NovContest 2022 Problem Sheet

Ernest Kiew - for Anderson Coding Group

November 2022

Task Alpha α: Card Counting

Shor the Duck recently watched a video about how to count cards in Blackjack. Convinced that he is a genius, he decided to try it out, and immediately lost all his money doing it. (Don't gamble, kids!)

He's decided to trust you to help him win all his money back. Help him code a program that helps him to card count!

Here's how card counting works. We keep track of a count, which acts like an integer variable, which initially starts at zero. When a card of face-value 2, 3, 4, 5, or 6 is shown, we increase the count by 1. When a card of face-value 7, 8, or 9 is shown, we do nothing to the count. When a card of face-value T (10), J (Jack), Q (Queen), K (King), A (Ace) is shown, decrease the count by 1. (Note that the count can be negative)

Do note the special way that 10 is represented! (T)

Help Shor the Duck count the cards by outputting the count value after each of the N cards is played!

Input Format

Your program must read from standard input.

The input consists of $N + 1$ lines.
The first line consists of one number, N.
The next N lines contain one **character** each, C_i, which will be a number from 2-9, a T, J, Q, K, or A, representing the corresponding number card, a Ten, Jack, Queen, King or Ace respectively.

Output Format

Your program must print to standard output.

The output should consist of N lines, one after each card is added, each consisting of one number, the count after that card is added.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $2 \leq N \leq 10^5$
- $C_i \in \{2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A\}$
- In other words, the card must be one of those characters

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	28	$C_i \in \{7, 8, 9\}$, in other words the card is only 7, 8 or 9
2	72	No additional restrictions

Sample Testcase 1

This testcase is valid for subtask 1.
Input:

```
4
7
8
9
7
```

Output:

```
0
0
0
0
```

Sample Testcase 2

This testcase is valid for subtask 2.
Input:

```
6
A
2
7
T
K
3
```

Output:

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

Explanation:
The Ace decreases the count from 0 to -1, the 2 increases the count to 0, the 7 does nothing, the Ten decreases the count to -1, the King decreases the count to -2, and finally the 3 increases the count back to -1.

