Fourteenth Annual
University of Central Florida

High School Programming Tournament: Online Edition

Problems – Division 2

Problem Name	**Filename**
BABA Strings	baba
Blimp Brigade	blimps
Lior's Books	books
First Contact Counting	contact
Diamond Mining	diamond
Traveling Sales Pigeon	pigeon
Anything Under the Sky	sky
No Way Out But Through	through
The *n* Days of Christmas	xmas

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

For example, if you are solving Traveling Sales Pigeon,
Call your program file:
pigeon.cpp, pigeon.java, or pigeon.py
Call your Java class: pigeon

BABA Strings

Filename: `baba`

BABA is a cool sounding word! A "BABA" string is a string consisting of exclusively A's and B's such that the number of A's and B's in the string are equal. Given this property, note that a BABA string can have multiple substrings in it that are also BABA Strings. For example, for the string "BABA" the four BABA strings are BABA, BA, BA, and AB.

BABA the bunny would like to know if a given string is a BABA string. If so, BABA the baby bunny would also like to know how many substrings of that BABA string are also BABA strings. BABA, being a bunny and therefore not understanding math, the English language or the concept of a "substring", is struggling to do this. Would you mind writing a program to help them?

ABBA, the brother of BABA, would like you to know that a substring is defined as non-empty continuous subsequence of characters.

The Problem:

Given a string, determine whether it is a BABA string, and if so, determine how many substrings of the given string are also BABA strings.

The Input:

The input is a single line containing a string, *s*, of length between 1 and 100,000 (inclusive). The string will contain only uppercase A's or B's.

The Output:

Output a single line containing a single integer representing the number of substrings of the given string that are BABA strings. If the inital input string is not a BABA string at all, output "`BABA IS NOT YOU!`" instead (even though these strings may have substrings that are BABA strings,we will ignore that substrings here).

Sample Input 1:	**Sample Output 1:**
`BABA`	`4`

Sample Input 2:	**Sample Output 2:**
`BABBB`	`BABA IS NOT YOU!`

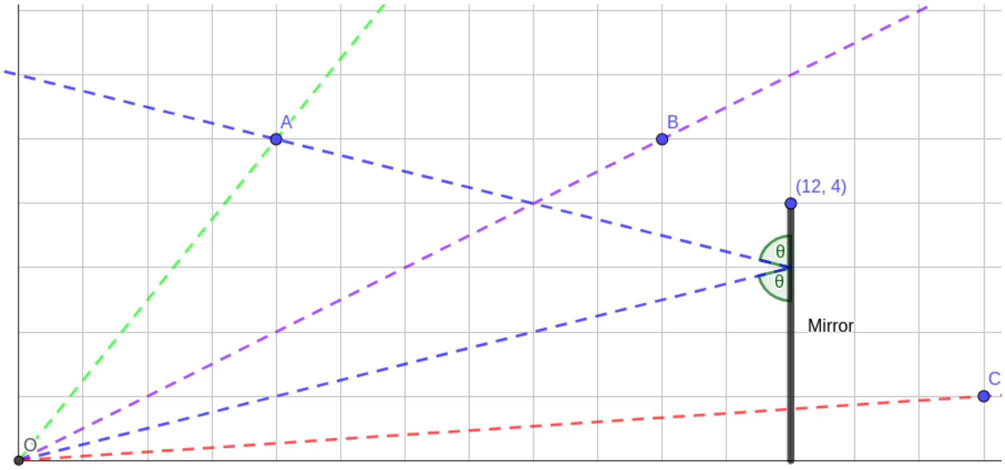
Sample Input 3:	**Sample Output 3:**
`AAABBB`	`3`

Blimp Brigade

Filename: `blimps`

The citizens of Spookyville have been waiting all year, and it's finally time for the annual Halloween Super Party-Tacular! This year, many festive citizens will take to the skies in their homemade party blimps. You reserved your spot months in advance at (0, 0) before construction began on the new Giant Mirror^{TM} close by. At first you were upset—your view is ruined!—but then you realized this might actually result in you being able to see even more blimps than before. Or at least, some blimps may be visible more than once by looking at them in the mirror. You know the locations of the blimps, as well as the x-coordinate and height of the Giant Mirror^{TM}. How many blimps will you be able to see?

As an example, assume the Giant Mirror^{TM} is at x=12 and is 4 units high, and that there are three blimps named A, B and C at (4, 5), (10, 5) and (15, 1), respectively This allows you to see blimp A twice (once by looking directly at it, and once as a reflection in the mirror), blimp B only once (by looking directly at it), and blimp C cannot be seen at all. Thus the number of "blimps" you see is 3. Note that any point on the mirror may reflect a blimp, including the endpoints.



