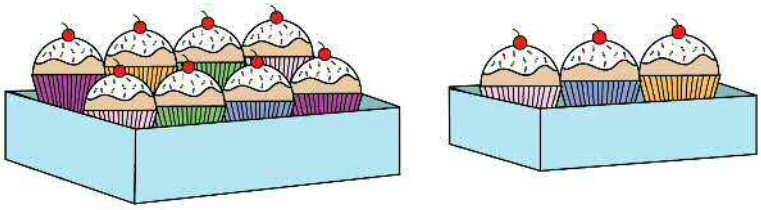
Problem J1: Cupcake Party

Problem Description

A regular box of cupcakes holds 8 cupcakes, while a small box holds 3 cupcakes. There are 28 students in a class and a total of at least 28 cupcakes. Your job is to determine how many cupcakes will be left over if each student gets one cupcake.



Input Specification

The input consists of two lines.

- The first line contains an integer $R \geq 0$, representing the number of regular boxes.
- The second line contains an integer $S \geq 0$, representing the number of small boxes.

Output Specification

Output the number of cupcakes that are left over.

Sample Input 1

```
2
5
```

Output for Sample Input 1

```
3
```

Explanation of Output for Sample Input 1

The total number of cupcakes is $2 \times 8 + 5 \times 3$ which equals 31. Since there are 28 students, there are 3 cupcakes left over.

Sample Input 2

```
2
4
```

Output for Sample Input 2

```
0
```

Explanation of Output for Sample Input 2

The total number of cupcakes is $2 \times 8 + 4 \times 3$ which equals 28. Since there are 28 students, there are no cupcakes left over.

La version française figure à la suite de la version anglaise.

Problem J2: Fergusonball Ratings

Problem Description
Fergusonball players are given a star rating based on the number of points that they score and the number of fouls that they commit. Specifically, they are awarded 5 stars for each point scored, and 3 stars are taken away for each foul committed. For every player, the number of points that they score is greater than the number of fouls that they commit.

Your job is to determine how many players on a team have a star rating greater than 40. You also need to determine if the team is considered a gold team which means that *all* the players have a star rating greater than 40.

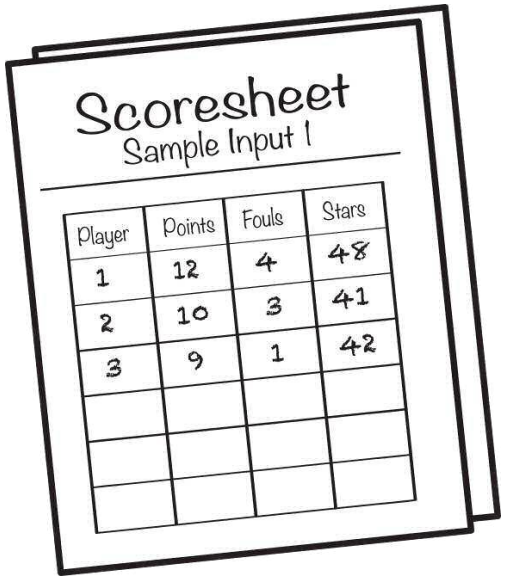
Input Specification
The first line of input consists of a positive integer N representing the total number of players on the team. This is followed by a pair of consecutive lines for each player. The first line in a pair is the number of points that the player scored. The second line in a pair is the number of fouls that the player committed. Both the number of points and the number of fouls, are non-negative integers.

Output Specification
Output the number of players that have a star rating greater than 40, immediately followed by a plus sign if the team is considered a gold team.

Sample Input 1
```
3
12
4
10
3
9
1
```

Output for Sample Input 1
```
3+
```

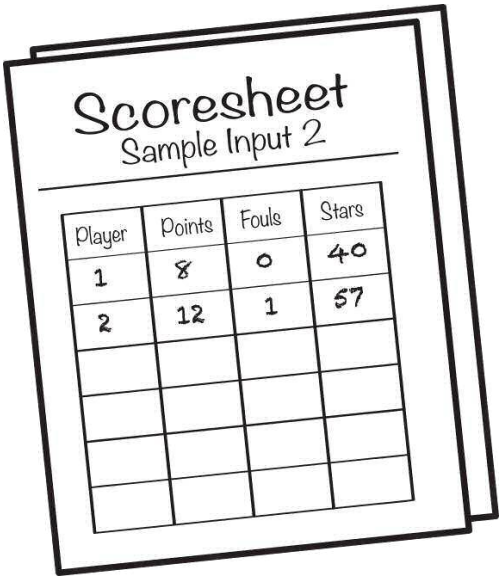
Explanation of Output for Sample Input 1
The image shows the star rating for each player. For example, the star rating for the first player is $12 \times 5 - 4 \times 3 = 48$. All three players have a rating greater than 40 so the team is considered a gold team.



La version française figure à la suite de la version anglaise.

Sample Input 2

```
2
8
0
12
1
```



Output for Sample Input 2

```
1
```

Explanation of Output for Sample Input 2

The image shows the star rating for each player. Since only one of the two players has a rating greater than 40, this team is not considered a gold team.

La version française figure à la suite de la version anglaise.

Problem J3: Harp Tuning

Problem Description

The CCC harp is a stringed instrument with strings labelled $A, B, \ldots, T$. Like other instruments, it can be out of tune.

A musically inclined computer science student has written a clever computer program to help tune the harp. The program analyzes the sounds produced by the harp and provides instructions to fix each string that is out of tune. Each instruction includes a group of strings, whether they should be tightened or loosened, and by how many turns.

Unfortunately, the output of the program is not very user friendly. It outputs all the tuning instructions on a single line. For example, the single line `AFB+8HC-4` actually contains two tuning instructions: `AFB+8` and `HC-4`. The first instruction indicates that harp strings A, F, and B should be tightened 8 turns, and the second instruction indicates that harp strings H and C should be loosened 4 turns.

Your job is to take a single line of tuning instructions and make them easier to read.

Input Specification

There will be one line of input which is a sequence of tuning instructions. Each tuning instruction will be a sequence of uppercase letters, followed by a plus sign (`+`) or minus sign (`-`), followed by a positive integer. There will be at least one instruction and at least one letter per instruction. Also, each uppercase letter will appear at most once.

The following table shows how the available 15 marks are distributed.

Marks Awarded	Maximum Input Values			Example Input
	Number of Instructions	Number of Letters in an Instruction	Number of Turns	
5 marks	1	20	9	`AFB+8`
5 marks	20	1	9	`A+8H-4`
3 marks	20	20	9	`AFB+8HC-4`
2 marks	20	20	999 999	`AFB+88HC-444`

Output Specification

There will be one line of output for each tuning instruction. Each line of output will consist of three parts, each separated by a single space: the uppercase letters referring to the strings, `tighten` if the instruction contained a plus sign or `loosen` if the instruction contained a minus sign, and the number of turns.

La version française figure à la suite de la version anglaise.

Sample Input 1
```
AFB+8HC-4
```

Output for Sample Input 1
```
AFB tighten 8
HC loosen 4
```

Explanation of Sample Output 1
The input contains two tuning instructions: `AFB+8` and `HC-4`.