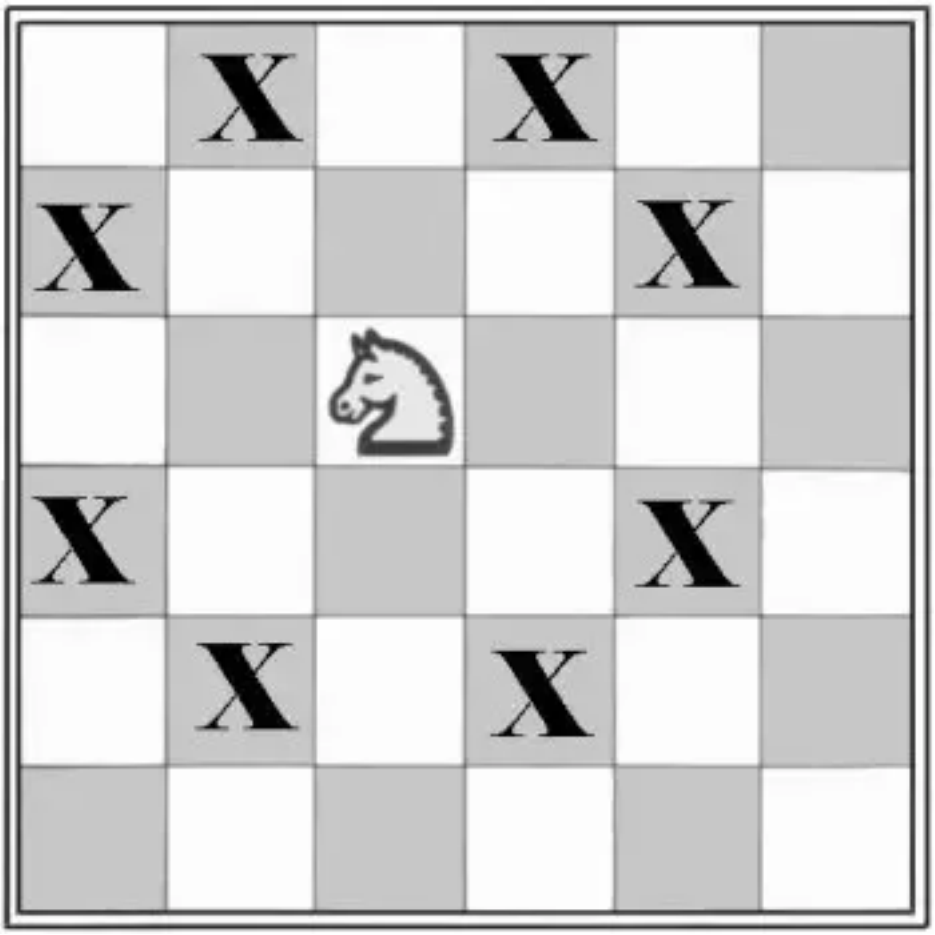
Task Beta β: Chess Champion

Magless Carsnone just became the Duck World's Duck Chess Champion, when one of his opponents, Shor the Duck showed him a new but quite boring variant. You may have heard of Lichess' Racing Kings variant. This is similar, it's called Racing Knights!

Racing knights is played with two knights on an infinite board. The two opposing knights are 8 spaces horizontally apart, and on the same horizontal line. Let's call the y-coordinate of the starting squares Y_s. The game ends when one of the knights reaches a square with the y-coordinate of Y_e. (If $Y_s = Y_e$, you may consider it as White winning with 0 moves)

As the best chess player, Magless Carsnone has already determined that White (who starts first) will always win. Make a program to calculate, for all possible games, what the minimum number of moves needed is to win the game for White? (When the knights start on the winning y-coordinate, White is considered to have won in 0 moves)

If you don't remember how the knight moves, here's a picture! The knight may move from its position to any of the squares with an X.



Input Format

Your program must read from standard input.

The input consists of one line, which contains two space-separated integers, Y_s and Y_e.

Output Format

Your program must print to standard output.

The output should consist of one line, which contains one number, the minimum number of moves needed for White to win the game.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $-10^{18} \leq Y_s, Y_e \leq 10^{18}$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	22	$-10^5 \leq Y_s \leq Y_e \leq 10^5$
2	33	$-10^5 \leq Y_s, Y_e \leq 10^5$
3	25	$0 \leq Y_s, Y_e \leq 10^{18}$
4	20	No additional constraints

Note: MODULO WORKS DIFFERENTLY WITH NEGATIVE NUMBERS. PLEASE USE abs() TO CONVERT TO POSITIVE NUMBERS OR YOU MAY SUFFER.
E.g. -3 % 2 == -1

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 3 and 4.
Input:

```
3 6
```

Output:

```
2
```

Explanation:
The knight may move 2 squares to the left and one square upwards on the first move, proceeded by moving 1 square to the right and 2 squares upwards to reach a square with the y-coordinate of 6. It can be proven that this is optimal.

Sample Testcase 2

This testcase is valid for subtasks 2 and 4.
Input:

```
-100 -200
```

Output:

```
50
```

Explanation:
The knight can make 50 moves of moving 2 squares down and one square left in a row. It can be proven that this is optimal.

Task Gamma γ: Desperate Developments

Shor the Duck is building a nation! (yes, for the fifteenth time!) This time, he needs to house his many citizens somewhere, as the current housing is unaffordable. He has decided to build government-funded housing!

Shor the Duck has land that can be represented as a straight line of N plots. He will pick a segment of the plots to reserve in a range of $[l, r]$ inclusive to build on. To prevent overcrowding, buildings will only be built on plots l, $l+2$, $l+4$, $l+6$, and so on, but never exceeding r.

Furthermore, each plot of land numbered i from left to right starting from 0 has its own quality Q_i. Thus, Shor the Duck wants to maximise the sum of qualities of the land used to build the buildings!

Help Shor to find out what the maximum possible sum of qualities of land used for building is!

