WOBURN CHALLENGE

2017-18 Online Round 1

Friday, January 26th, 2018

Senior Division Problems

Automated grading is available for these problems at:
wcipeg.com

For more problems from past contests, visit:
woburnchallenge.com

Problem S1: On the Rocks

13 Points / Time Limit: 2.00s / Memory Limit: 16M *Submit online: http://wcipeg.com/problem/wc171s1*

Two teams have just finished playing a riveting round of Canada's national sport, curling, and it's time to tally up their scores! Team A has N ($0 \le N \le 8$) stones in play, with the i-th of them at a distance of A_i ($0 \le A_i \le 370$) cm away from the "button" (the centre of the scoring ring). Meanwhile, Team B has M ($0 \le M \le 8$) stones, with the i-th of them at a distance of B_i ($0 \le B_i \le 370$) cm away from the button. No two stones are equidistant from the button.

If there are no stones in play at all, neither team will score any points. Otherwise, only the single team which owns the closest stone to the button will score points. That team will score 1 point for each of their stones which is closer to the button than all of the other team's stones are. If the other team doesn't even have any stones in play, then each of the scoring team's stones counts for a point.

Please help tally up the two teams' final scores! Note that at least one of these two scores must be equal to 0.

Subtasks

In test cases worth 3/13 of the points, $N = 1$ and $M = 1$.

Input Format

The first line of input consists of two space-separated integers, N and M.

The next line consists of integers, $A_{1..N}$.

The next line consists of integers, $B_{1..M}$.

Output Format

Output two space-separated integers, the number of points scored by Teams A and B respectively.

Sample Input

```
2 4
205 44
33 146 14 45
```

Sample Output

```
0 2
```

Sample Explanation

Team B owns the closest stone to the button (their 3rd one), so they'll be the team scoring some points. In particular, their 1st and 3rd stones will count for 1 point each. On the other hand, Team B's 2nd and 4th stones won't count for any points, as they're further from the button than Team A's 2nd stone is.

Problem S2: Ride the Rocket

22 Points / Time Limit: 2.00s / Memory Limit: 64M *Submit online: http://wcipeg.com/problem/wc171s2*

A certain TTC bus route involves a sequence of N ($2 \le N \le 10^9$) bus stops, numbered from 1 to N.

Every P ($1 \le P \le 100$) minutes, starting at time 0, a new bus will arrive at stop 1. It will wait a brief moment to allow new passengers to board and/or existing passengers to disembark, and then proceed onwards to stop 2, reaching it after another B ($1 \le B \le 100$) minutes. There, it will similarly give passengers an opportunity to board/disembark, before continuing onwards and reaching stop 3 after another B minutes, and so on. In this manner, $B \times (N - 1)$ minutes after any given bus starts its route, it will arrive at stop N, drop off any remaining passengers, and then go out of service. Note that a new bus drives along the route described above every P minutes, meaning that there may be multiple buses on the road at any time.

Each bus has a maximum capacity of C ($1 \le C \le 10^5$) passengers. If some passengers want to get off a bus while others simultaneously want to get on it, the former can happen first to make room for the new passengers. Note that buses take no extra time to pick up or drop off any number of passengers at a stop.

At time 0, your entire class of M ($1 \le M \le 10^5$) students is waiting at stop 1, the i-th of whom wants to get to stop D_i ($2 \le D_i \le N$) as quickly as possible. At any moment, each student not currently on a bus may either wait at their current stop, walk to the next stop along the route in W ($1 \le W \le 100$) minutes, or board a bus (if there's a below-capacity bus at their current stop at that moment). Meanwhile, each student currently on a bus may get off the bus if it's currently at a stop.

Each student's "travel time" is the amount of time which goes by (after time 0) before they arrive at their destination stop. You're thinking it would be nice if the sum of all M students' travel times could be as small as possible. As such, you'd like to determine the minimum possible value this sum could have, assuming that all of the students work together to minimize it. Though, on second thought, getting everyone in your class to cooperate might be the more difficult part…

Please note that the answer may not fit into a 32-bit signed integer.

Subtasks

In test cases worth 4/22 of the points, $N \le 10$ and $M \le 10$.

In test cases worth another 11/22 of the points, $N \le 10^5$ and $M \le 1000$.

Input Format

The first line of input consists of four space-separated integers, N, P, B, and C.
The next line consists of two space-separated integers, M and W.

M lines follow, the i-th of which consists of a single integer, D_i, for $i = 1..M$.

Output Format

Output a single integer, the minimum possible sum of travel times for all M students to reach their desired stops.

Sample Input 1

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

Sample Output 1

```
11
```

Sample Input 2

```
10 3 1 2
4 2
4
3
5
4
```

Sample Output 2

```
17
```

Sample Explanation 2

In the first case, one optimal strategy is as follows:

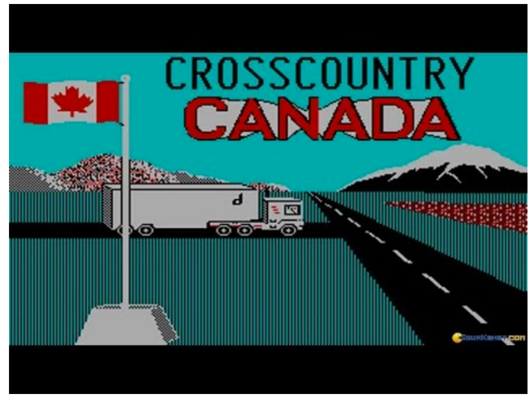
Student 1: Board the first bus immediately, and disembark at stop 2 (2 minutes)
Student 2: Wait for 2 minutes, board the second bus at time 2, and disembark at stop 2 (4 minutes)
Student 3: Walk to stop 2 (5 minutes)

