Nama: Raina Imtiyaz

Data Science

Binus University

Google Colab Link: <https://colab.research.google.com/drive/14A3zQPsZuIyofNI5ddLWGGpT4_xSzfSJ?usp=sharing>

Contents

[Mixed Team 2](#_Toc148267388)

[Steak House 5](#_Toc148267389)

[Obstacle Course 7](#_Toc148267390)

[Game Of Math 9](#_Toc148267391)

# 

# Mixed Team

B University has N students, numbered from 1 to N, Student i has a rating of Ri representing their estimated skill in competitive problem-solving.

Contest season is coming and Morgan, the coach of the club, would like to send one good team. A team consists of exactly 3 different students. Suppose that a team consists of student i, j, and k. Their team rating is Ai + Aj + Ak, and their rating difference is max(Ai , Aj , Ak ) − min(Ai , Aj , Ak ).

Morgan believes that a team is balanced if their rating difference is no more than a threshold of M. Additionally, he also would like the team rating to be as large as possible while being balanced as well.

Morgan asks you to compute two values. The first value is the number of different balanced team configurations that can be made. The second value is the largest team rating of a balanced team that can be made.

Two team configurations are different if and only if there is at least one different student between those team configurations.

**Input Format**

Input begins with two integers N M (3 ≤ N ≤ 200; 0 ≤ M ≤ 4000) representing the number of students and the threshold for rating difference, respectively. The next line contains N integers Ai (0 ≤ Ai ≤ 4000) representing the rating of student i.

**Constraints**

-

**Output Format**

If there is at least one balanced team configuration, then output two space-separated integers in a single line representing the number of different balanced team configurations and the largest team rating of any balanced team, respectively.

If there is no balanced team configuration, then output -1 in a single line.

**Sample Input 0**

5 150

1400 1425 1250 4000 1300

**Sample Output 0**

2 4125

**Explanation 0**

An example of a balanced team configuration is the team consisting of student 1, 3, and 5. Their team rating is 1400 + 1250 + 1300 = 3950. Their rating difference is 1400 − 1250 = 150, which is no more than 150.

The other balanced team configuration is the team consisting of students 1, 2, and 5. Their rating difference is 125 with a team rating of 4125, which is the highest team rating among all balanced team configurations that can be made.

**Sample Input 1**

4 100

2000 1900 1800 2100

**Sample Output 1**

-1

**Explanation 1**

Any team configuration has a rating difference of at least 200, which is more than the given threshold.

**Sample Input 2**

8 4000

100 200 300 400 500 600 700 800

**Sample Output 2**

56 2100

**Explanation 2**

Any team configuration in this example is a balanced team, while the team consisting of students 6, 7, and 8 has the largest team rating of 600 + 700 + 800 = 2100

**Sample Input 3**

8 0

10 10 10 20 20 20 30 30

**Sample Output 3**

2 60

**Explanation 3**

There are only 2 possible balanced team configurations: A team with students 1, 2, and 3 with a total team rating of 10+10+10 = 30, and a team with students 4, 5, and 6 with a total team rating of 20+20+20 = 60. The latter has the largest team rating.

# 

# Steak House

Morgan is a chef in a steak house. In his steak house, a steak can have N level of doneness, numbered from 1 to N. Currently, Morgan has Ai steaks of doneness level i ready in his steak house.

There are Bi orders of steaks with doneness level i that need to be fulfilled. Morgan can cook the steaks in order to match the doneness level. For each 1 ≤ i < N, it takes Morgan Ti seconds to cook a steak from doneness level i to i + 1. Note that Morgan can only cook one steak at a time.

Morgan asks for your help to find the minimum total time to fulfil all orders, or tell him that the orders are impossible to fulfil.

**Input Format**

Input begins with an integer N (2 ≤ N ≤ 100 000).

The next line contains N − 1 integers Ti (1 ≤ Ti ≤ 1000) representing the time required to cook a steak of doneness level i to i + 1.

The next line contains N integers Ai (0 ≤ Ai ≤ 1000) representing the number of steaks with doneness level i.

The next line contains N integers Bi (0 ≤ Bi ≤ 1000) representing the number of orders for a steak with doneness level i.

**Constraints**

-

**Output Format**

If all orders can be fulfilled, then output an integer in a single line representing the minimum total time to fulfil all orders. Otherwise, output -1 in a single line.

**Sample Input 0**

3

1 2

2 2 3

0 1 5

**Sample Output 0**

5

**Explanation 0**

First, Morgan can cook both steaks with doneness level 2 to level 3 in 2 seconds each. Then, Morgan can cook one steak with doneness level 1 to level 2 in 1 second. Now, Morgan has 1 steak of doneness level 1, 1 steak of doneness level 2, and 5 steaks of doneness level 3. It is enough to fulfil all orders. There is no other way to fulfil all orders in less than 5 seconds.