Input Format

Your program must read from standard input.

The input consists of 2 lines.
The first line consists of one integer, N.
The second line consists of N space-separated integers, denoting the value of Q_i for each value of i.

Output Format

Your program must print to standard output.

The output should consist of 1 line, the maximum possible sum of qualities of land used for building after he picks a certain segment.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq N \leq 10^5$
- $0 \leq Q_i \leq 10^{12}$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	13	$N = 1$
2	34	$N \leq 100$
3	18	$0 \leq Q_i \leq 10^3$
4	35	No additional restrictions

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 3, 4.
Input:

```
1
9
```

Output:

```
9
```

Explanation: Shor the Duck may pick the segment [0, 0] (0-indexed), and build on plot 0, to result in a maximum sum of qualities of 9.

Sample Testcase 2

This testcase is valid for subtasks 2, 3, 4.
Input:

```
5
1 8 7 4 2
```

Output:

```
12
```

Explanation:
Shor the Duck may pick segment [1, 3] (0-indexed), resulting in buildings on plots 1 and 3 only. Even though the segment [0, 4] has more plots for building, this segment [1, 3] gives a sum of qualities of 12, and it can be proven that this is optimal.

Task Delta δ: Fantastic Feast

Great news! Gordon, the legendary chef (who is also a duck) is coming to town! He has prepared a menu of N different food items, and in preparation, you've noted down how much you think you'd like each item, H_i. Furthermore, because of both financial and stomach-capacity related reasons, you can only buy up to M dishes.
You are allowed to buy multiple of the same dish, but you would be more satisfied if you were able to try many different dishes. Specifically, your happiness can be represented as the sum of H_i of the dishes you've bought, and an additional K happiness for every unique dish tried.
Naturally, you'd like to maximise the amount of happiness gained, so find out, given N, M, K and each value of H_i, what the maximum happiness attainable is!

Input Format

Your program must read from standard input.

The input consists of 2 lines.
The first line consists of three space-separated integers, N, M, and K.
The second line consists of N space-separated integers, denoting the value of H_i for each dish.

Output Format

Your program must print to standard output.

The output should consist of 1 line, with one integer, the maximum possible happiness at the end.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $2 \leq N \leq 10^5$
- $1 \leq M \leq 10^6$
- $0 \leq K \leq 10^8$
- $0 \leq H_i \leq 10^8$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	15	$K = 0$
2	16	$H_i = 0$
3	14	$N, M \leq 10^3$
4	23	$N \leq 10^3$
5	32	No additional restrictions

Sample Testcase 1

This testcase is valid for subtasks 1, 3, 4 and 5.
Input:

```
6 3 0
9 2 6 2 1 3
```

Output:

```
27
```

Explanation:
You may pick the first dish thrice, giving you a happiness value of 9+0 + 9+0 + 9+0 = 27.

Sample Testcase 2

This testcase is valid for subtasks 2, 3, 4 and 5.
Input:

```
6 4 3
0 0 0 0 0 0
```

Output:

```
12
```

Explanation:
You may pick dish 2, 5, 3 and 6. This gives you a happiness score of 12, as you have tried 4 unique dishes and gained 0 from the dishes themselves.

Sample Testcase 3

This testcase is valid for subtasks 3, 4 and 5.
Input:

```
6 7 4
6 5 3 8 10 3
```

Output:

```
76
```

Explanation:
You may pick dish 5 six times, and pick dish 4 once. This gives a value of 68, and since you tried 2 unique dishes, you obtain a value of 76. It can be proven that this is optimal.

Task Epsilon ϵ: Treacherous Thievery

You are a duck lawyer with an extreme passion for justice. So much so, that when you realised that the multinational corporation, Duck Co. was hiding a very dark secret, you decided to steal it.

You've made your way through most of the facility containing the safe with the secret. However, you are confronted with your greatest obstacle yet. A massive room with N locks! To get past this obstacle, you must open up at least K locks. With no way to get the keys, you decide to become a lockpicker. In other words, you are now a lock-picking lawyer.

