Editor's Choice Spring '22

April 21, 2022
No Time Limit
4 Questions
Problem Packet by CALICO

Table of Contents

Problem 1: Decks Against Programmability
3+2 Points

Problem ID: `decks`
Rank: 1+2

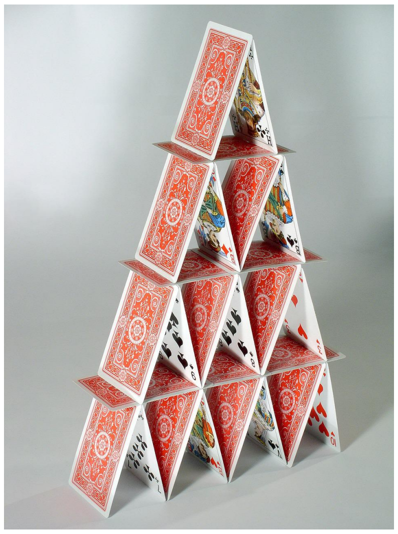
Introduction

You've built your digital house of cards, but your friends aren't impressed! Instead, they insist a real life house of cards would be much cooler! Being a lazy programmer lacking patience and agility, you decide to hire a robot to build the house for you. Before the robot can get started however, you'll need to buy some cards, which come in decks of 54.

Problem

Find the smallest number of decks required in order to build a house of cards with $\mathbf{N}$ layers.

In a house of cards, each layer has pairs of diagonal cards leaning on each other. The top layer has 1 pair, the next layer has 2 pairs, and so on such that the bottom layer has $\mathbf{N}$ pairs. Each pair is supported by a horizontal card beneath it, except for pairs in the bottom layer. A house of cards with 4 layers is shown below.



Input Format

The first line of the input contains a positive integer $\mathbf{T}$ denoting the number of test cases that follow. Each test case consists of a single line containing a positive integer $\mathbf{N}$ denoting the number of layers in the house of cards you're trying to build.

Output Format

For each test case, output a single line containing an integer denoting the smallest number of decks needed to build the house of cards.

Constraints

This problem has a bonus set! Test sets are independent of each other, so you don't need to solve the bonus set in order to get points for the main set. Furthermore, if you have a working solution for the bonus, you can submit it to get full points on the main test set!

$1 \leq \mathbf{T} \leq 100$
$\mathbf{N}$ is even.

Main Test Set

$1 \leq \mathbf{N} \leq 10^3$

Bonus Test Set

$1 \leq \mathbf{N} \leq 10^9$

Sample Test Case

Sample Input:

```
5
1
6
10
108
1337
```

Sample Output:

```
1
2
3
325
49668
```

Sample Explanations:

For Test Case #1:
A house of 1 layer has 2 cards, so we need at least 1 deck (54 cards).

For Test Case #2:
A house of 6 layers has 57 cards, so we need at least 2 decks (108 cards).

For Test Case #3:
A house of 10 layers has 155 cards, so we need at least 3 decks (162 cards).

For Test Case #4:
A house of 108 layers has 17550 cards, so we need at least 325 decks (17550 cards).

Problem 2: Dungeons & Dartboards
7 Points

Problem ID: `darts`
Rank: 2

Introduction

Who doesn't like a good game of darts? While the dartboards in this problem are not tournament-standard, that doesn't make them any less fun! You want to make a dartboard of any size, but drawing rings is tedious—so you want to write a program that will make your life easier.

Problem Statement

Your task is to output a dartboard with $\mathbf{N}$ rings.

A dartboard is a grid of characters with the following criteria:

- The dartboard must have alternating rings of `X`'s and spaces.
- The dartboard must be square.
- The dartboard must have an outer ring of `X`'s. The inner ring may be either `X`'s or spaces.

Input Format

The first line of the input contains a positive integer $\mathbf{T}$ denoting the number of test cases that follow. Each test case consists of a single positive integer $\mathbf{N}$ denoting the number of rings the dartboard must have.

Output Format

For each test case, your program should output a dartboard with $\mathbf{N}$ rings. Each test case output should be separated by a blank line.