Sample Input 2
```
AFB+8SC-4H-2GDPE+9
```

Output for Sample Input 2
```
AFB tighten 8
SC loosen 4
H loosen 2
GDPE tighten 9
```

Explanation of Sample Output 2
The input contains four tuning instructions: `AFB+8`, `SC-4`, `H-2`, and `GDPE+9`.

La version française figure à la suite de la version anglaise.

Problem J4/S2: Good Groups

Problem Description
A class has been divided into groups of three. This division into groups might violate two types of constraints: some students must work together in the same group, and some students must work in separate groups.

Your job is to determine how many of the constraints are violated.

Input Specification
The first line will contain an integer X with $X \geq 0$. The next X lines will each consist of two different names, separated by a single space. These two students *must* be in the same group.

The next line will contain an integer Y with $Y \geq 0$. The next Y lines will each consist of two different names, separated by a single space. These two students *must not* be in the same group.

Among these $X + Y$ lines representing constraints, each possible pair of students appears at most once.

The next line will contain an integer G with $G \geq 1$. The last G lines will each consist of three different names, separated by single spaces. These three students have been placed in the same group.

Each name will consist of between 1 and 10 uppercase letters. No two students will have the same name and each name appearing in a constraint will appear in exactly one of the G groups.

The following table shows how the available 15 marks are distributed at the Junior level.

Marks Awarded	Number of Groups	Number of Constraints
4 marks	$G \leq 50$	$X \leq 50$ and $Y = 0$
10 marks	$G \leq 50$	$X \leq 50$ and $Y \leq 50$
1 mark	$G \leq 100\,000$	$X \leq 100\,000$ and $Y \leq 100\,000$

The following table shows how the available 15 marks are distributed at the Senior level.

Marks Awarded	Number of Groups	Number of Constraints
3 marks	$G \leq 50$	$X \leq 50$ and $Y = 0$
5 marks	$G \leq 50$	$X \leq 50$ and $Y \leq 50$
7 marks	$G \leq 100\,000$	$X \leq 100\,000$ and $Y \leq 100\,000$

Output Specification
Output an integer between 0 and $X + Y$ which is the number of constraints that are violated.

La version française figure à la suite de la version anglaise.

Sample Input 1

```
1
ELODIE CHI
0
2
DWAYNE BEN ANJALI
CHI FRANCOIS ELODIE
```

Output for Sample Input 1

```
0
```

Explanation of Output for Sample Input 1

There is only one constraint and it is not violated: ELODIE and CHI are in the same group.

Sample Input 2

```
3
A B
G L
J K
2
D F
D G
4
A C G
B D F
E H I
J K L
```

Output for Sample Input 2

```
3
```

Explanation of Output for Sample Input 2

The first constraint is that A and B must be in the same group. This is violated.

The second constraint is that G and L must be in the same group. This is violated.

The third constraint is that J and K must be in the same group. This is *not* violated.

The fourth constraint is that D and F must not be in the same group. This is violated.

The fifth constraint is that D and G must not be in the same group. This is *not* violated.

Of the five constraints, three are violated.

La version française figure à la suite de la version anglaise.

Problem J5: Square Pool

Problem Description

Ron wants to build a square pool in his square N-by-N yard, but his yard contains T trees. Your job is to determine the side length of the largest square pool he can build.

Input Specification

The first line of input will be an integer N with $N \geq 2$. The second line will be the positive integer T where $T < N^2$. The remaining input will be T lines, each representing the location of a single tree. The location is given by two positive integers, R and then C, separated by a single space. Each tree is located at row R and column C where rows are numbered from top to bottom from 1 to N and columns are numbered from left to right from 1 to N. No two trees are at the same location.

The following table shows how the available 15 marks are distributed.

Marks Awarded	Length/Width of Yard	Number of Trees
3 marks	$N \leq 50$	$T = 1$
5 marks	$N \leq 50$	$T \leq 10$
4 marks	$N \leq 500\,000$	$T \leq 10$
3 marks	$N \leq 500\,000$	$T \leq 100$

Output Specification

Output one line containing M which is the largest positive integer such that some M-by-M square contained entirely in Ron's yard does not contain any of the T trees.

Sample Input 1

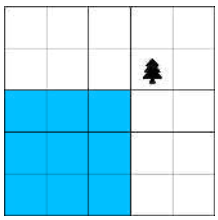
```
5
1
2 4
```

Output for Sample Input 1

3

Explanation of Output for Sample Input 1

A picture of the yard is below. The location of the tree is marked by 🌲 and one of several 3-by-3 squares that do not contain the tree is highlighted. All larger squares contain a tree.



La version française figure à la suite de la version anglaise.

Sample Input 2

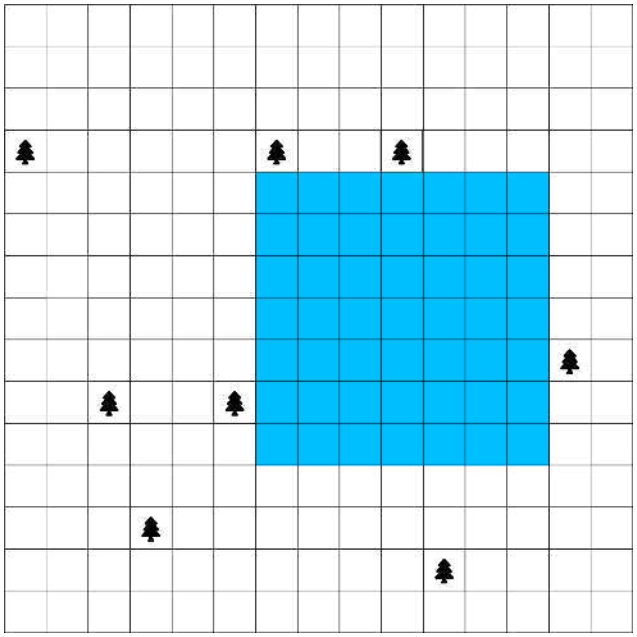
```
15
8
4 7
4 1
14 11
10 6
13 4
4 10
10 3
9 14
```

Output for Sample Input 2

7

Explanation of Output for Sample Input 2

A picture of the yard is below. The location of each tree is marked by 🌲 and one of several 7-by-7 squares that do not contain a tree is highlighted. All larger squares contain a tree.



La version française figure à la suite de la version anglaise.

Problem S1: Good Fours and Good Fives

Problem Description

Finn loves Fours and Fives. In fact, he loves them so much that he wants to know the number of ways a number can be formed by using a sum of fours and fives, where the order of the fours and fives does not matter. If Finn wants to form the number 14, there is one way to do this which is $14 = 4 + 5 + 5$. As another example, if Finn wants to form the number 20, this can be done two ways, which are $20 = 4+4+4+4+4$ and $20 = 5+5+5+5$. As a final example, Finn can form the number 40 in three ways: $40 = 4+4+4+4+4+4+4+4+4+4$, $40 = 4 + 4 + 4 + 4 + 4 + 5 + 5 + 5 + 5$, and $40 = 5 + 5 + 5 + 5 + 5 + 5 + 5 + 5$.

Your task is to help Finn determine the number of ways that a number can be written as a sum of fours and fives.

Input Specification

The input consists of one line containing a number N.

The following table shows how the available 15 marks are distributed.

