Fifteenth Annual
University of Central Florida

High School Programming Tournament: Online Edition

Problems – Division 1

Problem Name	Filename
Apocalypse Attack!	apocalypse
Camelot's Conundrum	camelot
Cost Cutting	cutting
The Ducks of Danger Island	ducks
GAMDAR	gamdar
Optical Illusion	illusion
The Rainbow Lightseba™	lightseba
Gravitational Paradox	paradox
All Roads Lead to Rome	rome

Call your program file:
filename.cpp, *filename*.java, or *filename*.py

For example, if you are solving The Ducks of Danger Island,
Call your program file:
ducks.cpp, ducks.java, or ducks.py
Call your Java class: ducks

Apocalypse Attack!

Filename: `apocalypse`

In a land long gone and a time long forgotten, two titans clashed in a battle that would define an era. Zephyra, a formidable sorceress with control over gravity itself, dueled Aetherius, a master of agility and grace.

Wishing to end the fight, Zephyra bent the forces of the world to her will. The atmosphere crackled with anticipation as she raised her hands, invoking the power of her signature move: the Apocalypse Attack. Meteors rained down from the cosmic expanse, their fiery trails tracing the descent toward her opponent, Aetherius. On the ground, where Aetherius stood, an immense gravitational field took hold, making it hard to move.

With tranquility unmatched even by the sea on its calmest day, Aetherius gazed upward, acknowledging the celestial tempest descending upon him. Due to the gravitational field, he knew that moving would be difficult, and dodging would be even harder. Quickly calculating the path with the least movement, he initiated his sequence, appearing effortless in his evasion and escaping many meteors by only the breadth of an inch. How did he do it?

Assume that Aetherius only moves on a one-dimensional number line and that his width is negligible. The meteors rain down in a fixed sequence and are defined by their center of impact and width on the number line. If Aetherius is on or beyond the edge of the meteor when it impacts, he successfully dodges the meteor. Assume that after Aetherius evades one meteor, he will have enough time to move to any location on the number line before the next meteor hits. Find out the minimum distance Aetherius must travel to dodge all of the meteors.

The Problem:

Given Aetherius's starting position and the ranges of impacts of meteors that fall down one by one, determine the minimum total distance Aetherius must travel to dodge all the meteors.

The Input:

The first line consists of two positive integers, n and s ($1 \le n \le 10^5$; $1 \le s \le 10^5$), representing the number of meteors that rain down and Aetherius's initial position, respectively. The next n lines each contain two integers, l_i and r_i ($1 \le l_i \le r_i \le 10^5$), representing the range of impact of the i^{th} meteor to fall, respectively.

The Output:

Output a single line containing a single integer: the minimum distance Aetherius must move to evade the meteors.

Sample Input 1:

```
1 5
1 7
```

Sample Output 1:

```
2
```

Sample Input 2:

```
3 7
6 12
1 9
8 14
```

Sample Output 2:

```
5
```

Camelot's Conundrum

Filename: `camelot`

In the ancient hall of Camelot, King Arthur sat in solemn contemplation, surrounded by his loyal knights gathered at the venerable Round Table. The once-prosperous kingdom now faced trials, for King Arthur, while valiant in battle, found that mastery in the economic arts and design of infrastructure was not in his long list of virtues.

As the assembly convened, Sir Lancelot, a paragon of chivalry, arose with a dire message. In a voice that resonated through the hallowed hall, he spoke, "My noble king, tidings of discontent echo across the realm. The Union of Merchants, aggrieved by economic hardships, murmurs of rebellion. Their demand: a new two-way road each passing year, linking two different towns that previously lacked a direct connection."

Yet, as the echoes of Lancelot's warning lingered, Sir Gawain, a knight of dauntless courage, stepped forth. "Sire, grave news reaches my ears. Mordred, that scheming scoundrel, has formed an unholy alliance with our Saxon neighbors. They silently infiltrate our sacred realm. Once the towns are linked such that Saxon invaders may reach every town from every other town, a war shall be waged upon us."

With profound wisdom befitting his regal station, King Arthur pondered the twin specters of economic discontent and impending war, quickly determining how many years he had to find a solution to stave off civil unrest and secure the kingdom's future..