Furthermore, you are only given access to the first M locks at first, meaning the others are restricted and cannot be opened. Opening up one lock unrestricts the next lock, which will be the first lock in the list that is still restricted, until there are no more locks that are restricted. (The unrestriction happens in order of the input.)

To complicate things even further, each lock has a different difficulty rating, D_i. A lock with a difficulty rating of D_i will take D_i seconds to unlock, and you may only unlock one lock at a time. Of course, you'd love to get the secret and get out of the facility as soon as possible. Thus, what's the minimum time (in seconds) needed to unlock K locks? You may assume it is always possible.

Input Format

Your program must read from standard input.

The input consists of 2 lines.
The first line consists of three space-separated integers, N, M, and K.
The second line consists of N space-separated integers, denoting the value of D_i for each lock.

Output Format

Your program must print to standard output.

The output should consist of 1 line, with one integer, the minimum amount of time needed.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq K, M \leq N \leq 10^5$
- $0 \leq D_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	7	$K = N$
2	17	$K = 1$
3	26	$M = N$
4	23	$N \leq 10^3$
5	27	No additional restrictions

Sample Testcase 1

This testcase is valid for subtasks 1, 4 and 5.
Input:

```
6 3 6
9 2 6 2 1 3
```

Output:

```
23
```

Explanation:
He should pick the second lock, first lock, third lock, fourth lock, fifth lock, then sixth lock in that order. This will take 23 seconds. Other solutions exist that will give the same optimal minimum time.

Sample Testcase 2

This testcase is valid for subtasks 2, 4 and 5.
Input:

```
6 3 1
8 6 4 2 8 3
```

Output:

```
4
```

Explanation:
He should pick the third lock. This fulfils the requirement of picking one lock, and can be proven to be the optimal solution. Note that he may not pick the fourth lock as it is still restricted.

Sample Testcase 3

This testcase is valid for subtasks 3, 4 and 5.
Input:

```
6 6 4
6 5 10 8 3 12
```

Output:

```
22
```

Explanation:
He should pick lock 4, lock 2, lock 5, lock 1 in that order. It can be proven that 22 is the minimum time needed to pick 4 locks in this case.

Sample Testcase 4

This testcase is valid for subtasks 4 and 5.
Input:

```
6 3 4
6 5 10 2 3 12
```

Output:

```
16
```

Explanation:
He should pick lock 1, lock 2, lock 4, then lock 5 in that order. It can be proven that 16 is the minimum amount of time needed to pick 4 locks.

Task Zeta ζ: Bundle Bee

You are a humble duck beekeeper, keeping track of your bumblebees. These are special bees however; They have been equipped with bundles! Although they do not produce honey, you love these bumblebees just as much, and want them to prosper!

You are given the layout of N flowers, where the flowers are arranged in a line from left to right, each having a nectar value of V_i (and it is always positive). A bundling bumblebee may fly anywhere to start the collection process without losing energy. However, the bundling bumblebee can only collect nectar from a contiguous segment of flowers, and must collect all the nectar from the flowers in the segment.

As the bundling bumblebee might tire out, it can only collect from segments that are at most length L, and to prevent wastage of precious nectar, the bee cannot collect more than C units of nectar, which is the maximum capacity of the bundle. If the chosen segment's total nectar amount exceeds C or has more than L elements, the segment is invalid and the bee cannot choose this segment. The bee is not allowed to collect extra nectar then discard it, as it doesn't want to be wasteful.

Thus, help the bee find out what the maximum total nectar it can collect is!

Input Format

Your program must read from standard input.

The input consists of 2 lines.
The first line consists of three space-separated integers, N, L, and C.
The second line consists of N space-separated integers, denoting the value of V_i for each flower.

Output Format

Your program must print to standard output.

The output should consist of 1 line, with one integer, the maximum amount of nectar it can collect!