Problem Constraints

1 ≤ **T** ≤ 100

1 ≤ **N** ≤ 100

Sample Test Case

Sample Input

```
3
3
1
8
```

Sample Output

```
XXXXX
X   X
X X X
X   X
XXXXX

X

XXXXXXXXXXXXXXX
X             X
X XXXXXXXXXXX X
X X         X X
X X XXXXXXX X X
X X X     X X X
X X X XXX X X X
X X X X X X X X
X X X XXX X X X
X X X     X X X
X X XXXXXXX X X
X X         X X
X XXXXXXXXXXX X
X             X
XXXXXXXXXXXXXXX
```

Problem 3: Blowhole Blues
7+4 Points

Problem ID: `blowholes`
Rank: 2+3

Introduction

Switching from computer science to learning marine biology has been the best choice of your life. You've recently been placed in charge of the Bay Area's newest and trendiest traveling whale show, and your first act is next week! However, your most ambitious display—the Blowhole Blast Bonanza—is in desperate need of work. Your whales need to synchronize the heights of their blowhole streams, but ever since you've dissolved their labor union and slashed their healthcare packages, they've been hopelessly disorganized. The first show is fast approaching, so you've begrudgingly decided to accept imperfection for the sake of time. Hopefully the audience won't notice...

Problem Statement

Your task is to output the smallest number of adjustments needed to ensure all $\mathbf{N}$ stream heights $\mathbf{S_1}$, $\mathbf{S_2}$, $\mathbf{\dots}$, $\mathbf{S_N}$ are within a given range $\mathbf{K}$ of each other.

- An adjustment is defined as setting the height of a whale's blowhole stream to any other height. The size of the adjustment is not relevant.
- Your adjustments need to satisfy the requirement that no two whales have blowhole stream heights more than $\mathbf{K}$ feet apart. In other words, all whales' blowhole stream heights should be within $\mathbf{K}$ feet of each other.

Input Format

The first line of the input contains a positive integer $\mathbf{T}$ denoting the number of test cases that follow. For each test case:

- The first line consists of two space-separated values $\mathbf{N}$ $\mathbf{K}$ where:
  - The positive integer $\mathbf{N}$ denotes the number of whales in your show.
  - The non-negative integer $\mathbf{K}$ denotes the maximum height difference you'll accept between any two whales, in feet.
- The second line contains a space-separated sequence of $\mathbf{N}$ non-negative integers $\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_N}$, denoting that whale i has a blowhole stream $\mathbf{S}_i$ feet tall.
- A blank line follows to separate individual test cases.

Output Format

For each test case, your program should output an integer denoting the minimum number of adjustments needed to satisfy the given criteria.

Problem Constraints

T $\le \mathbf{T} \le 100$
$0 \le \mathbf{K}, \mathbf{S_{1..n}} \le 10^9$

Main Test Set

$1 \le \mathbf{N} \le 10^3$
The sum of $\mathbf{N}$ across all test cases in an input does not exceed 10^3.

Bonus Test Set

$1 \le \mathbf{N} \le 10^5$
The sum of $\mathbf{N}$ across all test cases in an input does not exceed 10^5.

Sample Test Case

Sample Input:

```
4
3 1
3 1 4

3 0
3 1 4

6 67
68 69 70 221 253 255

7 67
68 69 70 219 221 253 255
```

Sample Output:

```
1
2
3
3
```

Sample Explanations:

For Test Case #1:
In order to have all blowhole stream heights be within 1 foot of each other, you could adjust whale 2's blowhole stream to be either 3 or 4 feet tall.

For Test Case #4:
In order to have all blowhole stream heights be within 67 feet of each other, you could adjust whales 1, 2, and 3's blowhole streams to be within range of whales 4, 5, 6, and 7's streams. One such adjustment could be assigning whales 1, 2, and 3 all stream heights of 230 feet.

Problem 4: why did nobody invite me to prom
13 Points

Problem ID: `flights`
Rank: 3

Introduction

You and your neighborhood friend Itani Mulli have finally finished building a drone to spy on the people going to prom without you! After testing it in the comfort of your own backyard, you decide to put it to the test by flying it to his house. However, your cranky neighbor Dory Toh absolutely hates the new century and has no patience for your newfangled technology flying over her precious hydrangeas. Will you be able to fly your drone to Itani's house without facing even more lawsuits?

Problem Statement

Given the corner coordinates of Dory's triangular backyard ($\mathbf{X_1}$, $\mathbf{Y_1}$), ($\mathbf{X_2}$, $\mathbf{Y_2}$), ($\mathbf{X_3}$, $\mathbf{Y_3}$) and the start and end coordinates of your drone's planned flight ($\mathbf{X_S}$, $\mathbf{Y_S}$), ($\mathbf{X_E}$, $\mathbf{Y_E}$), determine if the direct flight path travels through your neighbor's backyard.

All coordinates are given as a pair of integers in the 2D plane.

Although Dory is cranky, you can get away with the following:

- Flight paths that travel along the boundary of their backyard without entering inside are acceptable
- Flight paths that travel through a single corner of their backyard are also acceptable

Finally, the start and end coordinates of the drone's flight path are guaranteed to be strictly outside of Dory's backyard.