The Problem:

Given the location and height of the Giant Mirror^{TM}, and the locations of the blimps, how many blimps will you be able to see?

The Input:

The input will begin with a single line containing three positive integers, n ($n \leq 100$), x_m ($x_m \leq 1{,}000$) and y_m ($y_m \leq 1{,}000$), representing the number of blimps, the position of the Giant Mirror^{TM} on the x-axis, and the height of the Giant Mirror^{TM}, respectively. Then, n lines will follow, each containing two positive integers, x_i ($x_i \leq 1{,}000$) and y_i ($y_i \leq 1{,}000$), representing the x- and y-coordinates of the i^{th} blimp. No two blimps will be in the same location, and no blimp will be on or within the mirror. A blimp's view is obstructed if any point on the mirror (including the endpoint) blocks the view from your viewpoint. For the purposes of this problem, assume that a blimp does not obstruct the view of another blimp.

The Output:

Output a single integer: the number of blimps you can see.

Sample Input 1:

```
3 12 4
4 5
10 5
15 1
```

Sample Output 1:

```
3
```

Sample Input 2:

```
3 12 4
1 1
2 2
3 3
```

Sample Output 2:

```
6
```

Lior’s Books

Filename: `books`

Lior has become quite the avid reader and has assembled an impressive collection of books. He has stacked these books on a table to varying heights and eventually plans to organize them. For now, Lior will settle with all the book stacks having the same number of books. Since Lior would like to go back to playing Super Auto Pets, he wants to make as few moves as possible. A move consists of taking the top book from any stack and placing it on top of any other stack. Lior has asked you to write a program that will determine whether or not he can make all book stacks the same height. If it is possible, he would also like to know the minimum number of movements necessary so he can complete this tedious task and go back to playing Super Auto Pets as quickly as possible!

The Problem:

Given a list of how many books are in each stack, determine if it is possible to make all of the stacks the same height and if so, what is the minimum number of movements that must be made to make all of the stacks equal height.

The Input:

The input will start with a line containing a single integer, s ($1 \leq s \leq 10^3$) representing the number of book stacks Lior currently has. The following line will contain s integers, h_i ($0 \leq h_i \leq 10^9$), representing the height of the ith stack, respectively.

The Output:

Output a single integer: the minimum number of movements Lior must complete to make all of the stacks have an equal number of books. If it is impossible to do so, output “impossible” instead.

Sample Input 1:	**Sample Output 1:**

`4`
`1 7 5 4`
	`impossible`

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

`2`
`1 5`
	`2`

Sample Input 3:	**Sample Output 3:**

`1`
`5`
	`0`

First Contact Counting

Filename: `contact`

Extraterrestrials have made first contact. Travis has received the honor of being humanity's first liaison with the extraterrestrials, and he has brought you along to help him with the momentous task of communication.

After weeks of trial and error, you and Travis have taught the extraterrestrials about humanity's numerical system. Now, the extraterrestrials are attempting to teach you about their counting system. Their system appears to be a form of base 10. When counting, the numbers start at some positive integer and increment by a constant integer not necessarily equal to 1.

Travis has instructed you to focus your work on the task of understanding this counting system.

The Problem:

Given a sequence of ordered numbers, determine the next number in the sequence.

The Input:

The first line will contain an integer, n ($2 \leq n \leq 10$), representing the number of integers in the sequence. The following line will contain n integers, representing the integers in the sequence. The n integers will be strictly increasing and between 1 and 1,000, inclusive. Each sequence will always be a consistent one (the difference between adjacent numbers will be the same).

The Output:

Output the next number in the sequence.

Sample Input 1:	**Sample Output 1:**

`2`
`1 2`
	`3`

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

`4`
`4 6 8 10`
	`12`

Diamond Mining

Filename: `diamond`

You and your friends have a Minecraft server. After many weeks of playtime, you have gathered a tremendous amount of resources. To organize all these resources, you have built multiple storage hubs to house each resource you consider valuable. Presently, you are in the process of creating a series of underground tunnels to connect each of your storage centers. Each storage hub houses a single type of item.

While you enjoy spending your time constructing an elaborate series of tunnels, your friend very much enjoys skydiving, and repeatedly loses their inventory upon impact. Thus, they constantly have to build a new set of armor from scratch, taking advantage of your enviable stores of diamonds. Your friend continually interrupts your mining to request directions to a diamond hub, so you've decided to set up signs at each of the hubs from the starting home where your friend respawns to your nearest diamond hub to show your friend where to go.

