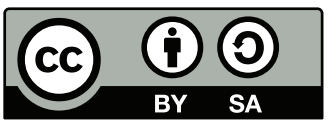
Freshmen Programming Contest

Contest Problem Set

June 11, 2020



Problem A
Alien Journey
Time limit: 5 seconds

UFO spying on Delft.
(source: www.syfy.com/syfywire/defense-department-declassifies-ufo-footage)

While wandering around Delft, after a long time in isolation, convinced that nothing could surprise you anymore (after the roller-coaster this year has already been), you encounter yet another peculiarity. From behind a bush, a tiny, friendly-looking alien, reveals himself. Rather confused, he kindly asks you for some help with his new, square shaped spaceship. After bragging about the spaceship's brand new features, the little alien expresses his worries about the costs of traveling nowadays: "Extraterrestrial economy is down as well, my dear human being!" he protested. "I need to make it to the interdimensional gateway which is across the town but I have scarcely any fuel left in my tank. Could you help me get back home?" Knowing that you are a master of algorithms, he pleads you to compute the minimum amount of power he would need to traverse the city, given that each unit of fuel can lift the spaceship one unit in height.

In order for the spaceship to travel above an area of the city, it should fly overhead all the cells underneath it while going in either of the 4 directions: North, South, East and West.

You also find out that moving the spaceship in any of the cardinal directions does not consume any fuel and that the entirety of the spaceship should be within the map at any given moment (otherwise the alien gets disoriented).

Animated by the desire to help your newly found friend, you get to coding in a heartbeat. Guided by the adventurous computer scientist that you are, you can quickly assess that both the map and the spaceship should be seen as square grids and that each cell of the map has a height (computed relatively to the sea level).

You make the further observations:

- Initially, the spaceship lies on the ground.
- For the spaceship to be above the ground, the bottom of the spaceship should be strictly higher than the height of all the ground cells beneath.
- All 4 edges of the square spaceship are aligned with the grid-like map of the city.
- The top-left corner cell of the spaceship overlays the top-left corner of the map.
- The area of the map the spaceship initially lies on is guaranteed to be at height 0.
- The ship can only be lifted an integer number of units in height.
- The final destination of the spaceship is reached when the bottom-right corner cell of the spaceship is floating above the bottom-right corner cell of the map.

Input

- One line with three integers: $1 \leq h \leq 500$ and $1 \leq w \leq 500$: the height and width of the city map and $1 \leq l < \min(h, w)$ the length of the spaceship edges.
- h lines, each with w positive integers in the range $[0, 10^9]$. Each of these integers describes the height above the ground of a 1×1 section of the map.

Output

The minimum amount of power units that the alien needs for traversing the city with the spaceship.

Sample Input 1

```
5 5 2
0 0 1 1 3
0 0 1 1 3
3 3 1 1 3
3 3 1 1 1
3 3 1 1 1
```

Sample Output 1

```
2
```

Sample Input 2

```
6 6 1
0 1 2 3 4 5
0 1 2 3 4 5
0 1 1 1 1 5
0 3 4 5 6 7
0 3 0 0 0 7
0 0 0 1 0 0
```

Sample Output 2

```
1
```

Sample Input 3

```
3 3 2
0 0 7
0 0 7
9 7 9
```

Sample Output 3

```
10
```

Problem B
Banitsa
Time limit: 3 seconds

The Yellow Duckling's favourite dish is banitsa (a traditional Bulgarian dish that is similar to a cheese pie)! That is why his mother made him one last week and since he is really greedy, he ate almost all of it and there were only a few pieces left. The Duckling chose to eat them for breakfast and thus went to buy his favourite yoghurt to put as a topping in the morning. Then he put the pieces in a circle, enumerated them in a clock-wise direction and just before putting a spoon of yoghurt on the first one, his mother interrupted him and said that his father should also try the banitsa (and his father really hates yoghurt for some reason?!). Thus, she indicated some pairs of pieces that were supposed to have different toppings. The interesting part of those pairs was that if you connected all pairs of pieces with a line, there were not any two lines that were ever crossing.