Marks Awarded	Bounds on N	Additional Constraints
3 marks	$1 \leq N \leq 10$	None
2 marks	$1 \leq N \leq 100\ 000$	N is a multiple of 4
2 marks	$1 \leq N \leq 100\ 000$	N is a multiple of 5
8 marks	$1 \leq N \leq 1\ 000\ 000$	None

Output Specification

Output the number of unordered sums of fours and fives which form the number N. Output `0` if there are no such sums of fours and fives.

Sample Input 1

```
14
```

Output for Sample Input 1

```
1
```

Explanation of Output for Sample Input 1

This is one of the examples in the problem description.

Sample Input 2

```
40
```

Output for Sample Input 2

```
3
```

La version française figure à la suite de la version anglaise.

Explanation of Output for Sample Input 2
This is one of the examples in the problem description.

Sample Input 3
```
6
```

Output for Sample Input 3
```
0
```

Explanation of Output for Sample Input 3
There is no way to use a sum of fours and fives to get 6.

La version française figure à la suite de la version anglaise.

Problem S3: Good Samples

Problem Description
You are composing music for the Cool Clarinet Competition (CCC). You have been instructed to make a piece of music with exactly N notes. A note is represented as a positive integer, indicating the pitch of the note.

We call a non-empty sequence of consecutive notes in the piece a *sample*. For instance, $(3, 4, 2)$, $(1, 2, 3, 4, 2)$ and (4) are samples of $1, 2, 3, 4, 2$. Note that $(1, 3)$ is not a sample of $1, 2, 3, 4, 2$. We call two samples different if they start or end at a different position in the piece.

We call a sample *good* if no two notes in the sample have the same pitch.

The clarinet players are picky in two ways. First, they will not play any note with pitch higher than M. Second, they want a piece with exactly K good samples.

Can you construct a piece to satisfy the clarinet players?

Input Specification
The first and only line of input will contain 3 space-separated integers, N, M and K.

The following table shows how the available 15 marks are distributed.

Marks Awarded	**Bounds on N**	**Bounds on M**	**Bounds on K**
3 marks	$1 \leq N \leq 16$	$M = 2$	$1 \leq K \leq 1\,000$
3 marks	$1 \leq N \leq 10^6$	$M = 2$	$1 \leq K \leq 10^{18}$
4 marks	$1 \leq N \leq 10^6$	$M = N$	$1 \leq K \leq 10^{18}$
5 marks	$1 \leq N \leq 10^6$	$1 \leq M \leq N$	$1 \leq K \leq 10^{18}$

Output Specification
If there is a piece of music that satisfies the given constraints, output N integers between 1 and M, representing the pitches of the notes of the piece of music. If there is more than one such piece of music, any such piece of music may be outputted.

Otherwise, output -1.

Sample Input 1
```
3 2 5
```

Sample Output 1
```
1 2 1
```

Explanation of Output for Sample Input 1
Notice that the piece is composed of $N = 3$ notes, each of which is one of $M = 2$ possible

La version française figure à la suite de la version anglaise.

pitches, 1 and 2. That piece of music has a total of 6 samples, but only $K = 5$ good samples: (1), (1, 2), (2), (2, 1), (1). Notice that the two good samples of (1) are different since they start at two different positions.

Note that the piece `2 1 2` is the only other valid output for this input.

One example of an output that would be incorrect is `3 2 3`, since it has notes with pitches larger than 2. Another incorrect output would be `1 1 2`, since it only has four good samples: (1), (1), (2) and (1, 2).

Sample Input 2
```
5 5 14
```

Sample Output 2
```
1 5 3 2 1
```

Explanation of Output for Sample Input 2
The 14 good samples are: (1), (1, 5), (1, 5, 3), (1, 5, 3, 2), (5), (5, 3), (5, 3, 2), (5, 3, 2, 1), (3), (3, 2), (3, 2, 1), (2), (2, 1), (1).

Sample Input 3
```
5 5 50
```

Sample Output 3
```
-1
```

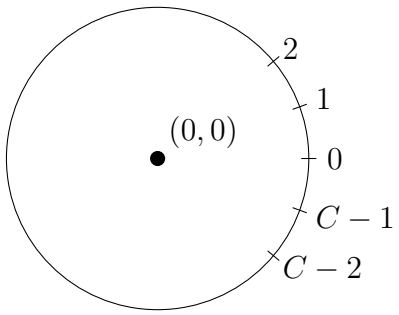
Explanation of Output for Sample Input 3
There are no pieces with 5 notes that can produce 50 different good samples.

La version française figure à la suite de la version anglaise.

Problem S4: Good Triplets

Problem Description

Andrew is a very curious student who drew a circle with the center at $(0, 0)$ and an integer circumference of $C \geq 3$. The location of a point on the circle is the counter-clockwise arc length from the right-most point of the circle.



Andrew drew $N \geq 3$ points at integer locations. In particular, the i^{th} point is drawn at location P_i $(0 \leq P_i \leq C - 1)$. It is possible for Andrew to draw multiple points at the same location.

A good triplet is defined as a triplet (a, b, c) that satisfies the following conditions:

- $1 \leq a < b < c \leq N$.
- The origin $(0, 0)$ lies strictly inside the triangle with vertices at P_a, P_b, and P_c. In particular, the origin is **not** on the triangle's perimeter.

Lastly, two triplets (a, b, c) and (a', b', c') are distinct if $a \neq a'$, $b \neq b'$, or $c \neq c'$.

Andrew, being a curious student, wants to know the number of distinct good triplets. Please help him determine this number.

Input Specification

The first line contains the integers N and C, separated by one space.

The second line contains N space-separated integers. The i^{th} integer is P_i $(0 \leq P_i \leq C - 1)$.

The following table shows how the available 15 marks are distributed.

La version française figure à la suite de la version anglaise.

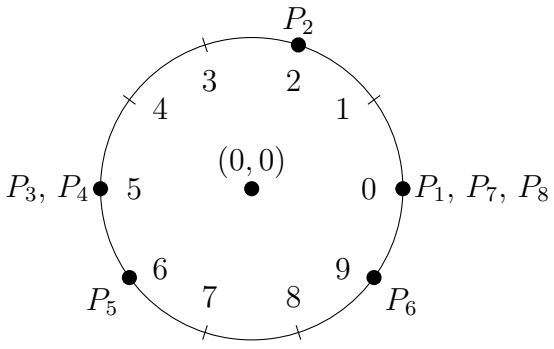
Marks Awarded	Number of Points	Circumference	Additional Constraints
3 marks	$3 \leq N \leq 200$	$3 \leq C \leq 10^6$	None
3 marks	$3 \leq N \leq 10^6$	$3 \leq C \leq 6\ 000$	None
6 marks	$3 \leq N \leq 10^6$	$3 \leq C \leq 10^6$	$P_1, P_2, \ldots, P_N$ are all distinct (i.e., every location contains at most one point)
3 marks	$3 \leq N \leq 10^6$	$3 \leq C \leq 10^6$	None

Output Specification
Output the number of distinct good triplets.

Sample Input
```
8 10
0 2 5 5 6 9 0 0
```

Output for Sample Input
```
6
```

Explanation of Output for Sample Input
Andrew drew the following diagram.



The origin lies strictly inside the triangle with vertices P_1, P_2, and P_5, so $(1, 2, 5)$ is a good triplet. The other five good triplets are $(2, 3, 6)$, $(2, 4, 6)$, $(2, 5, 6)$, $(2, 5, 7)$, and $(2, 5, 8)$.

