3rd Malaysian Computing Olympiad 2015

19 April 2014

9.30am - 12.30pm

Task	`badminton`	`honey`	`bitcoin`	`trains`	`secret`
Name	Badminton	Honey	Bitcoin	Trains	Secret
Time limit	1s	1s	1s	1s	1s
Memory limit	64MB	64MB	64MB	64MB	64MB
Points	30	10+25+35=70	15+45+50=110	8+24+41+57=130	7+16+137=160

input: standard input (*stdin*)

output: standard output (*stdout*)

Badminton

Anisa (Player A) and Ben (Player B) are playing a singles badminton match. Here are the basic badminton rules:

- To win a match, a player needs to win 2 out of 3 games.
- The first player to reach 21 points wins the game. (A different rule applies when the players are tied 20-20. You may assume this case does not happen.)
- Either player can score a point no matter who serves.
- After a game is won, a new game begins and the score resets to 0-0.

Input:

A sequence of A's and B's that records which player scored for each serve.

Output:

The score and who wins the match (A or B).

Constraints:

- Time Limit: 1s
- Memory Limit: 64MB

Subtasks:

There are no subtasks for this task. Total is 30 points.

Example 1

Input:
```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

Output:
```
21-0
21-0
A
```

Explanation:
A wins the match with scores 21-0, 21-0.

Example 2

Input:
```
BBBBBBBBBBBBBBBBBBBBBAAAAAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBB
B
```

Output:
```
0-21
21-0
0-21
B
```

Explanation:
B wins the match with scores 0-21, 21-0, 0-21.

Example 3

Input:
```
ABBBBAABABBAABABAABBBAABBBAABBBAAAAAAAABBBAABAAABBBBBBAABBBBBA
ABBBBAAABBAAABBBBAAAABBAABBBABBAABBABBBABAABAABAAA
```

Output:
```
21-18
12-21
21-19
A
```

Explanation:
A wins the match with scores 21-18, 12-21, 21-19.

Honey

Fluffy is a squirrel who likes honey very much. He lives on a tall and big tree and always collects honey from the **N** number of beehives on the tree.

One day, he notices that all the bees have gone out to work. Since it is now safe for him to collect the honey, he decides to do so. The only way for him to collect the honey is by using his **M** *ml* honey pot. On each trip, he can go to any one of the beehives and collect as much honey as he can using his honey pot. As Fluffy is a lazy squirrel, he decides not to collect honey more than **K** times.

Assume that Fluffy can determine the amount of honey $\mathbf{m_i}$ *ml* in the beehives accurately by just looking at them. Help Fluffy to calculate the maximum possible amount of honey that he can collect.

Input:

Line 1: Three positive integers: **N**, **M** and **K**.

Line 2 to (N + 1): Positive integers $\mathbf{m_i}$, one on each line.

Output:

A single integer stating the maximum possible amount of honey Fluffy can collect.

Constraints:

- Time Limit: 1s
- Memory Limit: 64MB
- **N** ≤ 200,000
- **K** ≤ 2,000,000,000
- **M** ≤ 500,000
- $\mathbf{m_i}$ ≤ 500,000

Subtasks:

Subtask 1 (10 points): **N**, **M** ≤ 10,000, **K** ≤ 100,000 and all $\mathbf{m_i}$ < **M**.

Subtask 2 (25 points): **N, M** ≤ 10,000 and **K** ≤ 100,000.

Subtask 3 (35 points): No additional constraints apply.

Example 1

Input:

```
6 10 4
11
7
5
9
3
7
```

Output:

```
33
```

Explanation:

Take 10 *ml* from the 1st hive, then 7 *ml* from the second, 9 *ml* from the 4^{th} and finally 7 *ml* from the last one. This is the maximum possible, totaling 10+7+9+7 = 33.

Example 2

Input:

```
3 10 3
13
19
4
```

Output:

```
29
```

Explanation:

Take 10 *ml* from the 1st hive, then 10 *ml* from the 2^{nd} hive and another 9 *ml* from the 2^{nd} hive, totaling 10+10+9 = 29

Bitcoin

Bitcoin mining is a very power consuming task. One day, both Ali and Betty wish to start their own mining fields (one field for each of them) in central of Cheras. Hence, Ali and Betty went to Siva, the Mayor of Cheras, to request for locations.

Siva presents Ali and Betty a grid map with the possible locations to set up their mining fields. As Bitcoin mining requires a large amount of power, Siva wants both mining fields to be situated as far as possible from each other to prevent power spikes in the local neighbourhood. Specifically, Siva wants Ali and Betty to maximize the distance between their fields.

Your task is to find the furthest Euclidean (straight line) distance between two possible mining sites given the coordinates of all mining sites. You can assume that the coordinate of a mining site is strictly an integer.

As handling floating points can be tricky (and may cause small precision errors), you are only required to output the **square** of the furthest Euclidean distance between the two possible sites for their mining fields.

The square of the Euclidean distance between two points (x_1, y_1) and (x_2, y_2) is defined as:

$$(x_1-x_2)^2 + (y_1-y_2)^2$$

Input:

Line 1: A single integer **N**, the number of possible mining sites.

Line 2: A single integer **M**, the maximum possible absolute value of the mining sites' coordinates, i.e. $-M \leq x \leq M$ and $-M \leq y \leq M$ where x and y are the coordinates.

Line 3 to (N + 2): Two integers each, coordinates $\mathbf{X_i}$ and $\mathbf{Y_i}$, corresponding to the coordinates of the possible mining sites.

Output:

A single integer corresponding to the **square** of the largest Euclidean distance.

Constraints:

- Time Limit: 1s
- Memory Limit: 64MB
- $2 \leq N \leq 1{,}000{,}000$
- $2 \leq M \leq 1{,}500$
- $-M \leq x \leq M$
- $-M \leq y \leq M$

Subtasks:

Subtask 1 (15 points): $N \leq 1{,}000$, $M \leq 1{,}000$, $x \geq 0$ and $y \geq 0$.

Subtask 2 (45 points): $N \leq 100{,}000$, $M \leq 1{,}000$, $x \geq 0$ and $y \geq 0$.

Subtask 3 (50 points): No additional constraints apply.

Example 1

Input:

```
2
15
-1 10
10 1
```

Output:

```
202
```

Explanation:
The longest distance can be obtained from the only two points in the fields: (-1, 10) and (10, 1). Their square of the Euclidean distance is $(-1-10)^2 + (10-1)^2 = 121 + 81 = 202$.

Example 2

Input:

```
3
15
1 10
2 10
10 10
```

Output:

```
81
```

Explanation:
The mining fields (1,10) and (10,10) are furthest apart. Their square Euclidean distance is $(1-10)^2 + (10-10)^2 = 81 + 0 = 81$.

Trains

MRT Corp has recently hired you, one of the most talented programmers in the country, to assist in building MRT tracks in the country.

To begin with, you are given a square grid map of size **N x N**, detailing the number of people, **C** residing in each grid segment. Since constructing your tracks would require all inhabitants of the utilised land to relocate, you are to decide on a potential railway connecting train stations **A** and **B**, with the coordinates ($\mathbf{A_x}$, $\mathbf{A_y}$) and ($\mathbf{B_x}$, $\mathbf{B_y}$) respectively, such that the number of people forced to relocate are kept to a minimum and output that number. Do note that there may be certain areas in which you cannot build your train tracks (e.g. unstable land, hills etc.).

You are also reminded that you cannot build diagonal train tracks and only build in the 4 main directions (e.g. left, right, up, down).

Input:

Line 1: An integer **N**, the length of the square grid of the map.

Line 2: Four integers $\mathbf{A_x}$, $\mathbf{A_y}$, $\mathbf{B_x}$ and $\mathbf{B_y}$ which signifies the x and y-coordinates (from top-left to bottom-right) of the two stations A and B. You will be guaranteed that the two train stations will be situated at distinct locations.

Line 3 to (**N**+2): **N** integers with a value, **C** each detailing the number of inhabitants of a particular area and separated by spaces. Areas in which you are not allowed to build train tracks on will be given a value of -1.

Output:

A single line stating the minimum number of inhabitants that will have to be relocated. Output -1 if it is impossible to construct the tracks without building on the areas in which you are not allowed to build tracks.

Constraints:

- Time Limit: 1s
- Memory Limit: 64MB
- $2 \le \mathbf{N} \le 400$
- $1 \le \mathbf{A_x}, \mathbf{A_y}, \mathbf{B_x}, \mathbf{B_y} \le \mathbf{N}$
- $-1 \le \mathbf{C} \le 1{,}000{,}000$

Subtasks:

Subtask 1 (8 points): **C** = 1.

Subtask 2 (24 points): -1 ≤ **C** ≤ 1 and **C** ≠ 0.

Subtask 3 (41 points): **N** ≤ 40.

Subtask 4 (57 points): No additional constraints apply.

Example 1

Input:

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

Output:

```
6
```

Explanation:
This example follows the restrictions of Subtask 1.

Example 2

Input:

5
3 1 5 4
1 1 1 1 1
1 1 1 1 -1
1 1 1 -1 1
1 1 -1 1 1
1 -1 1 1 1

Output:

-1

Explanation:

This example follows the restrictions of Subtask 2. Notice that it is impossible to connect the 2 stations without building on restricted land.

Example 3

Input:

5
1 3 4 5
10 **30** **20** **50** 10
10 20 -1 **25** 10
99 10 -1 **10** 10
10 10 -1 **10** 10
10 10 -1 **99** 12

Output:

363

Explanation:

The bold numbers indicate the path from (1,3) to (4,5) where the minimum number of inhabitants will have to be relocated.

Secret

Alice needs to send a secret password to Bob. The password consists of **N** space-separated integers. She decides to use a messenger, Eve, to send the password. To ensure that Eve does not steal the password, Alice uses a method of encoding she invented -- by writing it in a loop.

For example, if the password is “37 20 71 33 97”, Alice writes it down as “20 71 33 97 37”. She notifies Bob beforehand that the starting point for the message is the 5th integer, so he knows to decode the message starting from there. To make the password harder to guess, she may also use a different starting point. For example, Alice can also write the password as “71 33 97 37 20” where the starting point is the 4th integer.

Being an experienced hacker, Eve managed to figure out Alice’s encoding scheme (but not the starting points). Furthermore, Eve was Alice’s messenger twice, hence she has two of Alice’s encoded messages. Eve wishes to know whether it is possible that Alice has sent the same secret password twice.

Your task is work out whether it is possible two of those encoded messages are for the same secret password.

Input:

Line 1: A single positive integer **N** corresponding to the number of integers in the secret password.

Line 2: N space-separated positive integers $\mathbf{a_i}$ corresponding to the integers in the first encoded message.

Line 3: N space-separated positive integers $\mathbf{b_i}$ corresponding to the integers in the second encoded message.

Output:

YES or NO indicating whether it is possible that both loops are for the same password.

Constraints:

- Time Limit: 1s
- Memory Limit: 64MB
- $\mathbf{N} \le 100{,}000$
- $\mathbf{a_i}, \mathbf{b_i} \le 1{,}500{,}000{,}000$

Subtasks:

Subtask 1 (7 points): **N** ≤ 100

Subtask 2 (16 points): **N** ≤ 1,000

Subtask 3 (137 points): No additional constraints apply.

Example 1

Input:

```
5
20 71 33 97 37
71 33 97 37 20
```

Output:

```
YES
```

Example 2

Input:

```
5
20 71 33 97 37
71 33 20 97 37
```

Output:

```
NO
```

Example 3

Input:

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

Output:

```
NO
```