Since the Yellow Duckling is really concerned about the family budget (which isn't really high after the COVID-19 crisis anyway...), he wants to show empathy for his parents and to buy as few toppings for the banitsa as possible with which he would be able to satisfy his mom's conditions. But it is still only 3 years old and does not find algorithms such an interesting topic as we do (let's hope it understands how cool they are when it grows up) and asks you to help him with this task.

Input

- One line with two integers: $2 \leq n \leq 10^6$, the pieces of banitsa left. $2 \leq m \leq 10^5$, the number of pairs.
- m lines (one for each pair of pieces), containing the indices of the pieces which should be topped with different toppings.

Output

The minimum number of different toppings the Yellow Duckling needs to buy.

Sample Input 1	Sample Output 1
6 5 1 2 2 3 1 4 3 4 6 5	2

Sample Input 2	Sample Output 2
5 3 1 2 3 2 1 3	3

This page is intentionally left blank.

Problem C
Chill and Netflix
Time limit: 2 seconds

(source: https://www.theverge.com/2020/2/21/21136537/tivo-skip-button-tv-dvr-fast-forward)

You are in quarantine and there is no way you could step outside. It would be insanely dangerous. On the other hand, you have already exhaustively studied all the course materials. It seems then that you might be in danger of getting bored. Suddenly, you realize there is something that could save you from the disaster that boredom entails. It's streaming Netflix that could provide some entertainment! However, you already know that you would rather skip some parts of the movies or tv series you plan on watching. Therefore, to be prepared for your binge watching sessions, you configure a set of skip buttons that could skip a number of seconds each. Each time you play a movie, you start clicking on your self configured skip buttons that you are so proud of and often even forget to let the movie play. While excitedly clicking on your buttons, which definitely put your boredom to rest, you ask yourself: How many distinct moments of the movie am I able to reach by continuously clicking my beloved buttons, without letting the movie play for any second, of course.

Input

The input consists of two lines:

- One line with two integers: $1 \leq n \leq 10^9$, the length of the movie in seconds, and $1 \leq m \leq 10^3$, the number skip buttons.
- One line with m unique integers $1 \leq i \leq 10^3$, representing the number of seconds each button can skip.

You can assume that the time it takes to click on a button is negligible. There is no idle time between two consecutive button clicks, and you start clicking on buttons at beginning of second 1. All the buttons can be clicked any number of times, and you click on at least one button.

Output

The number of distinct moments (the start of seconds of the movie) that can be reached after an exciting and continuous sequence of skip button clicks (of length > 0). In other words, you never reach to see the first second of the movie, as it would be excessively boring.

Sample Input 1	Sample Output 1
10 2 2 3	8

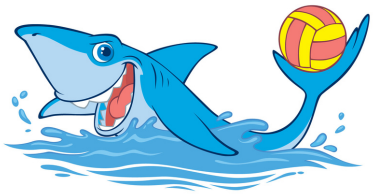
Sample Input 2	Sample Output 2
10 1 1	9

Sample Input 3	Sample Output 3
20 3 5 7 11	11

Problem D
Ducks and Sharks
Time limit: 2 seconds

The Ducks and the Sharks have had many adventures in the past year. Because the weather has been so nice lately, they want to organise a water polo tournament. After forming the teams, every team plays one match against every other team. Being caught up in the fun of playing the matches, they don't feel like figuring out who won, so they turn to you for help.

You receive a list of matches and the number of goals made by both teams in a match. For each match, the winner gets 3 points, the loser gets 0 points, and in case of a draw both teams get 1 point. It is your job to figure out which teams got the most points, and provide a top five of the best teams.



A shark playing water polo.
Credit: Kristijan Hranisavljevic

Input

- One line with one integer: $2 \leq n \leq 400$, the number of teams.
- $\frac{n \cdot (n-1)}{2}$ lines (one for each match), containing the names of the two teams and the number of goals both teams made during that match (all separated by one space).
  - The name of each team consists of at most 20 characters from the English alphabet (`A-Z` and `a-z`).
  - The scores are between 0 and 100 (inclusive).

Note that every team plays exactly once against every other team.

Output

The names of the *top five teams* in the water polo competition, followed by their total obtained score. Each team name should be written on a separate line. Each team name and score should be separated by exactly one space. Note:

- If there are less than five teams participating, simply list all participants (see example 1).
- If there are two teams with the same score, order them alphabetically (see example 2).

