Homework #1

Blue Correction Date: 09/20/2022 14:20

Red Correction Date: 09/16/2022 21:00

Due Time: 2022/10/06 14:20

Contact TAs: `ada-ta@csie.ntu.edu.tw`

Instructions and Announcements

- There are **four programming problems** and **two hand-written problems**
- **Programming.** The judge system is located at `https://ada-judge.csie.ntu.edu.tw`. Please login and submit your code for the programming problems (i.e., those containing "Programming" in the problem title) by the deadline. NO LATE SUBMISSION IS ALLOWED.
- **Hand-written.** For other problems (also known as the "hand-written problems"), you should upload your answer to **Gradescope** as demonstrated in class. For each sub-problem, please label (on Gradescope) the corresponding pages where your work shows up. NO LATE SUBMISSION IS ALLOWED.
- **Collaboration policy.** Discussions with others are strongly encouraged. However, you should write down your solutions **in your own words**. In addition, for **each and every** problem you have to specify the references (e.g., the Internet URL you consulted with or the people you discussed with) on the first page of your solution to that problem. You may get zero point due to the lack of references.
- **Tips for programming problems.** Since the input files for some programming problems may be large, please add
  - `std::ios_base::sync_with_stdio(false);`
  - `std::cin.tie(nullptr);`

  to the beginning of the main function if you are using `std::cin`.

Problem 1 - Omelet and the LEGO® Tower (Programming) (10 points)

Problem Description

Omelet loves playing with LEGO® bricks. His best friend Miku gave him N LEGO bricks as his birthday gift. These N bricks are not ordinary LEGO bricks. They are legendary Miku bricks! Each legendary Miku brick has a cute and unique photo of Miku on top of it. For convenience, we label these bricks from 1 to N.

Now, Omelet wants to take photos of the towers he built and send them to Miku so that he can prove his love to Miku.

To be more precise, Omelet wants to take **exactly** N photos, the i-th of them being an *i-perfect Miku tower*, defined as follows.

- A *perfect Miku tower* consists of exactly N legendary Miku bricks stacking up one above another.
- An *i-perfect Miku tower* is a *perfect Miku tower* with the legendary Miku brick labeled i on the top of the tower.

Initially, all N bricks are on the playground in Omelet's house, and the Miku tower is empty. There are three types of operation Omelet can do.

1. Pick one brick labeled with X and place it on top of the current tower(which could be empty).
   Notice that he can only do this when the brick labeled with X is not on the Miku tower.
2. Put the brick on top of the Miku tower back to the playground.
   Notice that he can only do this when the Miku tower is not empty.
3. Take a photo of the current Miku tower.
   Notice that he can only do this when the current Miku tower is a *perfect Miku tower*.

Hating to move his fingers, Omelet only does these operations at most $4 \cdot 10^6$ times. Can you help him by telling him a sequence of operations that achieves his goal?

Input

One positive integer $1 \leq N \leq 10^5$ in a line.

Output

Print an integer M on the first line, indicating the total number of operations you use.

M lines follow, each line is in one of the forms

1. `PLACE` X, which corresponds to the first type of operation, and X is the label of the brick.
2. `POP`, which corresponds to the second type of operation.
3. `PHOTO`, which corresponds to the third type of operation.

You will get `Accepted` if and only if

- $1 \leq M \leq 4 \cdot 10^6$.

- Your operations are valid.
- Omelet achieves his goal by doing these operations in order.

Notice that you **don't need to minimize** the number of operations.

Test Group 0 (0 %)

- Sample Input.

Test Group 1 (5 %)

- $1 \leq N \leq 1000$

Test Group 2 (10 %)

- $1 \leq N \leq 10000$

Test Group 3 (65 %)

- $1 \leq N \leq 50000$

Test Group 4 (20 %)

- No additional constraints.

Sample Input 1

```
2
```

Sample Output 1

```
8
PLACE 2
PLACE 1
PHOTO
POP
POP
PLACE 1
PLACE 2
PHOTO
```

Sample Input 2

```
1
```

Sample Output 2

```
2
PLACE 1
PHOTO
```

Sample Input 3

```
1
```

Sample Output 3

```
7
PLACE 1
PHOTO
POP
PLACE 1
POP
PLACE 1
POP
```

Problem 2 - Cash (Programming) (10 points)

Problem Description

Zisk is a wise man that knows everything. Little Z, on the other hand, is a normal guy.