The Problem:

Given the set of towns and roads connecting them, determine the maximum number of years King Arthur has before the kingdom is plunged into war. He must add one new two-way road every year, and war will occur when all of the towns are reachable from each other.

The Input:

The first line consists of two positive integers, n and m ($2 \le n \le 10^5$; $0 \le m \le 10^5$), representing the number of towns in the Kingdom of Camelot and the number of existing roads, respectively. Every town is assigned a unique number from 1 to n. The next m lines each contain two integers, a and b ($1 \le a \le n$; $1 \le b \le n$; $a \ne b$), representing two towns that are connected by a road. Every road is bidirectional. It is guaranteed that no pre-existing road connects the same two cities as another road. It is also guaranteed that the invaders can't currently reach every town from every other town, since the war has not yet started.

The Output:

Output a single line containing an integer, representing the maximum number of years that King Arthur can stave off war.

Sample Input 1:	**Sample Output 1:**
`2 0`	`1`

Sample Input 2:	**Sample Output 2:**

`5 3`
`1 2`
`2 3`
`3 1`
	`4`

Cost Cutting

Filename: `cutting`

Thing One and Thing Two have a birthday coming up, and the Cat in the Hat is preparing their cake! Unfortunately, the Cat's budget has been affected by a recent economic downturn; so, to save money, he has only bought the Things one cake to share. The Cat is bad at geometry, so he models this cake as an infinite coordinate plane.

The Cat also cannot afford enough candles for each Thing to get a number equal to their age. Luckily, as the Cat knows, the Things are satisfied as long as they each get the exact same number of candles. He has thus placed an even number of candles on the cake to split between the two Things.

The Cat needs to find a correct line along which to cut the cake so as to ensure both sides have an equal number of candles. If this cut goes through a candle, that candle is removed from the cake, and neither Thing receives any part of this candle.

The Cat has consulted you, the Honorable Slicer of Pastries and Tarts (HSPT), to find such a line for him to cut along.

The Problem:

Given the integer coordinates of candles on the Things' (infinitely large) cake, determine a line, represented by two points, along which to cut the cake, so that the resulting two pieces have an equal number of candles. If this line goes through any candles, they are removed from the cake entirely.

The Input:

The first line of input consists of an integer n ($2 \leq n \leq 10^3$; n is even), representing the total number of candles.

The next n lines each contain two integers x and y ($-10^9 \leq x, y \leq 10^9$), representing the coordinates of the candles. No two candles will be at the same point. Additionally, no three candles will be collinear. That is, it is impossible to draw a line that passes through three or more of the candles.

The Output:

Output two lines of text, each with two space-separated decimal values x_1, y_1 and x_2, y_2 respectively ($-10^9 \leq x_1, y_1, x_2, y_2 \leq 10^9$), representing the coordinates of two points on the line along which to cut. These points must be at a distance of at least 10^{-6} from each other.

Each coordinate must contain **no more than six** digits after its decimal point. A candle is considered to be on the output line if it is within a distance of 10^{-6}.

If there are multiple solutions, you may output any of them.

Sample Input 1:	**Sample Output 1:**
4 1 1 2 -3 -1 3 -2 -2	-2.000000 -3.000000 2.000000 3.000000

Sample Input 2:	**Sample Output 2:**
6 2 2 3 1 1 -3 -2 1 -2 -2 -3 -1	-3.421374 -2.339812 3.757521 2.484212

The Ducks of Danger Island

Filename: `ducks`

A mysterious magic took hold of Danger Island one night, leaving all of its quacking inhabitants with the remarkable ability to shoot bolts of lightning from their feet. The ducks were a peaceful population, and seldom felt the need to use their electrifying powers. But one fateful day, the clear sky turned stormy, and a flying pirate ship descended from the gathering gray clouds. It was an ambush by the Marauding Macaws!

The ducks of Danger Island hatched a daring plan. On a pedestal in the center of the island sat the Webbed Water Gem, a magical artifact with the ability to enhance the range and power of the ducks' lightning bolts. As told in the ancient scriptures of Danger Island, a duck must balance the gem on his foot in order to receive its magic. The ducks decided to line up and face this unstable undertaking one at a time.