La version française figure à la suite de la version anglaise.

Problem S5: Good Influencers

Problem Description
There are N ($N \geq 2$) students in a computer science class, with distinct student IDs ranging from 1 to N. There are $N-1$ friendships amongst the students, with the ith being between students A_i and B_i ($A_i \neq B_i$, $1 \leq A_i \leq N$ and $1 \leq B_i \leq N$). Each pair of students in the class are either friends or socially connected. A pair of students a and b are socially connected if there exists a set of students $m_1, m_2, \ldots, m_k$ such that

- a and m_1 are friends,
- m_i and m_{i+1} are friends (for $1 \leq i \leq k-1$), and
- m_k and b are friends.

Initially, each student i either intends to write the CCC (if P_i is `Y`) or does not intend to write the CCC (if P_i is `N`). Initially, at least one student intends to write the CCC, and at least one student does not intend to write the CCC.

The CCC has allocated some funds to pay some students to be influencers for the CCC. The CCC will repeatedly choose one student i who intends to write the CCC, pay them C_i dollars, and ask them to deliver a seminar to all of their friends, and then all of their friends will intend to write the CCC.

Help the CCC determine the minimum cost required to have all of the students intend to write the CCC.

Input Specification
The first line contains the integer N.

The next $N-1$ lines each contain two space-separated integers, A_i and B_i ($1 \leq i \leq N-1$).

The next line contains N characters, $P_1 \ldots P_N$, each of which is either `Y` or `N`.

The next line contains N space-separated integers, $C_1 \ldots C_N$.

The following table shows how the available 15 marks are distributed.

Marks Awarded	Number of students	Payment	Additional Constraints
5 marks	$2 \leq N \leq 2\,000$	$1 \leq C_i \leq 1\,000$	$A_i = i$ and $B_i = i+1$ for each i
7 marks	$2 \leq N \leq 2\,000$	$1 \leq C_i \leq 1\,000$	None
3 marks	$2 \leq N \leq 200\,000$	$1 \leq C_i \leq 1\,000$	None

Output Specification
Output the minimum integer number of dollars required to have all of the students to intend to write the CCC.

La version française figure à la suite de la version anglaise.

Sample Input 1

```
4
1 2
2 3
3 4
YNYN
4 3 6 2
```

Output for Sample Input 1

```
6
```

Explanation of Output for Sample Input 1

The CCC should pay $6 to student 3 to deliver a seminar to their friends (students 2 and 4), after which all 4 students will intend to write the CCC.

Sample Input 2

```
15
1 5
5 2
2 15
15 4
2 10
8 3
3 1
1 6
11 6
12 6
11 9
11 14
12 7
13 7
NNYYYNYYNNNNNNN
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Output for Sample Input 2

```
6
```

Explanation of Output for Sample Input 2

One optimal strategy is for the CCC to ask students 5, 1, 6, 11, 7, and 2 to deliver seminars, in that order, paying them $1 each.

La version française figure à la suite de la version anglaise.

2022 Canadian Computing
Problem 1

Alternating Heights

Time Limit: 2 seconds

Problem Description

Troy is planning to take a group photo of the students at CCO and has asked you for help.

There are K students, numbered from 1 to K. Troy has forgotten the students' heights, but remembers that no two students have the same height.

Troy has prepared a sequence $A_1, A_2, \ldots, A_N$ representing the order of students in the group photo, from left to right. It is possible for a student to appear multiple times in A. You aren't sure how this group photo would be taken, but you're unwilling to assume that Troy made a mistake.

Troy will ask you Q queries of the form x y, which is a compact way of asking "Given the sequence of students $A_x, A_{x+1}, \ldots, A_y$, can their heights form an alternating sequence?" More precisely, we denote the height of the ith student as $h[i]$. If there exists an assignment of heights $h[1], h[2], \ldots, h[K]$ such that $h[A_x] > h[A_{x+1}] < h[A_{x+2}] > h[A_{x+3}] < \ldots h[A_y]$, answer YES; otherwise answer NO.

Note that each of the Q queries will be independent: that is, the assignment of heights for query i is independent of the assignment of heights for query j so long as $i \neq j$.

Input Specification

The first line of input will contain three space-separated integers N, K, and Q.

The second line of input will contain the array $A_1, A_2, \ldots, A_N$ ($1 \leq A_i \leq K$).

The next Q lines will each contain a query of the form of two space-separated integers x and y ($1 \leq x < y \leq N$).

Marks Awarded	**Bounds on N**	**Bounds on K**	**Bounds on Q**
4 marks	$2 \leq N \leq 3000$	$K = 2$	$1 \leq Q \leq 10^6$
6 marks	$2 \leq N \leq 500$	$2 \leq K \leq \min(N, 5)$	$1 \leq Q \leq 10^6$
7 marks	$2 \leq N \leq 3000$	$2 \leq K \leq N$	$1 \leq Q \leq 2000$
8 marks	$2 \leq N \leq 3000$	$2 \leq K \leq N$	$1 \leq Q \leq 10^6$

Output Specification

Output Q lines. On the i^{th} line, output the answer to Troy's ith query. Note that the answer will be either `YES` or `NO`.

Sample Input

```
6 3 3
1 1 2 3 1 2
1 2
2 5
2 6
```

Output for Sample Input

```
NO
YES
NO
```

Explanation of Output for Sample Input

For the first query, we will never have $h[1] > h[1]$ so the answer is no.

For the second query, one solution to $h[1] > h[2] < h[3] > h[1]$ is $h[1] = 160$cm, $h[2] = 140$cm, $h[3] = 180$cm. Another solution could be $h[1] = 1.55$m, $h[2] = 1.473$m, $h[3] = 1.81$m.

For the third query, we cannot have both $h[1] > h[2]$ and $h[1] < h[2]$.

2022 Canadian Computing
Problem 2

Rainy Markets

Time Limit: 1.5 seconds

Problem Description
There are N covered bus shelters, labelled $1, \ldots, N$. The ith bus shelter can fit B_i people inside.

For each $i \in \{1, \ldots, N-1\}$, there is a sidewalk connecting bus shelter i to bus shelter $i+1$, with an open-air market in the middle. The ith market has U_i umbrellas for sale, each costing \$1.

Right now, the ith market has P_i people inside, and every person is in a market so all the bus shelters are empty.

Suddenly, it starts raining, and everyone at market i has to decide between three possibilities:

- to go to bus shelter i;
- to go to bus shelter $i+1$; or
- to stay and buy an umbrella.

If a person is unable to find a place in a bus shelter or buy an umbrella, they will get wet.

If everyone coordinates optimally, can they all stay dry? If so, what is the least amount of money they need to spend, and which bus shelter should each person move to?

Input Specification
The first line of input contains N.

The second line of input contains N space-separated integers B_i $(1 \leq i \leq N)$, the capacity of bus shelter i.

The third line of input contains $N-1$ space-separated integers P_i $(1 \leq i \leq N-1)$, the number of people at market i.

The fourth line of input contains $N-1$ space-separated integers U_i $(1 \leq i \leq N-1)$, the number of umbrellas for sale at market i.

Marks Awarded	Number of bus shelters	Maximum people/bus shelters	Maximum people/market	Maximum umbrellas/market
5 marks	$2 \leq N \leq 10^6$	$0 \leq B_i \leq 2 \cdot 10^9$	$0 \leq P_i \leq 10^9$	$U_i = 0$
5 marks	$2 \leq N \leq 2000$	$0 \leq B_i \leq 400$	$0 \leq P_i \leq 200$	$0 \leq U_i \leq 200$
6 marks	$2 \leq N \leq 4000$	$0 \leq B_i \leq 4000$	$0 \leq P_i \leq 2000$	$0 \leq U_i \leq 2000$
9 marks	$2 \leq N \leq 10^6$	$0 \leq B_i \leq 2 \cdot 10^9$	$0 \leq P_i \leq 10^9$	$0 \leq U_i \leq 10^9$

Output Specification
If every person can stay dry either under an umbrella or at a bus shelter, the output will be $N + 1$ lines:

- the first line will contain the word `YES`.
- the second line will contain the least amount of money necessary to spend on umbrellas
- the next $N - 1$ lines will each contain three space-separated integers:
  - the number of people at market i moving to bus shelter i
  - the number of people at market i buying an umbrella
  - the number of people at market i moving to bus shelter $i + 1$

  where $1 \leq i \leq N - 1$.

If not every person can stay dry, the output will be one line containing the word `NO`.

If there are multiple possible correct outputs, any correct output will be accepted.

Sample Input 1
```
3
10 15 10
20 20
0 0
```

Output for Sample Input 1
```
NO
```

Explanation of Output for Sample Input 1
There are 35 spots available at bus shelters and no umbrellas available, but there are 40 people in the markets.

Sample Input 2
```
3
```