To do so, you first need to know how many signs you need. You will set up signs at each hub (including home) until you reach the nearest diamond hub. It is not necessary to add a sign in the diamond hub.

The Problem:

Given the number of hubs and the tunnels between them, determine the number of signs required to guide your friend from home to the nearest diamond hub.

The Input:

The first line will contain two integers, n ($1 < n < 1{,}000$) and m ($0 < m < 1{,}000$), representing the number of hubs and the number of tunnels between the hubs. The following n lines will contain a string, r_i, representing the resource of hub i. The first string, r_1, will always have "home" as its name. Then, m lines follow, each containing two integers, i and j ($1 \le i \le n$; $1 \le j \le \text{n}$), denoting a tunnel between hub i and hub j. Note that tunnels are bidirectional, and there will only be a single tunnel between a given (i, j) pair (furthermore, there are no tunnels connecting a given hub i to itself). In addition, it is guaranteed that a diamond hub will be reachable. The names of resources will contain only lowercase English letters and will be of length 1 to 15, inclusive.

The Output:

Output a single integer: the number of signs you need to use to direct your friend from home to the nearest diamond hub.

(Sample Inputs and Sample Outputs appear on the following page)

Sample Input 1:

```
3 2
home
emerald
diamond
1 2
2 3
```

Sample Output 1:

```
2
```

Sample Input 2:

```
6 6
home
emerald
diamond
diamond
redstone
iron
1 2
2 5
2 6
5 6
6 3
6 4
```

Sample Output 2:

```
3
```

Traveling Sales Pigeon

Filename: `pigeon`

Tony, the Traveling Sales Pigeon, has a long day ahead of him delivering packages to different cities (each of which can be represented as points on the Cartesian plane). Since he's a pigeon and not particularly strong, he can only carry one package at a time. Therefore, Tony will have to fly back to headquarters (located at the origin) between each delivery to pick up a new package except for after the very last package as he plans to fly to a nearby park to relax. Since he doesn't want to be tired after work, he would like to know the minimum distance he will have to travel to deliver all the packages. Assume that Tony starts his day at the headquarters (located at 0,0).

The Problem:

Given the x and y coordinates of each city to which Tony has to deliver a package, determine the minimum distance Tony needs to travel while returning to headquarters to pick up a new package after every delivery except the last one assuming he delivers the packages in an optimal order.

The Input:

Input will begin with a line containing a single integer, p ($1 \le p \le 1{,}000$), representing the number of packages to be delivered that day. For the next p lines, each line will contain two integers, x and y ($-1{,}000 \le x \le 1{,}000$; $-1{,}000 \le y \le 1{,}000$) representing the (x, y) location of each city to which that package is being delivered. A package will never be delivered to headquarters.

The Output:

Output a single number representing the minimum distance that Tony must travel. Answers within an absolute or relative error of 10^{-6} of the answer will be considered correct.

Sample Input 1:

```
2
4 3
9 1
```

Sample Output 1:

```
19.0553851
```

Sample Input 2:

```
3
-4 -3
0 7
4 3
```

Sample Output 2:

```
27.0000000
```

Anything Under the Sky

Filename: `sky`

With banner flip-flapping,

once more you'll ride high!

Ready for anything under the sky.

Ready because you're that kind of a guy!

- From Dr. Seuss's *Oh, the Places You'll Go!*

There is just one problem: there are so many things under the sky that you don't even know how to begin counting them all! You decide that it would be easier to count the number of things that are not under the sky, and then subtract that number from the number of things in the countable universe.

You come up with a simple rule: an object is not under the sky if and only if the name of the object starts with the word "space."

Your mountain is waiting. So … get on your way!

The Problem:

Given a series of objects that make up the universe, determine how many of them are not under the sky.

The Input:

The first line of input contains an integer, n ($1 \leq n \leq 100$), representing the number of objects in the universe. The following n lines each contain a string, s (whose length is between 5 and 100, inclusive), representing the name of each object. Each string will consist only of lowercase English letters.

The Output:

Output a single integer: the number of objects in the universe that are not under the sky.

Sample Input 1:

```
9
spacesword
spaceoddity
spacebar
spock
spackle
aerospace
backspace
misplace
space
```

Sample Output 1:

```
4
```

Sample Input 2:

```
3
friends
family
spacedust
```

Sample Output 2:

```
1
```

Sample Input 3:

```
2
mountain
theamazingyou
```

Sample Output 3:

```
0
```

No Way Out But Through

Filename: `through`

Temra the warrior jolts awake in a modest cottage in the village of Almanor. She stares uncomprehendingly as the very ground seems to come alive with a fierce shaking. Windows rattle and shatter. Racing footsteps pound past her door. Screams rend the air. Almanor is under attack!