In the second case, one optimal strategy is as follows:

Student 1: Board the first bus immediately, and disembark at stop 4 (3 minutes)
Student 2: Walk all the way to stop 3 (4 minutes)
Student 3: Board the first bus immediately, and disembark at stop 5 (4 minutes)
Student 4: Walk to stop 2, wait for 2 minutes, board the second bus at time 4, and disembark at stop 4 (6 minutes)

Problem S3: Crosscountry Canada

28 Points / Time Limit: 2.00s / Memory Limit: 64M *Submit online: http://wcipeg.com/problem/wc171s3*



There are N ($2 \le N \le 1000$) cities, with M ($0 \le M \le 10{,}000$) roads running amongst them. The i-th road connects two different cities A_i and B_i ($1 \le A_i, B_i \le N$), and can be driven along in either direction in C_i ($1 \le C_i \le 100$) minutes. There may be multiple roads running directly between any given pair of cities.

You're taking a roadtrip across Canada from city 1 to city N. You'd like to reach your destination in as little time as possible, by following a sequence of roads from city to city. However, as every Canadian knows, Tim Horton's pit stops are an essential part of any trip. It's vital that you stop at a Tim Horton's every L ($1 \le L \le 100$) minutes or less. In particular, you must never spend strictly more than L consecutive minutes during your trip outside of Tim Hortons' establishments, not counting any time before you leave city 1 or after you arrive at city N.

Somehow, not every city has a Tim Horton's! If $R_i = 0$, then the i-th city doesn't have one, and if $R_i = 1$, then it does ($0 \le R_i \le 1$). Whenever you arrive in a city which has a Tim Horton's, you may choose to stop at it before continuing on your trip, which takes T ($1 \le T \le 100$) minutes.

What's the minimum amount of time required to reach city N from city 1 without ever spending more than L consecutive minutes outside of Tim Horton's establishments? If it can't be done, output -1 instead.

Subtasks

In test cases worth 7/28 of the points, $L = 1$ and $C_i = 1$ (for $i = 1..M$).

In test cases worth another 10/28 of the points, $C_i = 1$ (for $i = 1..M$).

Input Format

The first line of input consists of four space-separated integers, N, M, L, and T.

The next line consists of integers, $R_{1..N}$.

M lines follow, the i-th of which consists of three space-separated integers, A_i, B_i, and C_i, for $i = 1..M$.

Output Format

Output a single integer, the minimum amount of time required to validly reach city N from city 1, or –1 if it's impossible.

Sample Input 1

```
6 10 6 3
0 1 0 1 0 0
1 3 3
1 4 6
1 4 7
2 4 2
2 5 4
2 6 3
3 4 6
4 5 1
4 6 6
5 6 5
```

Sample Output 1

```
14
```

Sample Input 2

```
2 1 10 1
1 1
2 1 11
```

Sample Output 2

```
-1
```

Sample Explanation 2

In the first case, one optimal route is as follows:

1 → 4 (6 mins) Stop at Tim Horton's (3 mins) 4 → 2 (2 mins) 2 → 6 (3 mins)

In the second case, the single road between the cities is just too long for the trip to be possible, as driving along it would result in a lack of Tim Horton's for more than 10 minutes.

Problem S4: Change

37 Points / Time Limit: 2.00s / Memory Limit: 64M *Submit online: http://wcipeg.com/problem/wc171s4*

Your friend is just getting into competitive programming, and is trying to solve the classic "change" problem. You'll give him a target value K ($1 \le K \le 10^9$) and some distinct coin denominations (each of which is between 1 and K, inclusive), and he'll try to determine whether or not a total of K can be produced using a set of (possibly duplicate) coins having only those denominations. The algorithm he's come up with to do so is greedy - starting from an empty set of coins, it repeatedly adds on the largest possible coin which won't cause the total to exceed K, until it either reaches a total of K, or is unable to add on any more coins and gives up.

You're not sure what coin denominations you'd like to give him, but there are N ($0 \le N \le 2000$) distinct denominations which you definitely don't want to include. The i-th of these is D_i ($1 \le D_i \le K$).

Your friend is convinced that his so algorithm is so good that he'll be able to attain a total of K no matter what denominations he's given! This is quite the bold claim. You'd like to prove him wrong by choosing a (possibly empty) set denominations such that his algorithm will fail to reach a total of K using them. Note that it doesn't matter whether or not a more correct algorithm would succeed, as long as your friend's greedy algorithm fails to achieve a total of K.

Clearly, one option would be to simply give your friend 0 denominations to use. However, that's too easy - you'd like to prove as convincingly as possible that their algorithm is sub-par. As such, you'd like to determine the maximum number of distinct denominations which you can give your friend, such that their algorithm will still fail to reach a total of K using them.

Subtasks

In test cases worth 6/37 of the points, $K \le 20$.
In test cases worth another 20/37 of the points, $K \le 1000$.

Input Format

The first line of input consists of two space-separated integers, K and N.
N lines follow, the i-th of which consists of a single integer, D_i, for $i = 1..N$.

Output Format

Output a single integer, the maximum number of distinct denominations which you can give your friend.

Sample Input 1

```
7 4
3 7 6 1
```

Sample Output 1

```
2
```

Sample Input 2

```
4 1
3
```

Sample Output 2

```
0
```

Sample Explanation 2

In the first case, one possibility is to give your friend the set of denominations (2, 4}. His algorithm would use a 4 coin followed by a 2 coin, and then give up.

In the second case, you must resort to giving your friend no denominations, as his algorithm would achieve a total of 4 if given any non-empty subset of the denominations {1, 2, 4}.