```
10 15 10
20 20
0 11
```

Possible Output for Sample Input 2

```
YES
5
10 0 10
5 5 10
```

Explanation of Output for Sample Input 2

Looking at market 1, 10 people will go to bus shelter 1, no one will buy an umbrella, and 10 people will go to bus shelter 2.

Looking at market 2, 5 people will go to bus shelter 2, 5 people will stay and buy an umbrella, and 10 people will move to bus shelter 3.

In total, 5 umbrellas were purchased, which costs $5.

2022 Canadian Computing
Problem 3
Double Attendance

Time Limit: 3 seconds

Problem Description

Due to a rather ambitious school schedule, two of your classes are about to be held starting at exactly the same time, in two different classrooms! You can only be in one place at a time, so the best you can hope for is catching the important bits of both, even if that means sneaking back and forth between the two.

The two classrooms are numbered 1 and 2. In classroom i, the teacher will show N_i different slides during the class, with the jth slide shown throughout the *exclusive* time interval $(A_{i,j}, B_{i,j})$ $(0 \leq A_{i,j} < B_{i,j})$ where $A_{i,j}$ and $B_{i,j}$ are times elapsed since the start of class measured in seconds. In both classes, there does not exist a point in time at which multiple slides are simultaneously being shown. For example, a class may have slides spanning the pair of intervals $(1, 2)$ and $(5, 6)$, or the pair $(10, 20)$ and $(20, 30)$, but *not* the pair $(10, 20)$ and $(19, 30)$.

You begin the day in classroom 1 with both classes starting at time 0. Following that, at any point in time (not necessarily after an integral number of seconds), you may move from your current classroom to the other one in K seconds. You consider yourself to have seen a slide if you spend a positive amount of time in its classroom strictly within the time interval during which it's being shown. When moving between the two classrooms, you're not considered to be inside either of them for K seconds, and are thus unable to see any slides.

For example, if classroom 1 has a slide being shown for the time interval $(10, 20)$, classroom 2 has a slide being shown for the time interval $(15, 25)$, and $K = 8$, then you'll get to see both slides if you move from classroom 1 to classroom 2 at time 11.5 seconds (arriving at time 19.5 seconds). On the other hand, if you leave classroom 1 at time 17 seconds (arriving at time 25 seconds), then you'll enter classroom 2 just after its slide stops being shown and will therefore miss it.

What's the maximum number of distinct slides which you can see at least once?

Input Specification

The first line contains three space-separated integers, N_1, N_2, and K.

The next N_1 lines each contain two space-separated integers $A_{1,i}$ and $B_{1,i}$ $(1 \leq i \leq N_1)$.

The next N_2 lines each contain two space-separated integers, $A_{2,i}$ and $B_{2,i}$ $(1 \leq i \leq N_2)$.

Marks Awarded	Bounds on N_i	Bounds on $A_{i,j}$ and $B_{i,j}$	Bounds on K
5 marks	$1 \leq N_i \leq 10$	$0 \leq A_{i,j} < B_{i,j} \leq 2000$	$1 \leq K \leq 10^9$
10 marks	$1 \leq N_i \leq 2000$	$0 \leq A_{i,j} < B_{i,j} \leq 2000$	$1 \leq K \leq 10^9$
6 marks	$1 \leq N_i \leq 2000$	$0 \leq A_{i,j} < B_{i,j} \leq 10^9$ $B_{i,j} - A_{i,j} \leq 2K$	$1 \leq K \leq 10^9$
4 marks	$1 \leq N_i \leq 300\,000$	$0 \leq A_{i,j} < B_{i,j} \leq 10^9$	$1 \leq K \leq 10^9$

Output Specification
Output one integer which is the maximum number of distinct slides which you can see.

Sample Input 1
```
3 1 8
10 20
100 101
20 21
15 25
```

Output for Sample Input 1
```
3
```

Explanation of Output for Sample Input 1
One possible optimal strategy is to remain in classroom 1 until time 11.5, then move to classroom 2 (arriving at time 19.5), remain there until time 19.6, and finally return to classroom 1 (arriving at time 27.6). In the process, you'll see all but the 3rd slide. It's impossible for you to see all 4 slides.

Sample Input 2
```
1 5 3
1 100
1 2
2 3
3 4
4 5
5 6
```

Output for Sample Input 2
```
4
```

Explanation of Output for Sample Input 2
Even if you begin moving to classroom 2 immediately at the start of the classes, (e.g., at time 0.0001), you'll miss the first 2 slides shown there. As such, it is possible to see a total of at most four slides.

2022 Canadian Computing

PROBLEM 4

Bi-ing Lottery Treekets

Time Limit: 1 second

Problem Description

In a parallel universe, everyone scored perfect on the CCO. As a result, Troy needs to pick the winner based on a lottery. Each contestant will choose numbers to create a ticket. A ticket is an array of size N indexed from 1 to N where each entry is a number from 0 to K.

The winning ticket is determined by dropping K balls (numbered from 1 to K) in a random sequence into a rooted binary tree. The tree has N nodes (numbered from 1 to N) and is rooted at node 1.

Each ball has a designated drop node that it will drop at. When a ball is dropped at an unoccupied node or enters an unoccupied node, one of three things happens:

1. If all of the current node's children are occupied by balls (or if a node has no children), the current ball rests at the current node. That is, it remains there and does not move again.
2. If the current node only has one unoccupied child, the current ball will move to this child.
3. If the current node has two unoccupied children, and if the current ball was just dropped, it could go either left or right. Otherwise, it will continue in the direction of its previous movement.

If all K balls cannot be dropped, a winning ticket is not determined. This happens when a ball is dropped and its drop node is occupied by another ball.

If all K balls have been dropped, the balls' resting positions determine the winning lottery ticket. The ith entry of the winning lottery ticket is the number of the ball that rests at node i or 0 if no ball rests at node i.

Troy would like to know the number of possible winning tickets (which could be zero).

Input Specification

The first line contains two space-separated integers N and K, denoting the number of nodes in the binary tree and the number of balls, respectively.

The next line contains K space-separated integers, where the ith integer denotes the designated drop node of the ball numbered i.

The last N lines each contain two space-separated integers. The ith line contains L_i and R_i denoting the ith node's left and right child, respectively, where 0 means no such child exists.

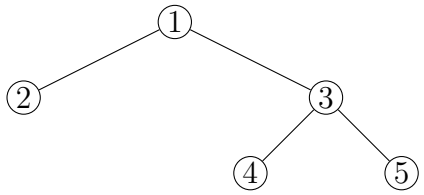
Marks Awarded	Bounds on N	Bounds on K	Additional constraints
3 marks	$1 \leq N \leq 12$	$1 \leq K \leq 6$	None
4 marks	$1 \leq N \leq 4000$	$1 \leq K \leq 4000$	All nodes do not have a left child
9 marks	$1 \leq N \leq 4000$	$1 \leq K \leq 4000$	The N drop nodes are distinct
9 marks	$1 \leq N \leq 4000$	$1 \leq K \leq 4000$	None

Output Specification
Output the remainder of the number of winning lottery tickets divided by $10^9 + 7$.

Sample Input 1
```
5 2
1 3
2 3
0 0
4 5
0 0
0 0
```

Output for Sample Input 1
```
4
```

Explanation of Output for Sample Input 1
The tree looks like this:



Consider when ball 1 is dropped first. If ball 1 goes left, then it will rest at node 2. Afterward, ball 2 is dropped and can rest at node 4 or 5. If ball 1 goes right, it will rest at node 5. Then, ball 2 will rest at node 4.

Consider when ball 2 is dropped first. Ball 2 can go left or right, resting at nodes 4 or 5, respectively. Then if ball 1 moves left after being dropped, it will rest at node 2. However, if ball 1 moves right, it will rest at either node 4 or 5, whichever place ball 2 does not occupy.

The possible winning tickets are [0, 1, 0, 2, 0], [0, 1, 0, 0, 2], [0, 0, 0, 1, 2], and [0, 0, 0, 2, 1].

Sample Input 2

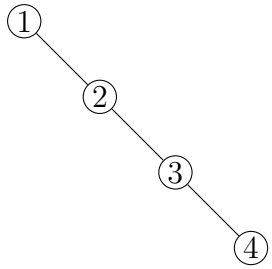
```
4 3
1 2 4
0 2
0 3
0 4
0 0
```

Output for Sample Input 2