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq L \leq N \leq 10^5$
- $1 \leq C \leq 10^{15}$
- $0 \leq V_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	6	$L = N$, $C = 10^{15}$
2	18	$C = 10^{15}$, $N \leq 10^3$
3	19	$C = 10^{15}$
4	32	$L = N$
5	25	No additional restrictions

Sample Testcase 1

This testcase is valid for subtasks 1, 2, 3, 4 and 5.
Input:

```
6 6 1000000000000000
9 2 6 2 1 3
```

Output:

```
23
```

Explanation:
The bee should pick the segment [0, 4] inclusive. (0-indexed)

Sample Testcase 2

This testcase is valid for subtasks 2, 3, 4 and 5.
Input:

```
6 3 6
8 6 4 2 8 3
```

Output:

```
6
```

Explanation:
The bee should pick the segment [2, 3] inclusive.

Sample Testcase 3

This testcase is valid for subtasks 4 and 5.
Input:

```
6 6 22
6 7 10 8 3 12
```

Output:

```
21
```

The bee should pick the segment [2, 4] inclusive.

Sample Testcase 4

This testcase is valid for subtask 5.
Input:

```
6 2 12
6 5 2 7 3 13
```

Output:

```
11
```

Explanation:
The bee should pick the segment [0, 1] inclusive. The segment [2, 4] is invalid as it has too many elements, and the segment [5, 5] is invalid as it exceeds the capacity.

Task Eta η: Tempting Treats

Shor the Duck is easily tempted by treats. He has N muffins in the fridge (don't ask how or why) and knows he shouldn't eat them all. Through flawed logic, he has decided that as long as he doesn't eat all the muffins, he should be fine. (He probably won't be)

Shor the Duck will start off with a desire value of D, and each muffin will have a temptation value of T_i, and an addiction value of A_i. Shor the Duck will automatically consume any muffin with a temptation value T_i that is less than or equal to his current desire value of D. Furthermore, after the consumption of the ith muffin, his value of D increases by A_i.

Help Shor the Duck find out what the largest value of D he can have initially such that he doesn't consume **all** of the muffins! (D cannot be negative)

Input Format

Your program must read from standard input.

The input consists of 3 lines.
The first line consists of two space-separated integers, N, D.
The second line consists of N space-separated integers, denoting the value of T_i for each muffin.
The third line consists of N space-separated integers, denoting the value of A_i for each muffin.

Output Format

Your program must print to standard output.

The output should consist of 1 line, with one integer, the greatest value of D such that he will not consume all of the muffins.

Subtasks

The maximum execution time on each instance is 1.0s, and the maximum memory usage on each instance is 512MB. For all test cases, the input will satisfy the following bounds:

- $1 \leq N \leq 10^5$
- $1 \leq T_i \leq 10^{12}$
- $0 \leq A_i \leq 10^9$

Your program will be tested on input instances that satisfy the following restrictions:

Subtask	Marks	Additional Constraints
1	11	$min(A_i) > max(T_i)$
2	18	$A_i = 0$
3	28	$T_i \leq 100$
4	43	No additional restrictions

Sample Testcase 1

This testcase is valid for subtasks 1, 3, and 4.
Input:

```
4
3 2 8 9
12 12 13 13
```

Output:

```
1
```

Explanation:
When Shor has a desire value of 1, he eats no muffins. It can be proven that this is the maximum desire he can have without eating all muffins.

Sample Testcase 2

This testcase is valid for subtasks 2, 3 and 4.
Input:

```
5
3 7 9 2 3
0 0 0 0 0
```

Output:

```
8
```

Explanation:
When Shor has a desire value of 8, he eats all muffins except the third one. It can be proven that this is the maximum desire he can have without eating all muffins.

Sample Testcase 3

This testcase is valid for subtasks 3 and 4.
Input:

```
5
1 3 5 7 9
1 2 1 3 10
```

Output:

```
2
```

Explanation:
When Shor has a desire value of 2, he eats the first muffin, then has a desire value of 3, then eats the second muffin, then has a desire value of 5, then eats the third muffin and has a desire of 6. No more muffins are eaten, and it can be proven that 2 is the maximum desire value he can have without eating all muffins.