Some of the ducks are more surefooted than others. Specifically, each duck has a certain probability of dropping the gem while balancing it. If a duck drops the gem, he will still receive the gem's magic, but the gem will shatter and become unusable for all the ducks after him in line.

The battle cries of the Marauding Macaws drew steadily closer, and a critical question hung in the air: How could the ducks of Danger Island ensure their victory?

The Problem:

Given the probability that each duck will drop the Webbed Water Gem, print the expected number of ducks that will have their lightning powers amplified if they line up in the best way.

The Input:

The first line of input contains an integer, n ($1 \le n \le 10^5$), representing the number of ducks on Danger Island. The second line of input contains n integers, where the i^{th} integer p_i ($0 \le p_i \le 100$), represents the percent chance that the i^{th} duck will drop the gem while balancing it.

The Output:

Output the expected number of ducks that will have their lightning powers amplified. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-3}.

Sample Input 1:	**Sample Output 1:**
5 0 100 0 100 0	4

Sample Input 2:	**Sample Output 2:**
2 50 50	1.5

Sample Input 3:	**Sample Output 3:**
10 50 79 48 39 41 53 98 17 32 50	3.14168

GAMDAR

Filename: `gamdar`

GAMDAR™ is the newest technology for quantum particle mapping, invented by its namesake Glenn "gamucf" Martin. This technology is quite useful when mapping High-Speed Particle Trajectories (HSPTs); however, with quantum particles, we can never quite be sure where the particles are.

When planning HSPTs, one useful measurement that GAMDAR needs to supply is the Manhattan distance between two particles. GAMDAR has narrowed down the possible locations for the particles to a set of three-dimensional points, so it will report the expected Manhattan distance between two randomly selected points. Each point selection is independent, so it is possible to have both selected points be the same point. In addition, selecting two points in a different order is considered a different pair. Each point has an equiprobable chance of being selected. You have been asked to find the expected Manhattan distance to help implement this feature.

Note: In three dimensions, the Manhattan distance between two points (x_1, y_1, z_1) and (x_2, y_2, z_2) is defined as $| x_2 - x_1 | + | y_2 - y_1 | + | z_2 - z_1 |$.

The Problem:

Determine the expected Manhattan distance between two randomly selected points from a list.

The Input:

The first line of input contains a single integer, n ($1 \le n \le 10^5$), representing the number of points in the set. The next n lines will each contain three integers, x, y, z ($-10^9 \le x, y, z \le 10^9$), representing the x, y, and z coordinates of a point.

The Output:

Output a single line containing a single real number: the expected Manhattan distance between two randomly chosen points. The expected Manhattan distance must be accurate to an absolute or relative error of 10^{-6}.

Sample Input 1:

```
3
0 0 0
1 1 1
2 2 2
```

Sample Output 1:

```
2.666666666667
```

Sample Input 2:

```
3
0 1 0
5 2 3
6 1 4
```

Sample Output 2:

```
4.888888888889
```

Optical Illusion

Filename: `illusion`

Anders loves optics. As he was exploring a funhouse, he entered his favorite room: the mirror maze! This maze isn't composed of typical mirrors, though. Instead, these mirrors only reflect portions of an image and leave the rest of the image the same.

This completely broke everything Anders knew about optics, and he started exploring what different things looked like in these mirrors. He held up an array of items and the mirrors reflected a certain range of these items. He sent you a picture of this peculiar reflection along with the range of items that were reflected and challenged you to determine what the original array of items looked like.

The Problem:

Given an array of items, represented by integers, and a range that was reflected, determine what the original array looked like. A reflection will be represented by the corresponding subarray being reversed.

The Input:

The first line of input will consist of three integers: n, L, and R ($1 \le L \le R \le n \le 10^5$) where n is the size of the array, and L and R represent the range of indices (inclusive) that was reflected in the array. The second line of input will consist of n integers between 0 and 100, inclusive, representing the elements at each position in the reflected array.

The Output:

Output a single line consisting of n integers: the original array that Anders had.

