounded by a rectangular frame, with each word (counted by spaces) on a separate line. The frame must surround the text with no margins. The capitalization of the words must be conserved. Each test case output should be separated by a blank line.

Example Output:

```
*******
*Live *
*Laugh*
*Love *
*******

*******
*Bless*
*This *
*Mess *
*******

**********
*i       *
*have    *
*high    *
*blood   *
*pressure*
**********

***
*A*
***
```

Problem Constraints

`T` ≤ 100

0 < `n` ≤ 500

The input phrase may consist of multiple words.

The input phrase may have certain letters capitalized.

The input phrase may contain symbols or numbers.

Problem Author: Chris Liu

Contributor(s): Sunny Guan

Problem 5: Fair Ferries Have Fair Ferry Fares
13 Points

Problem ID: `ferries`

Introduction

You're planning a road trip for your summer break, and you just can't wait! There's just one problem: in your world, cars do not exist. Instead, a large collection of roads and a robust ferry system is responsible for getting people where they need to go. However, ferries cost money, and you don't want to be wasteful. Can you plan a trip route that saves you the most money?

Your task is to create a program that will take in a map of roads and docks and output the minimum cost needed to complete a given trip.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A first line consisting of four values separated by spaces. The values are as follows:
  - A positive integer `n` denoting the total number of roads and docks.
  - A positive decimal amount `f` denoting the cost of one ferry ticket in dollars.
  - A name representing the starting location of your trip.
  - A name representing the destination of your trip.
- The next `n` lines each consist of either a road or dock.
  - Roads connect two locations by land and are denoted in the format:
    `ROAD <NAME1> <NAME2>`
  - Docks give a location ocean access and are denoted in the format:
    `DOCK <NAME>`
- A blank line separating individual test cases.

Example Input:

```
3
9 $1.00 Farica Zabril
ROAD Liche Zabril
ROAD Napama Zabril
ROAD Biseria Napama
DOCK Dorjan
ROAD Rausi-Daabia Dorjan
ROAD Biseria Rausi-Daabia
ROAD Rasiel Lapestine
ROAD Rasiel Farica
DOCK Lapestine

2 $12.00 Sursia Nicha
DOCK Sursia
ROAD Rani Nicha

6 $4.50 Fan-Sanfrisco Presno
ROAD Banta-Sarbara Cranta-Suz
ROAD Presno Cranta-Suz
DOCK Presno
DOCK Fan-Sanfrisco
ROAD Banta-Sarbara Fan-Sanfrisco
ROAD Frostand Fulicer
```

Program Output

For each test case, your program should output the minimum cost needed to complete the given trip. Your output should be created based on the following criteria:

- Your output should be in the format: `Bring $<MIN-COST> for the trip.`
    - `<MIN-COST>` should be rounded to two decimal places.
- Roads allow you to walk between locations freely without charge.
- Docks allow you to take a ferry to other docks for a fee `f`
- If the trip does not cost anything, your program should output `The trip's free!`
- If it is impossible to complete the given trip, your program should output `Just stay home`

Example Output:

```
Bring $1.00 for the trip.
Just stay home
The trip's free!
```

Problem Constraints

`T` ≤ 50

1 ≤ `n` ≤ 2.0×10^4

1.0 ≤ `f` ≤ 1.5×10^5

Assume the starting location and the ending location both appear in the input.

Assume the starting location and the ending location are different.

Assume there are no duplicate roads or docks.

Problem Author: Chris Liu

Problem 6: Spiraling Out of Control
13 Points

Problem ID: `spiral`

Rank: 2

Introduction

Who doesn't like a good spiral? Whether you draw them from the outside-in or inside-out (or some other way like a monster), everyone can appreciate a neat, even swirl.

Your task is to create a program that will output a spiral with the given amount of loops.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will consist of single positive integer `n` denoting the number of loops the spiral must have.

Example Input:

```
3
1
2
3
```

Program Output

For each test case, your program should output a spiral based on the following criteria:

- The spiral must consist of `@` characters and begin at the top left corner, looping clockwise.
    - A loop is completed when an upper edge of the spiral is drawn within a perimeter of `@` characters.
- There must be a one-character margin between adjacent edges of the spiral.
- Each test case output should be separated by a blank line.