Little Z is paid for his daily work by **cash**, whose amount is an integer between 1 and W (inclusive). After hearing an investment advice that "Do not hold too much **cash**," he hopes to minimize the amount of his cash by purchasing **gem**. There are N types of **gem**, and each has an unlimited supply. The price of the i-th types of **gem** is denoted by a_i.

Little Z uses a simple strategy to choose what types of **gem** to buy, one at a time: Given the amount of **cash** he currently holds, he buys the most expensive types of **gem** he can afford. He repeats the above action until he cannot buy anymore.

Little Z found that sometimes this simple strategy might be suboptimal. A strategy is optimal if and only if the remaining **cash** is the smallest among all possible strategies.

Hence, he wants to know, for every possible initial amount of **cash** (each integer from 1 to W), how much more **cash** he could have invested if he uses the optimal strategy.

Input

The first line of input contains two integers N, W, separated by a space. Each of the following N lines contains an integer represents a_i, the price of the i-th type of gem.

- $1 \leq N \leq 5000$
- $1 \leq W \leq 5000$
- $1 \leq a_i \leq 5000$

Output

Output W lines of non-negative integers, the i-th of which represents the difference between Little Z's plan and the optimal one when the initial amount of **cash** is i.

Test Group 0 (0 %)

- Sample Input.

Test Group 1 (30 %)

- $W \leq 50$

Test Group 2 (70 %)

- No additional constraints.

Sample Input 1

```
2 6
3
5
```

Sample Output 1

```
0
0
0
0
0
1
```

Sample Input 2

```
4 20
6
8
10
12
```

Sample Output 2

```
0
0
0
0
0
0
0
0
0
0
0
0
0
2
2
4
4
0
0
0
```

Hint

- In the first sample test case, $N = 2; W = 6; a = [3, 5]$. When the initial amount of **cash** is 6, the optimal plan is different from the plan which Little Z's strategy gives. Little Z's strategy will buy only the gem with price 5, while the optimal strategy is to buy the gem with price 3 twice.
- By the way, **cash** is a currency unit in the Grand ADA Kingdom.
- The prices of **gems** are not guaranteed to be sorted. The a_i in the input might be given in arbitrary order.

Problem 3 - Monsoon in Kingdom (Programming) (15 points)

Problem Description

As a clever student, ZCK exchanges to the ADA kingdom this semester. There are n cities in the kingdom, and there is a flight connecting each pair of cities. Due to the special geographical location, monsoon often affects flights in the fall. To ensure flight safety, an airplane cannot fly in some range of directions for a long time.

Specifically, for a pair of cities A and B, if the slope of $\overleftrightarrow{AB}$ is in a specific range $[l, r]$, then the flight connecting them cannot fly in the fall.

Professor Chen finds that ZCK is very smart, so she gives ZCK homework to calculate the number of flights that cannot fly in the fall. Can you help him to calculate the answer?

Input

The first line of the input contains an integer N. The second line contains four space-separated integers l_1, l_2, r_1, r_2, where $l = \frac{l_1}{l_2}$, $r = \frac{r_1}{r_2}$. Each of the following n lines has two space-separated integers x_i, y_i indicating the position of the airport in the i-th city.

- $1 \leq N \leq 10^6$
- $-10^9 \leq l_1, r_1 \leq 10^9$
- $1 \leq l_2, r_2 \leq 10^9$
- $0 \leq x_i, y_i \leq 10^9$
- $\frac{l_1}{l_2} \leq \frac{r_1}{r_2}$
- $(x_i, y_i) \neq (x_j, y_j), \forall i \neq j$

Output

Output an integer m denoting the number of flights that cannot fly in fall.

Test Group 0 (0 %)

- Sample Input

Test Group 1 (15 %)

- $N \leq 3000$

Test Group 2 (25 %)

- $l = -10^9,\ r = 0$
- $x_i \neq x_j, \forall i \neq j$

Test Group 3 (40 %)

- $l = -10^9$
- $x_i \neq x_j, \forall i \neq j$

Test Group 4 (20 %)

- No additional constraints

Sample Input 1

```
5
-1 1 1 1
1 1
0 0
2 0
2 2
0 2
```

Sample Output 1

```
8
```

Sample Input 2

```
5
-1000000000 1 0 1
3 9
0 0
12 0
6 3
5 3
```

Sample Output 2

```
7
```

Sample Input 2

```
5
-1000000000 1 2 3
0 2
1 0
2 3
3 0
4 2
```

Sample Output 2

```
8
```

Hint

- The slope of two cities located at $(x_A, y_A), (x_B, y_B)$ is $\dfrac{y_A - y_B}{x_A - x_B}$.
- It is undefined for the slope of two cities having the same x-coordinate. Hence, the flight that connects them will always not be in the range $[l, r]$.
- It seems like **Task Group 2** is similar to the **counting inversions problem**. You may want to search for it to get further hints.