Temra fetches her most prized possession, a magical war hammer enchanted by her sister, and runs outside. The sight almost brings her to her knees: there are n armor-plated dire wolves wreaking havoc in the village square and lives are in danger!

The war hammer is a special artifact that imbibes power from Temra's desire to maintain peace in the village. The hammer's power level can be represented by a non-negative integer. Because the hammer is so volatile, Temra maintains it at power level 0 whenever it is not in use. Temra can increase the power level of her hammer in two ways:

- Pray to the Goddess of Bounty, who will multiply the hammer's power level by a positive integer, k.
- Pray to the Goddess of Hope, who will increase the hammer's power level by 1.

The goddesses are benevolent, and will answer every prayer that Temra makes. Temra wishes to make her hammer's power level equal to exactly n, so that she will have just enough blows to kill every wolf without bringing harm to the townspeople.

The Problem:

Given the number of wolves attacking the village square and the integer parameter, k, determine the minimum number of prayers Temra will need to make her war hammer’s power level exactly equal to the number of wolves.

The Input:

The input consists of a single line containing two integers, n ($1 \le n \le 10^{18}$) and k ($2 \le k \le 10$), representing the number of wolves in the square and the parameter k as described in the statement.

The Output:

Output a single integer: the minimum number of prayers Temra will need to make her war hammer’s power level exactly equal to the number of wolves.

(Sample Inputs and Sample Outputs appear on the following page)

Sample Input 1:

```
197 2
```

Sample Output 1:

```
11
```

Sample Input 2:

```
1000000000000000000 10
```

Sample Output 2:

```
19
```

Sample Input 3:

```
6 3
```

Sample Output 3:

```
3
```

The *n* Days of Christmas

Filename: `xmas`

You want to be productive and finish that project you've been procrastinating on for weeks. You and your friend Cameron decide to meet and work on it together. Unfortunately, Cameron sings the "Twelve Days of Christmas" song when you try working. This wouldn't be so bad, but sometimes Cameron forgets the lyric corresponding to a certain day and has to sing the whole song again. Not only that but he's decided to make up even more days of Christmas because he loves the song so much. The i^{th} "day" takes *i* seconds for Cameron to sing. If he forgets a day of Christmas, he pauses 1 second and then restarts the whole song again. He will only forget that day the first time he reaches it.

> From Wikipedia, "The Twelve Days of Christmas" is a cumulative song, meaning that each verse is built on top of the previous verses. The first three verses run, in full, as follows:
>
> *On the first day of Christmas my true love sent to me*
> *A partridge in a pear tree*
>
> *On the second day of Christmas my true love sent to me*
> *Two turtle doves,*
> *And a partridge in a pear tree.*
>
> *On the third day of Christmas my true love sent to me*
> *Three French hens,*
> *Two turtle doves,*
> *And a partridge in a pear tree.*
>
> Subsequent verses follow the same pattern. Each verse deals with the next day of Christmastide, adding one new gift and then repeating all the earlier gifts, so that each verse is one line longer than its predecessor.

Let's say that Cameron wants to sing all twelve days but will forget days 4, 7, and 9. He will sing the song normally through the 3rd verse. He will pause for one second, realizing that he forgot what the 4th gift of Christmas is, and then will sing the song from the very beginning, starting with the 1st day of Christmas. He will continue singing normally through the 6th day, again realizing he has forgotten what the 7th gift of Christmas is, pause for one second, and then start from the very beginning again. He will sing the song normally through the 8th day, realize he has forgotten what the 9th gift is, pause for one second, and start the song from the beginning. This time, he will sing the full "Twelve Days of Christmas" song normally.

The Problem:

Given the number of days of Christmas in Cameron's version of the song, and a list of the days he forgets, output how many seconds it will take for Cameron to finish singing the song.

The Input:

The first line of the input contains two integers, n and k ($1 \leq k \leq n \leq 10{,}000$), representing the number of days of Christmas and the number of days that Cameron forgets, respectively. On the next line, there will be k distinct integers in sorted order where k_i is the i^{th} day of Christmas that Cameron initially forgets.

The Output:

Output the amount of time, in seconds, it will take Cameron to finish singing the song.

Sample Input 1:

```
12 3
4 7 9
```

Sample Output 1:

```
144
```

Sample Input 2:

```
5 1
3
```

Sample Output 2:

```
19
```