Input Format

The first line of the input contains a positive integer $\mathbf{T}$ denoting the number of test cases that follow. For each test case:

- The first line contains 4 space-separated integers $\mathbf{X_S}$ $\mathbf{Y_S}$ $\mathbf{X_E}$ $\mathbf{Y_E}$ where:
  - $\mathbf{X_S}$ $\mathbf{Y_S}$ denotes the start coordinate of your flight path in 2D space
  - $\mathbf{X_E}$ $\mathbf{Y_E}$ denotes the end coordinate of your flight path in 2D space
- The second line contains 6 space-separated integers $\mathbf{X_1}$ $\mathbf{Y_1}$ $\mathbf{X_2}$ $\mathbf{Y_2}$ $\mathbf{X_3}$ $\mathbf{Y_3}$ where:
  - $\mathbf{X_1}$ $\mathbf{Y_1}$ denotes the first corner coordinate of Dory's backyard in 2D space
  - $\mathbf{X_2}$ $\mathbf{Y_2}$ denotes the second corner coordinate of Dory's backyard in 2D space
  - $\mathbf{X_3}$ $\mathbf{Y_3}$ denotes the third corner coordinate of Dory's backyard in 2D space
- The third line is blank to separate individual test cases

Output Format

For each test case:

- If the flight path does not travel through Dory’s backyard, output a single line containing:
  `SAFE`
- If the flight path travels through Dory’s backyard, output a single line containing:
  `I'LL SEE YOU IN COURT`

Constraints

$1 \le \mathbf{T} \le 100$
$-10^4 \le \mathbf{X}, \mathbf{Y} \le 10^4$
The triangle corner coordinates are not collinear.
All given coordinates are unique.
The start and end coordinates of the flight path are strictly outside the triangle.

Sample Test Case

Sample Input:

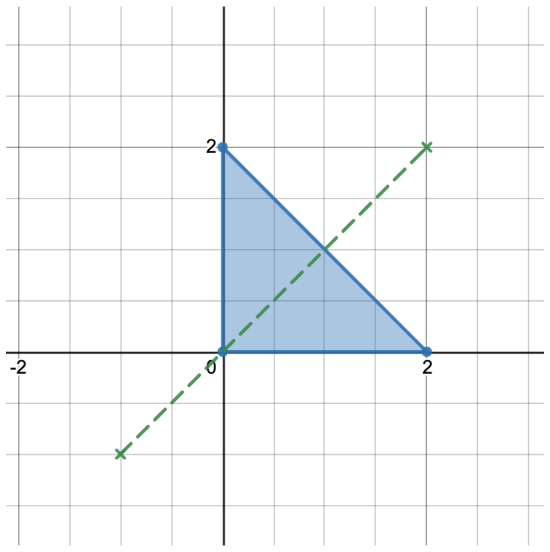
```
4
4 4 -2 6
-2 -2 -4 0 2 3

-1 -1 2 2
0 0 2 0 0 2

2 6 2 -2
2 2 4 6 6 0

3 11 6 -4
-5 3 4 6 5 1
```

Sample Output:

```
SAFE
I'LL SEE YOU IN COURT
SAFE
SAFE
```

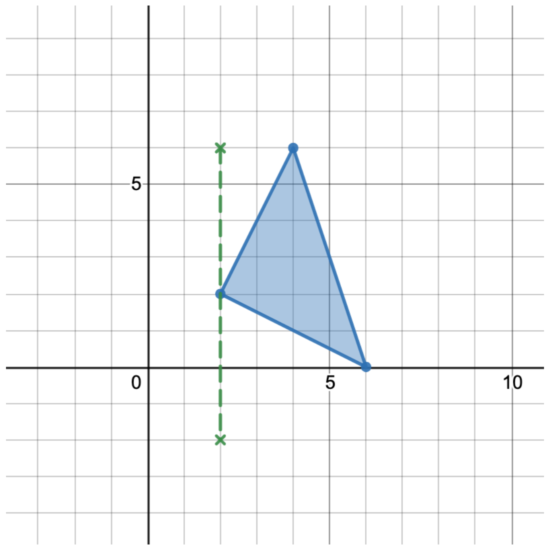
Sample Explanations:

For Test Case #2:
The given flight path crosses through the boundaries laid out by the three points, so the flight path passes through Dory's backyard.



For Test Case #3:

The given flight path only crosses over a corner of Dory's backyard, so the flight path is safe.



For Test Case #4:

The given flight path travels along a boundary, but doesn't cross over it. Therefore, the flight path is safe.