```
2
```

Explanation of Output for Sample Input 2

The tree looks like this:



This test case satisfies the second subtask.

Ball 3 must be dropped first because if either ball 1 or ball 2 are dropped before ball 3, it would rest at node 4, which wouldn't allow ball 3 to be dropped.

Thus, we can either first drop ball 3, then ball 2 and finally ball 1 or we can first drop ball 3, then ball 1 and finally ball 2.

The possible winning tickets are [0,1,2,3] and [0,2,1,3].

Formal Definitions

A *binary tree* is a set of nodes that is either empty, or a root node with a left subtree and a right subtree both of which are binary trees. Given a node x, if its left subtree is not empty, then the root of that subtree is called the *left child* of x. Similarly, given a node x, if its right subtree is not empty, then the root of that subtree is called the *right child* of x.

2022 Canadian Computing

PROBLEM 5

Phone Plans

Time Limit: 3 seconds

Problem Description

The mayor of CCOland, Jason, wants to install telephone lines amongst N households, which are numbered from 1 to N. To do so, he has asked two rivalling companies, Keenan Mobile Phones and Chris Home Telephone, for their phone plans. A phone plan for a company corresponds to a certain level and every telephone line has a level and company associated with it. If you have purchased a phone plan from a company with level l, then you are able to use all the telephone lines whose level is less than or equal to l that is associated with that company. A phone plan of level l costs $\$l$ and you cannot pick a phone plan of less than \$0.

Two households can only communicate with each other if they are connected by a path of telephone lines of the **same company**. Jason would like to buy one phone plan from each company of minimal cost such that there are at least K different pairs of households that can communicate with each other.

Input Specification

The first line contains four space-separated integers N, A, B and K, which represent the number of households, number of telephone lines from Keenan Mobile Phones, number of telephone lines from Chris Home Telephone and the minimum pairs of homes that need to be able to communicate with each other, respectively.

The next A lines each contain three space-separated integers u, v and l, which represents a Keenan Mobile Phones telephone line between household u and v ($1 \leq u, v \leq N$) that has a level l ($1 \leq l \leq 10^9$).

The next B lines have the same format as the previous A lines but for Chris Home Telephone.

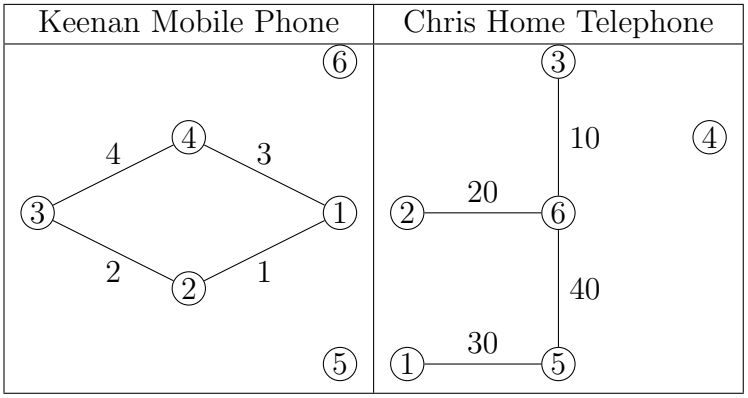
Marks Awarded	**Bounds on N**	**Bounds on A and B**	**Bounds on K**	**Additional Constraints**
6 marks	$1 \leq N \leq 2000$	$0 \leq A, B \leq 200\,000$	$0 \leq K \leq \frac{N(N-1)}{2}$	None
5 marks	$1 \leq N \leq 200\,000$	$0 \leq A, B \leq 200\,000$	$0 \leq K \leq \frac{N(N-1)}{2}$	Keenan Mobile Phones only connects to odd indexed households; Chris Home Telephone only connects to even indexed households
6 marks	$1 \leq N \leq 200\,000$	$0 \leq A \leq 10$; $0 \leq B \leq 200\,000$	$0 \leq K \leq \frac{N(N-1)}{2}$	None
8 marks	$1 \leq N \leq 200\,000$	$0 \leq A, B \leq 200\,000$	$0 \leq K \leq \frac{N(N-1)}{2}$	None

Output Specification
Output the cheapest cost needed to connect at least K different pairs of households or -1 if it is not possible.

Sample Input
```
6 4 4 9
1 2 1
2 3 2
1 4 3
3 4 4
5 6 40
1 5 30
2 6 20
3 6 10
```

Output for Sample Input
```
33
```

Explanation of Output for Sample Input
For each company, consider these pictures of the way the 6 households are connected by telephone lines:



If Jason buys phone plan level 3 from Keenan Mobile Phones and phone plan level 30 from Chris Home Telephone, then $(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)$ can communicate through Keenan Mobile Phones's lines and $(1,5),(2,6),(3,6),(2,3)$ can communicate through Chris Home Telephone's lines. There are no cheaper ways.

2022 Canadian Computing

PROBLEM 6

Good Game

Time Limit: 1 second

Problem Description

Finn is playing a game of Twos and Threes. Twos and Threes is a one-player game played on a one-dimensional board. In the starting position, there are N blocks arranged in a row, with each block labelled either A or B. Blocks are numbered from 1 to N from left to right. Finn is allowed to make moves of the following form:

- Select 2 or 3 consecutive blocks that share the same label. Remove them from the board. Connect any remaining blocks together. Re-index the blocks from left to right starting with index 1.

Finn wins the game if all blocks are removed from the board. Your task is to help Finn determine a winning sequence of moves, or determine if the game cannot be won.

Input Specification

The first line of input will contain the integer N.

The second line of input will contain the string S which is the starting position of the game. There are N characters in S, and each of these characters in S is either A or B.

Marks Awarded	**Bounds on N**
3 marks	$1 \leq N \leq 15$
6 marks	$1 \leq N \leq 300$
7 marks	$1 \leq N \leq 6000$
9 marks	$1 \leq N \leq 10^6$

Output Specification

If there is a winning sequence of moves, output K, the number of moves in the winning sequence. On each of the next K lines, print an index i, followed by one space, followed by a number j, denoting a move that will remove the blocks currently at indices i to $i + j - 1$, inclusive.

If there is no winning sequence of moves, output -1.

If there are multiple winning sequences, then any winning sequence will be accepted. There is no need to minimize or maximize K.

Sample Input