Sample Input 1:	**Sample Output 1:**
`4 2 3` `1 3 2 4`	`1 2 3 4`

Sample Input 2:	**Sample Output 2:**
`6 4 6` `100 5 51 51 92 39`	`100 5 51 39 92 51`

The Rainbow Lightseba™

Filename: `lightseba`

Thomas, a freshly trained padawan, is about to create his very own Lightseba™. Recent advancements in kyber crystal technology have revolutionized these sebas, allowing for multiple colors across distinct sections. Amazed by the opportunity, Thomas has bombarded his helpful droid with many requests to color sections of the seba. These requests come in the form *L, R, C,* where *L* and *R* represent the inclusive endpoints of the section to be colored with the color *C*.

When the droid receives a new request, it will use high energy proton lasers to erase the previous color in that section and then—using food coloring, a bolt of superheated plasma, some shrimp fried rice, and a bucket of midichlorians—will engage in an advanced secretive process to color the new section with the desired color. Thus, previously colored segments will be replaced with the new requested color. Since it is the objectively best Lightseba color, blue is the starting color of all Lightsebas (indicated by a B) and any sections that are not recolored in any request will remain blue.

Thomas has already sent off his list of customization requests to the droid, but, because of his impatience, he wants to know the final colors of his new Lightseba. Please help him out. You are his only hope.

The Problem:

Given a blue Lightseba, determine what the Lightseba will look like after an ordered series of coloring requests.

The Input:

The first line contains two integers: N ($1 \leq N \leq 10^5$), representing the length of the Lightseba, and Q ($1 \leq Q \leq 10^5$), representing the number of coloring requests. Q lines follow, each containing 2 integers, L and R ($1 \leq L \leq R \leq N$), and one uppercase letter C (A-Z) representing a request to color the section L to R of the Lightseba with the color C.

The Output:

Output a single line containing a string of uppercase letters representing the final state of the Lightseba after all requests are processed in order. All sections uncolored by any request will remain blue (B).

Sample Input 1:	**Sample Output 1:**
6 3 1 2 G 6 6 R 5 6 P	GGBBPP

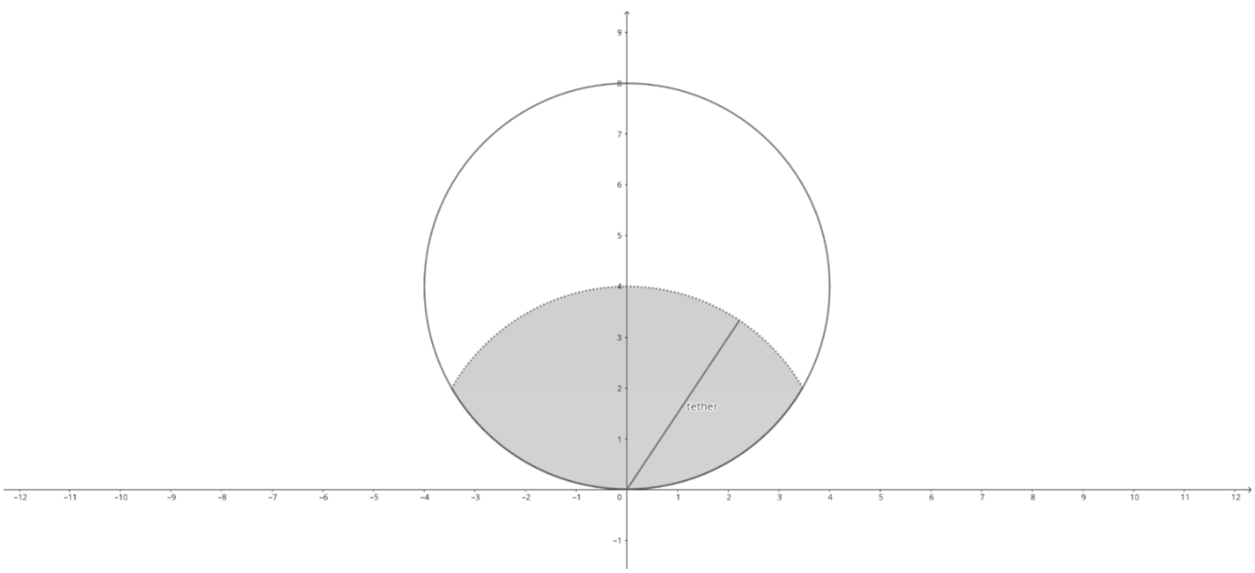
Sample Input 2:	**Sample Output 2:**
5 1 1 5 R	RRRRR