Sample Input 1	Sample Output 1
4 Splashers Sharks 4 2 Ducks Fins 5 2 Fins Splashers 3 5 Ducks Sharks 0 5 Splashers Ducks 8 1 Fins Sharks 1 1	Splashers 9 Sharks 4 Ducks 3 Fins 1

Sample Input 2	Sample Output 2
5 A B 5 3 C D 2 9 A E 0 0 C B 1 1 A C 2 5 B E 2 5 D A 1 6 C E 4 2 B D 0 2 E D 5 2	A 7 C 7 E 7 D 6 B 1

Problem E
Excursion
Time limit: 7 seconds

Jimmy goes on an excursion in the country of Treenidad and Treebago. People there are obsessed with trees so much, they modeled their country after them. Being a careful planner, Jimmy wants to know in advance which cities should be visited to maximize the total appeal of his excursion. The appeal of a city is defined by a not necessarily positive integer. Since he went through a lot of hassle to get his visa, he wants to visit at least one city. Jimmy's excursion can start from any city. His only requirement when visiting the country is that he mustn't visit the same city twice.

Input

The first line in the input contains an integer $1 \leq n \leq 10^6$, the number of cities in Treenidad and Treebago.
After that, n lines follow, the first of which describes the root of the tree.
Each of the following lines contains two integers, V and C, which describe the properties of a node in the tree:

- V represents the value at that node, with $-2^{31} \leq V < 2^{31}$.
- C represents the number of children of the node, with $0 \leq C < 10^6$.

After that, C lines follow, each recursively defining the child trees. It is guaranteed that the height of the tree is less than or equal to 990.

Output

The maximum appeal Jimmy can gather in his excursion.

(Examples on the next page.)

Examples

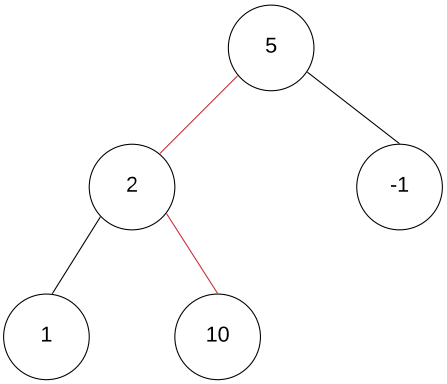


Figure E.1: Illustration of the first sample test case

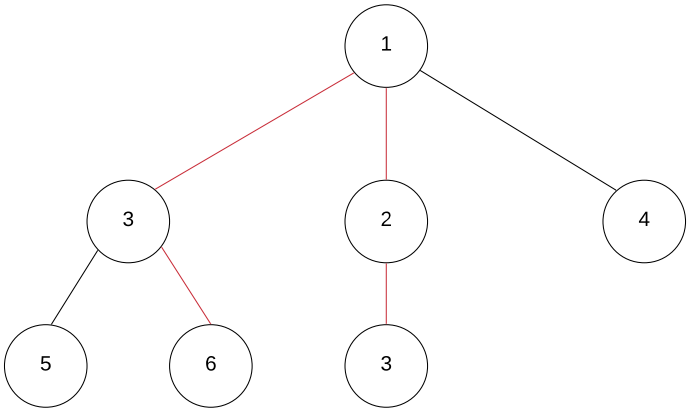


Figure E.2: Illustration of the second sample test case

Sample Input 1	Sample Output 1
5 5 2 2 2 1 0 10 0 -1 0	17

Sample Input 2	Sample Output 2
7 1 3 3 2 5 0 6 0 2 1 3 0 4 0	15

Problem F
Family Tree
Time limit: 3 seconds

While you are attending the yearly family gathering of your family, you notice that your family keeps growing bigger and bigger. You are having a hard time remembering for every member of the family what their name is and to what part of the family they belong to. To solve this, you decide to create a family tree on a big sheet of paper. You collect a portrait picture from every member of the family and stick them onto the paper in the shape of a tree, putting pictures of children exactly one level below the picture of their parents.

Before actually sticking the pictures on the paper, you need to figure out how much paper you need. Environmentally aware as you are, you try to minimize the amount of paper needed. You decide to allow pictures within the same level to move to the left or right if this makes the tree less wide (see also the examples below).