Example Output:

```
@@@@@
    @
@@@ @
@   @
@@@@@

@@@@@@@@@
        @
@@@@@@@ @
@     @ @
@ @@@ @ @
@ @   @ @
@ @@@@@ @
@       @
@@@@@@@@@

@@@@@@@@@@@@@
            @
@@@@@@@@@@@ @
@         @ @
@ @@@@@@@ @ @
@ @     @ @ @
@ @ @@@ @ @ @
@ @ @   @ @ @
@ @ @@@@@ @ @
@ @       @ @
@ @@@@@@@@@ @
@           @
@@@@@@@@@@@@@
```

Problem Constraints

`T` ≤ 150

`n` ≤ 150

Problem Author: Chris Liu

Problem 7: Checkmate!
13 Points

Problem ID: `queens`

Rank: 2

Introduction

Having more than two queens on a chessboard is strange enough—but what if we didn't stop there? The chessboards in this problem are certainly not standard—they might not even be square! One thing is for certain: this game is far from fair.

Your task is to create a program that will output the number of spots a king will be in checkmate on a chessboard.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A first line consisting of two positive integers `r` and `c` separated by an `x`, denoting the number of rows and columns in the chessboard that will follow.
- A chessboard with the given dimensions following the first line. The chessboard is as follows:
  - Empty spaces are represented by a dash `-`
  - Spots with a queen on them are represented by the character `Q`
  - Queens are the only pieces on the chessboard.
- A blank line separating individual test cases.

Example Input:

```
7
3x4
----
Q---
----

4x4
Q-Q-
-QQ-
----
----

1x3
---

2x1
Q
-

4x4
-Q--
----
----
----

5x5
-----
-----
--Q--
-----
-----

5x8
--------
--------
--Q--Q--
--------
--------
```

Program Output

For each test case, your program should output the number of spots a king would be in checkmate on the chessboard. A chess king is in checkmate if it is currently under attack and is unable to move a safe spot. Chess kings can move one square in any direction (including diagonals).

Example Output:

```
2
12
0
1
1
0
10
```

Problem Constraints

`T` ≤ 60

1 ≤ `r`, `c` ≤ 250

Assume capturing a queen is not a move available to you.

Problem Author: Chris Liu

Contributor(s): Surya Rajan

Problem 8: Radiation Irritation
29 Points

Problem ID: `microwave`

Rank: 3

Introduction

Your dorm room kitchen has the worst microwave you've ever seen. The inside is filthy and sticky, the door hinge is creaky, and the noise it makes when heating could wake everyone on your floor. However, the worst part about this microwave, by far, is its number pad. The buttons jam constantly, and you need to press them with a force so great that you're probably denting the exterior with every push. To make things worse, your microwave is ridiculously weak, meaning you need to compensate with extra-long cook times. One day, you've had enough. You vow to use as few button presses as humanly possible—never again cramping your fingers more than necessary.

Your task is to create a program that will output the shortest sequence of button presses needed to set a microwave timer.

Program Input

The first line of the input from STDIN will contain a positive integer `T` denoting the number of test cases that follow. Each test case will have the following input:

- A single line consisting of two numbers separated by a space. The numbers are as follows:
  - A positive integer `n` denoting the time duration you want to set on the microwave, in seconds.
  - A positive integer `m` denoting the how much the microwave's "Add" button adds to the timer, in seconds.

Example Input:

```
5
15 3
1359 453
300 30
12222223 1111111
23 17
```

Program Output

For each test case, your program should correctly output the shortest sequence of buttons needed according to the following criteria:

- There are two types of buttons on your microwave. They are as follows:
  - Number buttons, represented by a single 0-9 digit. Pressing a number button appends its digit to the end of the current timer duration. Pressing 0 when the current timer duration is zero will have no effect on the timer.
  - The "Add" button, represented by the word `Add`. Pressing the "Add" button adds `m` seconds to the current timer duration.
- Your output should be in the format: `<BUTTON>-<BUTTON>-<...>`
  - The leftmost `<BUTTON>` represents the first button pressed, and the rightmost `<BUTTON>` represents the last button pressed.
- All button presses must occur before the microwave begins cooking.
- If multiple button sequences share the shortest length, your program should output the sequence that presses the least unique buttons. If multiple button sequences of the same length press the same number of unique buttons, your program should output the sequence that comes first in alphanumeric order. The "Add" button should be considered as last in alphanumeric order.

Example Output:

```
1-5
Add-Add-Add
Add-0
Add-2-Add
6-Add
```

Problem Constraints

Test Set #1:

`T` $\leq 1.0\times10^{5}$

$1 \leq$ `n`, `m` $\leq 1.0\times10^{10}$

Test Set #2:

`T` ≤ 150

$1 \leq$ `n`, `m` $\leq 1.0\times10^{19}$

Assume `n` is not preceded by a zero.

Note: If you are a Java or C/C++ programmer, be aware that the `int` variable type is too small to contain values `n` and `m`! Java programmers should use variable type `long`, and likewise `long long` for C/C++.

Problem Author: Chris Liu