Problem 4 - ZCK Loves Apex (Programming) (15 points)

Problem Description

ZCK likes to play Apex, but for some special reasons, he doesn't want others to know he is playing. To prevent from being discovered, ZCK used to lie about where he was and what he was doing every time he played Apex, but that didn't work since he usually played Apex in public places and would be discovered by others.

As the man on top of the world, ZCK has now found a solution for this problem - teleportation! Whenever ZCK is caught not staying at the place he claimed, he can teleport there and claim that he has just come out from the bathroom.

Though ZCK can control the power of teleportation, there are still some restrictions on teleportation. Following the magic rule in this world, there is a mysterious value for each location. For every two locations x and y, you can teleport from x to y if and only if the difference between the mysterious value for that two locations is the same as the minimum mysterious value you'll pass when you're walking from x to y.

To simplify this problem, assume that ZCK lives on a straight street with N locations where he can play Apex. For the i-th location on the street, there is a mysterious value a_i. ZCK can teleport between location x and y ($x < y$) if and only if $|a_x - a_y| = \min_{x \le i \le y} a_i$. ZCK needs your help to find out how many pairs of locations are there on the street that he can teleport between.

Input

For each testcase, there would be two lines: The first line contains an integer N representing the length of the straight street. The second line contains N integers $a_1, a_2, \cdots, a_n$ representing the mysterious value for each location.

- $1 \le N \le 5 \cdot 10^5$
- $0 < a_i < 100000$

Output

For each testcase, print a single integer representing the number of pairs that ZCK can teleport between.

Test Group 0 (0 %)

- Sample Input.

Test Group 1 (40 %)

- $1 \le N \le 2000$

Test Group 2 (60 %)

- No additional constraints.

Sample Input 1

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

Sample Output 1

```
4
```

Sample Input 2

```
10
1 5 3 4 3 5 9 6 3 6
```

Sample Output 2

```
7
```

Sample Input 3

```
20
3 1 4 1 5 9 2 6 5 3 5 8 9 7 9 3 2 3 8 4
```

Sample Output 3

```
31
```

Problem 5 Asymptotic Notations (Hand-Written) (25 points)

Welcome to ADA2022! Wish you a nice trip in the world of algorithm.

For problem 5, please assume that:

$$n \in \mathbb{R}^+,$$

and also:

$$\log n \equiv \log_2 n,\ \lg n \equiv \log_2 n$$

You can also write $\log n$ and lg in your answer, which take 2 as base by default for convenience.

(a) Complexity (10 points)

Prove or disprove each of the following statements:

1. $\ln n = O(\sqrt{n})$ (1pt)
2. $928 \log_{830} n = \Theta(\ln n)$ (1pt)
3. $\Sigma_{i=1}^{n} i^3 = \Theta(n^4),\ n \in \mathbb{Z}^+$(1pt)
4. $n^{\pi}(\log n)^{112} = O(n^3\sqrt{n})$ (2pt)
5. $\lfloor \log \log n \rfloor ! = O(n)$ (2pt)

6(a). $\ln(n!) = \Theta(n \ln n),\ n \in \mathbb{Z}^+$(2pt)

6(b). prove $\ln(n!) - n \ln n + n = \Theta(\ln n)$ (1pt)

Please note that you can't use 6(b) to prove 6(a) without your proof of 6(b).

(b) Fill-in (9 points)

Please fill in the **numbers** according to their complexity in correct order.

Every pair $f(n) < g(n)$ in your answer means $f(n) = O(g(n))$.

You don't need to prove your answer in (b)!

- Points $= 9 - 0.6 \times \{(i,j) \mid i < j, answer[i] > answer[j]\}$
- **If your answer is not a permutation of [1,6], you would get 0 point from (b)!**

1. $\sqrt{n^{2.048}}$
2. $n \log(\lfloor n \rfloor !)$
3. 2^n
4. $n \log n$
5. n^k, for a large constant $k \to \infty,\ k \in \mathbb{R}^+$
6. $(\log n)^{1.0001^n}$

___ < ___ < ___ < ___ < ___ < ___

Fun fact: Every choice that is smaller than 5. is said to be "polynomially bounded".