A family of trees. (Source: unknown)

Input

- One line with one integer: $2 \leq n \leq 10^5$, the number of people in the family tree.
- One line that indicates the earliest ancestor that you have information about.
- $n-1$ lines, each in the format "`A - B`", indicating that `A` is the parent of `B`. These lines have no particular order.

The name of each person consists of at most 20 characters from the English alphabet (`A-Z` and `a-z`).

Output

The minimum width of the family tree in the number of portrait pictures.

(Examples on the next page.)

Examples

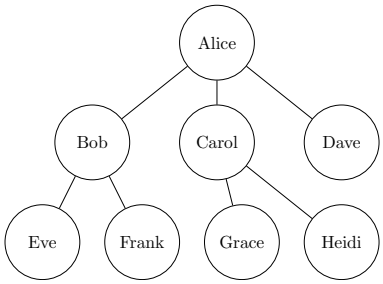


Figure F.1: The family tree belonging to the first example, with a maximum width of 4 on the bottom row.

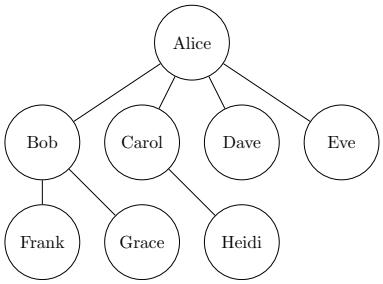


Figure F.2: The family tree belonging to the second example, with a maximum width of 4 on the middle row.

Sample Input 1

```
8
Alice
Alice - Bob
Alice - Carol
Alice - Dave
Bob - Eve
Bob - Frank
Carol - Grace
Carol - Heidi
```

Sample Output 1

```
4
```

Sample Input 2

```
8
Alice
Alice - Bob
Alice - Carol
Alice - Dave
Alice - Eve
Bob - Frank
Bob - Grace
Carol - Heidi
```

Sample Output 2

```
4
```

Problem G
Group Activities
Time limit: 1 second

The lockdown is still in place, so organizing the yearly group excursion is a no-go. Instead, the study association is organizing the Fictional Programmer Convention (FPC): a digital event where programmers can meet other programmers while performing various digital activities. You are on the committee that is concerned with planning the events of the FPC. They give you the following task: determine how many programmers you want to invite.

The committee will plan different activities that will take place for all participants at the same time, with the participants divided into groups. Each activity requires an exact number of participants per group. For example, a variety of digital games are played in groups of 5, a pop quiz about programming history is played in teams of 3, and a workshop on clean coding practices is given in separate sessions of 31.

Because of a limited budget, you want to invite as few people as possible, but still at least one person of course. Can you determine the exact number of people you must invite to exactly fill each activity?

Input

- One line containing one integer $1 \leq n \leq 10^5$, the number of activities organized.
- For each of the n activities, one line with an integer $1 \leq i \leq 10^{18}$: this activity requires exactly i participants per group.

Output

An integer indicating the smallest number of people p you can invite, such that each activity can exactly fill a number of groups. It is guaranteed that $1 \leq p \leq 10^{18}$.

Sample Input 1	Sample Output 1
3 5 3 31	465

Sample Input 2	Sample Output 2
4 15 8 7 11	9240

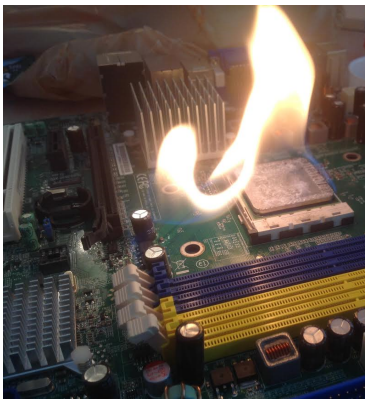
This page is intentionally left blank.

Problem H
Halt and Catch Fire
Time limit: 1 second

Research has led to the creating of a new super fast CPU called the Fast Processing Chip (FPC). This product has a never-before seen debugging function: an instruction called `hcf`, providing debugging power unmatched by any previously designed CPU.

With the product becoming popular fast, companies can't wait to get their hands on the chip to start development immediately. One problem, however, is the chip is too expensive to just hand over to every developer all willy-nilly.