Gravitational Paradox

Filename: `paradox`

Many argue between a round Earth and a flat Earth, but this is merely a distraction from the truth: hollow Earth. Despite what Big Geology says, the Earth is merely a shell surrounding empty space. To prove this, John Doe (who has chosen to keep his real identity anonymous) will travel to the center of the Earth. Although he is not the brightest, he knows that gravitational effects are negligible inside a hollow body, which seems quite paradoxical to him. To account for this, he decided to tether himself to the inside surface of the Earth. He has asked for your help in determining the length of the tether he should bring.

John Doe has modeled this problem in two dimensions because three dimensions are too many for his feeble mind. He will model the Earth as a circle with diameter d, centered on the y-axis $d/2$ units above the origin. He will enter the circle from the origin. He has asked you to determine the minimum length of a tether needed such that he can float around exactly p percent of the circle's area.



Sample 1: A tether with a length of 4 units covers about 39.1% of the total area of an Earth with a diameter of 8.

The Problem:

Given the diameter of the Earth modeled in 2D and a target percentage, determine the minimum length of a tether John Doe needs to be able to reach the target percentage of the Earth's inner area.

The Input:

The input will consist of a single line containing an integer, d ($1 \le d \le 10^5$), and a real number, p ($0 \le p \le 1$), with at most six decimal places, representing the diameter of the Earth and the percentage of the area that needs to be covered, respectively. So if the target percentage was 50%, this could be given as 0.5, 0.50, and so forth.

The Output:

Output a single line containing a single real number: the length of the tether. The length must be accurate to an absolute or relative error of 10^{-4}.

Sample Input 1:	**Sample Output 1:**
`8 0.391003`	`4.00000000`

Sample Input 2:	**Sample Output 2:**
`8 0.50`	`4.63491389`

All Roads Lead to Rome

Filename: `rome`

Romulus, after feuding with his twin brother and establishing the not-yet-magnificent city of Rome, is dreaming of creating a vast empire. This empire shall be the epitome of cultural diversity, flooded with wealth, and filled with the happiest and healthiest populace.

To ensure this empire is successful, Romulus wants to create a trading city which will handle the import of all goods. In order to give the merchants many different choices for how they traverse this trading journey Romulus wants there to be exactly K paths from the trading city to Rome. Since resources are scarce, he can't afford to construct more than 1,000 roads. Each of these roads must go between exactly 2 cities and can only be traveled in one direction. Romulus also does not want any city to be able to reach itself via a positive length path as this may lead to merchants with poor navigation skills getting lost endlessly.

Given that the empire has 500 cities, help Romulus construct a road map using at most 1,000 roads such that there are exactly K routes from the trading city (City 1) to Rome (City 2)

The Problem:

Construct a series of one-way roads such that the number of distinct paths starting at the trading city (City 1) and ending at Rome (City 2) is exactly *K*. Output this as a list of roads to be built. Specifically, output a list of pairs of integers *U* and *V*, indicating that a road is built allowing travel from city *U* to city *V*. The list of roads must fulfill the following requirements:

- The number of routes from the trading city to Rome is exactly *K*
- No city can reach itself via a path consisting of a **positive** number of roads
- The number of roads constructed is at most 1,000
- There is at most one road directly connecting any pair of cities
- All cities have a unique integer label in the range of 1 to 500*,* inclusive

Note: Two paths are considered distinct if they have different lengths, or if they differ in cities at any position along the path.

The Input:

The first and only line of input will consist of a single integer K ($1 \le K \le 10^9$), representing the exact number of distinct routes that must exist between the trading city and Rome (Cities 1 and 2).

The Output:

First, output a line containing a single integer *M***,** representing the number of roads you will construct. Then output *M* lines each containing 2 integers, *U* and *V,* which indicates a road constructed between cities *U* and *V* allowing travel from city *U* to city *V*.

Note: if there are multiple solutions, you may print any of them.

Sample Input 1:	**Sample Output 1:**
2	3 1 2 1 3 3 2

Sample Input 2:	**Sample Output 2:**
3	8 1 7 1 3 7 3 7 5 7 2 5 2 5 9 9 2