(c) Recursion (6 points)

In this section, we assume $T(1) = 1$.

Please give the tightest upper bound for each subproblem and justify your answer with a brief proof.

1. $T(n) = 26T\left(\frac{n}{9}\right) + n^{1.5}$ (3pt)
2. $T(n) = 3n + T\left(\frac{2n}{3}\right) + T\left(\frac{2n}{9}\right)$ (3pt)

Problem 6 (Hand-Written) (25 points)

Note: In this problem, if you use any data structures or algorithms that haven't been taught in class, you need to explain what they are and prove their complexity, if needed. This problem can be solved with the knowledge taught in class. Therefore, we recommend you answer it with the knowledge taught in class.

In the United Kingdom of ADA and DSA, a pandemic called **Covy-19** spreads rapidly. To protect the residents, the king of ADA collects some **gene** from **Covy-19**. Formally, one **gene** is represented by a string with length k. Furthermore, we say two genes r, s are **similar** if and only if there exists at most one i ($1 \leq i \leq k$) such that $r[i] \neq s[i]$.

To recruit the best researchers in the kingdom, the king of ADA held a competition with up to 100,000,000 ADA (1 ADA = 0.4626 USD as I wrote this) worth of rewards. You, as an intelligent engineer, need the reward to repair your liver, so you decide to participate in this competition. Due to the lack of biological knowledge, the only tool you can rely on is a **gene comparator** that can compare two characters (you can assume that the characters have some ordering) in constant time.

As you arrive at the battlefield, the king started to announce the goal: Given n **gene**s, your **final task** is to find the number of pairs that are **similar**. Along with the words, you suddenly realized that you are teleported to a dark room.

(a) (5 points)

For the following statements, you only need to answer **T**(True) or **F**(False). No proofs or reasons need to be provided.

Assume that r, s are **similar** strings with length k and $1 \leq i(\in \mathbb{N}) < k$. Define $r_1 = r[1..i], r_2 = r[(i+1)..k]$, and define s_1, s_2 accordingly.

1. If $r_1 = s_1$, then $r_2 = s_2$
2. If $r_1 \neq s_1$, then $r_2 = s_2$
3. If $r_1 = s_1$, then $r_2 \neq s_2$
4. If $r_1 = s_1$, then r_2, s_2 are **similar**
5. If $r_1 \neq s_1$, then r_2, s_2 are **similar**

These problems are too easy for you. After you answered the questions, some **gene**s and a screen with "separation test" appeared in front of you.

(b) (5 points)

Note: according to policy, you need to prove the correctness and time complexity for (b).

Given n strings with length k, please design an $O(nk \log n)$ algorithm to divide them into groups, where

- each string in the same group are identical
- if two strings are in different group, then they are different.

You finally separated them and took some **gene** samples. For the following problems, you can assume that the n **gene**s are distinct.

(c) (5 points)

Design a **divide-and-conquer** algorithm to solve the **final task**. No correctness or complexity analysis needed. (Hint: **Problem (a) & (b)**)

This algorithm needs to run in $O(nk \log n \log k)$, although you don't need to prove it in this subproblem. Also, we recommend you to explain the algorithm in words. If you want to answer it in pseudo code, make sure it is correct, readable (comment or explain if needed), and **less than 35 lines**.

(d) (3 points)

Prove the correctness for **(c)**.

However, since you are very busy, you are not sure about how much time it will take. Therefore, you decide to analyze it first.

(e) (2 points)

Let $f(x) = x \log x$, and $a, b > 1$, prove that:

$$f(a) + f(b) \leq f(a + b)$$

(f) (5 points)

Prove that the algorithm in **(c)** runs in $O(nk \log n \log k)$.

You got the champion and is about to receive the award. Suddenly, you notice that something is wrong: you finished much faster than you expected! Although this is a great news, someone suspected that you were cheating. To prove your innocence, you use all your power in math and finally figured out the reason.
Warning: the following problem might use out-of-class knowledge and might be difficult.
Note: Although (g) is hard, you can get at most 2 partial credits if you write a correct reason why the bound in **(f)** could not be tight.
Another note: you are allowed to reference (without proof, but make sure it is correct) out-of-class theorems in subproblem (g).

(g) (bonus 5 points)

Prove that the algorithm in **(c)** runs in $O(nk \log n)$.