You have been hired to design a program that runs as if it were a processor, so the programmers can write programs and easily test these programs without needing the hardware.



The FPC, after processing the `hcf` instruction.
(Source: https://imgur.com/gallery/zDbw6en)

Input

- One line with one integer: $1 \leq n \leq 1000$, the number of lines in the program.
- n lines with three sections, separated with spaces:
  - The operation, always 3 characters
  - An argument: Either an integer $-2^{31} \leq a \leq 2^{31} - 1$ or a variable prepended with `$`
  - An argument: Either an integer $-2^{31} \leq b \leq 2^{31} - 1$ or a variable prepended with `$`

There are five possible operations:

- `mov` – Move value from the first argument to the second
- `add` – Add value from the second argument to the first and store in `$acc`
- `sub` – Subtract value of the second argument from the first and store in `$acc`
- `jeq` – Jump to the instruction on the value of the second argument if the value of the first argument equals `$cmp`
- `hcf` – Halt and Catch Fire: Stop the program immediately, takes only integer arguments

An argument can be noted as an integer immediate or a variable reference, in which case it's prepended by `$`. All variable names consist of at most 20 lowercase characters from the English alphabet (`a-z`) or underscores (_).

There are four predefined variables:

- `$acc` – Contains the value of an `add` or `sub` operation
- `$pc` – Contains the address of the currently processed operation
- `$cmp` – Used to compare with the `jeq` operation

- `$out` – Should be output when the program finishes

Any other variables are defined as soon as a value is moved to them.
Assume any given input program to be syntactically correct.
Integers will never over- or underflow.

Output

The program exits either when the value of the `$pc` variable reaches out of the bounds of the program, or when an `hcf` instruction is reached.

- In the first case, output one line containing the value of `$out` at the end of the program.
- In the second case, output the following:
  - One line containing the `hcf` instruction that was called.
  - One line containing the content of the `$acc` variable.
  - One line containing the content of the `$cmp` variable.
  - One line containing the content of the `$out` variable.

Sample Input 1	Sample Output 1

`6`
`mov 3 $cmp`
`jeq 3 3`
`jeq $cmp 5`
`mov 42 $out`
`jeq $cmp 7`
`mov 50 $out`
	`42`

Sample Input 2	Sample Output 2

`4`
`mov 3 $cmp`
`mov 2 $out`
`add $cmp $out`
`hcf 0 0`
	`hcf 0 0`
	`5`
	`3`
	`2`

Problem I
Integrity Overflow
Time limit: 1 second

The Fortune Protection Corporation (FPC) is a company specialized in securely storing the money of their clients. The money safe is sealed with a 50 cm thick steel door that can only open after correctly entering the secret password in a terminal next to the door. The FPC takes the integrity of this system very serious, and wants to be sure that the door never opens when somebody enters an incorrect password. Since some of the employees of the FPC have dyslexia, the safe will also open when the password contains one incorrect character (it must have the correct length, though).

Given an audit log containing all access attempts to the safe, can you tell whether the integrity of the safe is compromised?

Input

- One line with one integer: $1 \leq n \leq 1000$, the number of lines in the audit log.
- One line containing the correct password.
- n lines with two words each, separated with a single space:
  - The first word is the entered password.
  - The second word is either "`ALLOWED`" or "`DENIED`", depending on whether the password was correct or incorrect, respectively.

All passwords are at most 32 characters in length and only contain letters from the English alphabet (`A-Z` and `a-z`) and numbers (`0-9`).

Output

One line, containing "`SYSTEM SECURE`" if the audit log contains no suspicious access attempts, or "`INTEGRITY OVERFLOW`" if the security of the system has been compromised.

Sample Input 1

```
4
SuperSecret123
SuperSecret123 ALLOWED
SuperSekret123 ALLOWED
Hunter02 DENIED
password123 DENIED
```

Sample Output 1

```
SYSTEM SECURE
```

Sample Input 2

```
4
CorrectHorseBatteryStaple
CorrectHorseBatteryStaple ALLOWED
HorseCorrectStapleBattery ALLOWED
CorrectHorseBatteryStaple ALLOWED
123456789 DENIED
```

Sample Output 2

```
INTEGRITY OVERFLOW
```

This page is intentionally left blank.