```
9
ABAABBBAA
```

Possible Output for Sample Input

```
4
6 2
3 2
2 2
1 3
```

Explanation of Output for Sample Input

The sample output denotes this winning sequence:

$$ABAAB\underline{BB}AA$$
$$AB\underline{AA}BAA$$
$$A\underline{BB}AA$$
$$\underline{AAA}$$

Sample Input 1

```
4
1 2
2 3
3 4
YNYN
4 3 6 2
```

Output for Sample Input 1

```
6
```

Explanation of Output for Sample Input 1

The CCC should pay $6 to student 3 to deliver a seminar to their friends (students 2 and 4), after which all 4 students will intend to write the CCC.

Sample Input 2

```
15
1 5
5 2
2 15
15 4
2 10
8 3
3 1
1 6
11 6
12 6
11 9
11 14
12 7
13 7
NNYYYNYYNNNNNNN
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Output for Sample Input 2

```
6
```

Explanation of Output for Sample Input 2

One optimal strategy is for the CCC to ask students 5, 1, 6, 11, 7, and 2 to deliver seminars, in that order, paying them $1 each.

La version française figure à la suite de la version anglaise.

J1 Cupcake Party

Solving this problem involves correctly reading the input and performing some arithmetic. The provided samples illustrate how to compute the number of cupcakes. Then, 28 must be subtracted from this value to determine the number of cupcakes left over.

J2 Fergusonball Ratings

Loops and if statements are required for this problem. A clean approach is to loop N times and read two lines of input per loop. This allows you to process one player at a time. A calculation can be performed to determine the player's star rating, and we can also maintain a count of the number of players with a star rating greater than 40. Care may be needed to produce only one line of output exactly as specified.

J3 Harp Tuning

There is a lot to consider when attempting to solve this problem. The biggest challenge is to break the input into pieces. This is known as tokenizing the input and the pieces are typically called tokens.
Specifically, the first subtask requires breaking a single instruction into a sequence of letters, the character + or -, and the number of turns. The focus of the second subtask is to break the input into different instructions, but each instruction has a simpler form. The third subtask combines these two objectives. The final subtask adds an extra layer of difficulty by allowing the number of turns to be a multi-digit number.
One elegant way to tokenize the input is to iterate through the input character by character while also remembering the last character read. This allows you to recognize when and where one token begins and another token ends. An alternative is to keep track of the current "state" which is an indication of which of the three types of token is currently under consideration.
Outputting the translation of each instruction as it is tokenized, rather than storing up all the translations until the end, removes the need for any extra data structures.

J4 Group Work

The fourth problem in this year's competition is the first that requires a collection of input to be stored together in a data structure. Because the assignment of students to groups is not given until the end of the input, the constraints cannot be fully considered as they are encountered, and so must be stored in memory. It is helpful to also store the student groups in memory.
Once both types of input are in memory, a solution can be obtained by considering each constraint individually and maintaining a count of how many are violated.
Likely pitfalls include incorrectly assuming something about the order in which student names appear, and erroneously thinking that a group can only violate one constraint.
Care must be taken with the Boolean logic required to test if a given constraint is violated. This will require determining if a given student is in a given group. Doing this inefficiently by looping through all the groups or looping through all the students should have allowed a submission to earn 14 of the available 15 marks. To earn the final mark, this look-up must be made more efficient. This can be accomplished using a data structure built-in to the language designed for this very purpose (e.g. a dictionary in Python, a HashMap in Java, or an unordered map in C++).

J5 Square Pool

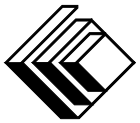
Any participant who was able to make any progress on this problem (or the previous problem, J4) is strongly encouraged to attempt the Senior Contest next year if they remain eligible. Earning more than 8 marks on J5 was especially difficult and intended to provide a significant challenge.
The first subtask can be solved by recognizing that when there is only one tree, a largest square pool must sit at one of the "corners" of the tree. An example of this is drawn in the explanation for the first sample input where a largest pool can sit at the "bottom-left corner" of the tree. As a result, the first three marks

can be earned by considering the four possible cases and taking the maximum value.

Once there is more than one tree, a lot more work must be performed. For the second subtask, an approach that uses brute-force will earn full marks. There are different ways to accomplish this but they all amount to essentially testing all possible locations and sizes of a square pool and for each possibility, determining whether or not it contains a tree. Doing this successfully is done most naturally using several nested loops.

Allowing yards to be as large as 500 000-by-500 000 prevents a brute-force approach from finishing within the time limit. However, an important observation for the third subtask is that the number of trees is still guaranteed to be quite small. This is a clue that we might want a solution which depends only on the number of trees and not on the size of the yard. Recursion can be used to do just this. The key idea of the first subtask can be reused here. By inspecting one tree, we can limit our further search for an optimal placement of the pool to above, below, to the left, or to the right of this tree. This reduces the problem to a smaller (now possibly rectangular) yard.

Recursion will fail once the number of trees climbs much higher than 10 because the runtime will grow exponentially as the number of trees increases. Therefore, we require a solution which still depends only on the number of trees, but to a lesser extent. A very clever idea allows us to accomplish this. A square pool that is as large as possible can be moved up until it hits a tree or the upper boundary of the yard. Similarly, a square pool that is as large as possible can be moved left until it hits a tree or the left boundary of the yard. If we imagine adding coordinates to the yard, this means that each tree and the upper boundary provide $T + 1$ candidates for the topmost position of an optimally placed pool. Similarly, there are $T + 1$ candidates for the leftmost position of an optimally placed pool. By considering all pairs consisting of a candidate for the leftmost position and a candidate for the topmost position, we considerably limit the number of possible locations for an optimally placed pool. Moreover, we can figure out the size of the largest possible pool at one of these possible locations by only considering the remaining trees and boundaries. This approach yields a solution with a runtime that computer scientists characterize as cubic in T .