**Sample Input 1**

3

1 2

2 2 3

1 2 1

**Sample Output 1**

0

**Explanation 1**

The steaks ready in his steak house can fulfil all orders without any further cooking.

**Sample Input 2**

3

1 2

2 2 3

5 0 0

**Sample Output 2**

-1

**Explanation 2**

It is impossible to have 5 steaks of doneness level 1.

# 

# Obstacle Course

As a businessman and parkour enthusiast, Morgan owns an obstacle course consisting of N road segments numbered from 1 to N . These road segments are arranged sequentially such that road 1 is adjacent to road 2, road 2 is adjacent to road 3, ..., road N − 1 is adjacent to road N . Currently, each odd-numbered road segment has a height of a positive integer Ai, and each even-numbered road segment has a height of 0. In Morgan’s course, anyone can start from any road segment, parkour towards one direction, and stop at any road segment as long as they can reach that road segment.

As a beginner athlete, Adrian thinks that Morgan’s obstacle course may be too hard due to the height dif- ference between any two consecutive roads. Adrian enjoys doing parkour whenever the height difference between two consecutive roads is exactly 1. If the next road segment has a difference of more than 1 from the current road segment, then Adrian cannot climb or jump down to the next road segment and will stop his parkour activity. On the other hand, if the next road segment has the same height as the current road segment, then Adrian will lose interest and stop his parkour activity. He will also stop his parkour activity if he reaches the end of the obstacle course, i.e. road segment N or road segment 1.

As his friend, Morgan wants Adrian to be happy with the course. Adrian’s happiness is defined as the number of road segments that he parkoured before he lost interest and stops his parkour activity. Adrian will start from any road segment that will give him the largest happiness. Based on these facts, Morgan decided to change the heights of some (possibly none) even-numbered road segments such that Adrian’s largest happiness for the course is maximized. However, before he makes any changes, he asked you to determine the largest possible Adrian’s happiness that can be obtained after changing the height.

For example, let A1..9 = [6, 0, 4, 0, 8, 0, 12, 0, 14]. By changing A into [6, 5, 4, 5, 8, 11, 12, 13, 14], Adrian can do parkour from segment 1 to 4 or from segment 6 to 9, and his happiness will be 4, which is also the largest happiness after that change. There are some other changes that can make Adrian’s largest happiness equal to 4 as well, e.g. [6, 5, 4, 3, 8, 7, 12, 12, 14], but no change can make Adrian’s largest happiness larger than 4.

**Input Format**

Input begins with an integer N (2 ≤ N ≤ 100 000) representing the number of road segments. The next line contains N integers Ai (Ai = 0 if i is even; 1 ≤ Ai ≤ 100 000 if i is odd) representing the height of segment i from 1 to N respectively.

**Constraints**

-

**Output Format**

Output an integer in a single line that represents the largest possible Adrian’s happiness in Morgan’s new course.

**Sample Input 0**

9

6 0 4 0 8 0 12 0 14

**Sample Output 0**

4

**Explanation 0**

This is the example from the problem description.

**Sample Input 1**

7

3 0 5 0 5 0 7

**Sample Output 1**

7

**Explanation 1**

By changing A to [3, 4, 5, 6, 5, 6, 7] Adrian’s happiness will be 7.

**Sample Input 2**

6

6 0 2 0 8 0

**Sample Output 2**

3

**Sample Input 3**

7

4 0 6 0 10 0 9

**Sample Output 3**

4

# Game Of Math

Adrian is playing a game. When the game starts, Adrian will be given P points as his initial points. The game consists of N rounds, numbered from 1 to N. During round i, Adrian has two options. Each option can be one of the following types:

* + c (−1000 ≤ c ≤ 1000) which will add his current points by c, or
* x c (−2 ≤ c ≤ 2) which will multiply his current points by c.

Adrian wants to maximize his points at the end of the game.

**Input Format**

Input begins with two integers N P (1 ≤ N ≤ 50; −1000 ≤ P ≤ 1000) representing the number of rounds and the initial points during the game, respectively. Each of the next N lines contains the two options in each round separated by a space. Each option is given in the format T c (T ∈{+,x};−1000 ≤ c ≤ 1000 if T = +, or − 2 ≤ c ≤ 2 if T = x).

**Constraints**

-

**Output Format**

Output an integer in a single line representing the maximum points Adrian can achieve at the end of the game.

**Sample Input 0**

3 123

+ 100 x 2

+ -100 x -2

+ 0 + 0

**Sample Output 0**

146

**Explanation 0**

Adrian can choose the second option in round 1, first option in round 2, and any option in round 3.

**Sample Input 1**

3 123

+ 100 x 2

+ -100 x -2

x 0 x 0

**Sample Output 1**

0

**Explanation 1**

Adrian will always achieve 0 points regardless of his decision in each round, because round 3 will multiply his points by 0.