The CENTRE for EDUCATION in MATHEMATICS and COMPUTING

cemc.uwaterloo.ca

2022 CCC Senior Problem Commentary

Creation of this commentary is a new initiative for the Canadian Computing Competition. Our goal is to give a brief outline of what is required to solve each problem and what challenges may be involved. The notes below can be used to guide anyone interested in trying to solve these problems, but are not intended to describe all the details of a correct solution. We encourage everyone to try and implement these details on their own using their programming language of choice.

S1 Good Fours and Good Fives

The first problem is designed to be accessible with some insight required to obtain full marks.

The first subtask can be hard-coded. That is, the following table can be calculated by hand and a solution can mimic looking up the correct output in the table.

N	1	2	3	4	5	6	7	8	9	10
Output	0	0	0	1	1	0	0	1	1	1

For the second and third subtasks, we can notice that we need $N = 4a + 5b$ for non-negative integers a and b where $a \leq \frac{n}{4}$ and $b \leq \frac{n}{5}$. This means we can try all possible values for a and b using two nested loops.

For a full solution, we can loop through only all possible values of a to determine if there is a corresponding value of b. For each of these values i, this is equivalent to checking if $N - 4i$ is divisible by 5. Alternatively, we take a similar approach but loop through only all possible values of b.

There is also a clever extremely quick solution that involves extending the table listed above for the first subtask to $N = 20$ and considering the quotient and remainder when N is divided by 20.

S2 Good Groups

The second problem in this year's competition requires input to be stored together in a data structure. Because the assignment of students to groups is not given until the end of the input, the constraints cannot be fully considered as they are encountered, and so must be stored in memory. It is helpful to also store the student groups in memory.

Once both types of input are in memory, a solution can be obtained by considering each constraint individually and maintaining a count of how many are violated.

Likely pitfalls include incorrectly assuming something about the order in which student names appear, and erroneously thinking that a group can only violate one constraint.

Care must be taken with the Boolean logic required to test if a given constraint is violated. This will require determining if a given student is in a given group. Doing this inefficiently by looping through all the groups or looping through all the students should have allowed a submission to earn 14 of the available 15 marks. To earn the final mark, this look-up must be made more efficient. This can be accomplished using a data structure built-in to the language designed for this very purpose (e.g. a dictionary in Python, a `HashMap` in Java, or an `unordered_map` in C++).

S3 Good Samples

For the first subtask, it suffices to run a brute-force check of all possible outputs to determine if there is a piece that will satisfy the clarinet players.

For the second subtask, note that no sequence of length 3 is good. Also, all sequences of length 1 are good. Thus, we should try to find a piece with $K - N$ good samples of length 2. We can build it as we go. From the left, if we still need good samples, we can swap to the other pitch, otherwise, we can keep the same pitch. Note that all values between N and $2N - 1$ inclusive are possible and no others are.

For the third subtask, we can build on our solution to the second subtask. As before, suppose we are building a piece, our current sequence is S, and our required count is $K - N$. Depending on how many more good samples we need, we can decide what to append to our sequence S. For instance, if we no longer need any good samples, we can simply add a value equal to the current last value of the sequence. If we need 3 more, we can add the note 4 notes back. That way, there are 3 additional good samples. We can also pick a new number if we want even more good samples. It suffices to greedily choose the number that gives us the most samples up to the remaining amount. When implementing this idea, it's a good idea to keep track of the longest suffix of your sequence S that is good.

The last subtask involves the same idea but when adding a new number to the end, we have to instead do a proper check to see if the new number is too large.

S4 Good Triplets

For the first subtask, it suffices to follow the definition of a good triplet. Firstly, there are $O(N^3)$ ways to select (a, b, c). Secondly, consider the condition that the origin is strictly inside the triangle. This is equivalent to saying that P_a, P_b, and P_c divide the circle into 3 arcs, and each arc is strictly shorter than $\frac{C}{2}$. This leads to the following approach. Sort the positions so that $P_a \leq P_b \leq P_c$, then check that

$$P_b - P_a < \frac{C}{2}, P_c - P_b < \frac{C}{2}, \text{ and } (P_a + C) - P_c < \frac{C}{2}.$$

For example, the first of these checks can be implemented in code as `P[b]-P[a] < (C+1)/2` in C++ or Java, or `P[b]-P[a] < (C+1)//2` in Python 3.

For the second subtask, the goal is to find a solution that works well when C is small. Let L_x be the number of points drawn at location x. If three locations i, j, k satisfy the second condition, we want to add $L_i \times L_j \times L_k$ to the answer. This approach is $O(N + C^3)$ and is too slow. However, we can improve from three nested loops to two nested loops. If i and j are chosen already, either k does not exist, or k is in an interval. It is possible to replace the third loop with a prefix sum array. This algorithm's time complexity is $O(N + C^2)$.

It is possible to optimize this approach to $O(N + C)$ and earn 15 marks. The idea is to eliminate both the j and k loop. Start at $i = 0$ and compute $L_j \times L_k$ in $O(C)$ time. The next step is to transition from $i = 0$ to $i = 1$, and to maintain $L_j \times L_k$ in $O(1)$ time. This requires strong familiarity with prefix sum arrays. Care with overflow and off-by-one errors is required.

S5 Good Influencers

In the first subtask, the students and their friendships form a line.

One approach to this subtask involves dynamic programming. Let $DP[i]$ be the minimum cost required for the first i students to end up intending to write the CCC (ignoring any later students). Note that $DP[0] = 0$, and our answer will be $DP[N]$.

La version française figure à la suite de la version anglaise.

For each value of i from 0 to $N-1$, we'll consider transitions onwards from that state, with the subarray of students from $i+1$ to j (for each possible j such that $i < j \leq N$) ending up intending to write the CCC. If at least one of those students has a P value of `Y`, then it's possible to perform this transition by choosing a `Y` student and "expanding their influence" to all others, updating $DP[j]$ accordingly.

For each i, we can consider all possible values of j while computing their transitions' minimum costs in a total of $O(N)$ time, resulting in an overall time complexity of $O(N^2)$.

To solve the remaining subtasks, we'll use a different dynamic programming formulation, this time on the tree structure formed by the students and their friendships. We'll consider the tree to be rooted at an arbitrary node (student), such as node 1.

Let $DP[i][j]$ (defined for $1 \leq i \leq N$ and $0 \leq j \leq 2$) be the minimum cost required for all students in i's subtree to end up intending to write the CCC, such that:

- if $j = 0$, i will be influenced by its parent
- if $j = 1$, i will have no particular interaction with its parent
- if $j = 2$, i will influence its parent

Our answer will then be $DP[1][1]$.

As is typical for DP on trees, we'll recurse through the tree, and for each node i, we'll compute $DP[i][0..2]$ based on the DP values of i's children. Each value $DP[i][j]$ must be computed carefully, with different logic depending on the values of $P[i]$ and j.

An implementation of this algorithm may take $O(N^2)$ time (with some DP transitions taking $O(N)$ time each to compute), but optimizations can be applied to reduce it to an overall time complexity of $O(N)$ and have it earn full marks.