Homework #2

Due Time: 2020/11/10 14:20
Contact TAs: `ada-ta@csie.ntu.edu.tw`

Instructions and Announcements

- There are **four programming problems** and **two hand-written problems**.
- **Programming.** (55 pt + 5 bonus) 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. **Note that if you got more than 55 points in programming part, the score would be reduce to 55 points.** NO LATE SUBMISSION IS ALLOWED.
- **Hand-written.** (50 pt) For other problems (also known as the "hand-written problems"), you should upload your answer to **Gradescope** as demonstrated in class. Please **briefly** explain your solution in this part. 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 or comment in code of your solution to that problem. You may get zero point due to the lack of references.

Problem 1 - Tower Defense (Programming) (10 points)

> Don't hold on to the past; it won't help in moving forward.
>
> Rajeev Suri

Problem Description

It's game time again! This time you have to build defense towers to protect your castle from foreign attacks.

The playfield can, again, be seen as a one-dimensional grid containing N cells numbered from 1 to N. Since the terrain of each cell varies, some of them may be well suited to build a defense tower on, while at the same time, towers built on some other cells can be useless. We denote the *suitability* of the i^{th} cell by s_i. **Note that s_i may be negative.**

For each cell, you can decide whether to build a defense tower on it or not, but the budget you have is only enough to build K towers. Suppose you build t towers on some t cells, where $0 \leq t \leq K$. We denote the cells you choose by a sequence of t indices $1 \leq x_1 < x_2 < \cdots < x_t \leq N$. The defense level of this arrangement is

$$\sum_{i=1}^{t} s_{x_i},$$

which is the sum of the suitability of the chosen cells.

Moreover, it will be harder for your enemy to sneak into your castle if you build the towers closer to each other. More precisely, each valid index i has a bonus defense level of

$$A \cdot \max\{0, B - (x_{i+1} - x_i)\},$$

where both A and B are positive integers. In other words, if you build two towers close enough, you receive a bonus that is negative linear to the distance between these two towers.

Now, your task is to plan out an arrangement wisely to maximize the total defense level.

Input

The first line of the input contains four integers N, K, A and B, indicating the length of the field, the budget, and the constants used in the bonus formula, respectively. The second line of the input contains N space-separated integers $s_1, s_2, \ldots, s_N$, where the i^{th} integer denotes the suitability of the i^{th} cell.

- $1 \leq B \leq N \leq 10^5$
- $1 \leq K \leq 400$
- $1 \leq A \leq 10^9$
- $-10^9 \leq s_i \leq 10^9$

Test Group 0 (0 %)

- Sample Input

Test Group 1 (10 %)

- $N \leq 1000$
- $K \leq 2$

Test Group 2 (30 %)

- $N \leq 1000$
- $K \leq 25$

Test Group 3 (10 %)

- $B = 1$

Test Group 4 (10 %)

- $B = N$

Test Group 5 (20 %)

- $K \leq 25$

Test Group 6 (20 %)

- No additional constraints

Output

Output a single integer representing the maximum total defense level.

Sample Input 1

```
5 3 9 3
4 -1 -11 6 5
```

Sample Output 1

```
37
```

Sample Input 2

```
3 2 10000 1
1 2 7
```

Sample Output 2

```
9
```

Sample Input 3

```
2 2 6 2
10 -30
```

Sample Output 3

```
10
```

Sample Input 4

```
8 4 4 4
2 15 -12 14 9 14 11 -10
```

Sample Output 4

```
84
```

Sample Input 5

```
12 4 6 2
11 13 6 -6 -8 6 -6 14 8 6 1 -10
```

Sample Output 5

```
58
```

Hint

- Try to figure out what the constraints of each test group mean.
- Oops! We're running out of space. Maybe the quote at the beginning can help you.
  - Note that *Runtime Error* may be an indication of *Memory Limit Exceeded.*
- You can solve this problem without using anything too fancy.

Problem 2 - ADA Sequence (Programming) (15 points)

Problem Description

YP and BB love sequences very much. They define a sequence $b_1, b_2, \ldots, b_Y$ of a positive length Y as an ADA sequence if for all $i \in [1, Y]$, the following condition holds:

$$|b_i - b_{Y-i+1}| \leq D,$$

where D is a given parameter.

Now, BB gives YP three N-length sequences $a_1, a_2, \ldots, a_N$; $c_1, c_2, \ldots, c_N$, and $s_1, s_2, \ldots, s_N$. Also, BB will give YP K dollars and decides to play a game with YP!

YP can do the following operations at most 1000 times (with zero times being possible):

- Select an integer $i \in [1, N]$.
- Spend c_i dollars to change a_i into an arbitrary value.

Note that the total amount of money YP spends on the operations should not exceed K.

After finishing the operations, YP needs to choose some indices $b_1, b_2, \ldots, b_M$. BB defines the indices **valid** if the following conditions are satisfied:

- $M > 0$
- $1 \leq b_1 < b_2 < \cdots < b_{M-1} < b_M \leq N$
- $a_{b_1}, a_{b_2}, \ldots, a_{b_M}$ is an ADA sequence.

Also, BB defines the **score** of the ADA sequence as $s_{b_1} + s_{b_2} + \cdots + s_{b_M}$.

Please help YP find out the maximum possible score, and tell YP how to do the operations and select $b_1, b_2, \ldots, b_M$ for forming the ADA sequence.

Input

The first line of the input contains three integers N, K, and D, as defined in the problem description.

The second line of the input contains N integers $a_1, a_2, \ldots, a_N$.

The third line of the input contains N integers $c_1, c_2, \ldots, c_N$.

The fourth line of the input contains N integers $s_1, s_2, \ldots, s_N$.

Test Group 0 (0 %)

- Sample Input

Test Group 1 (10 %)

- $1 \leq N \leq 200$
- $K = 0$
- $D = 0$
- $1 \leq a_i \leq 10^9$
- $1 \leq c_i \leq 600$
- $s_i = 1$

Test Group 2 (30 %)

- $1 \leq N \leq 200$
- $K = 0$
- $D = 0$
- $1 \leq a_i \leq 10^9$
- $1 \leq c_i \leq 600$
- $1 \leq s_i \leq 3$

Test Group 3 (30 %)

- $1 \leq N \leq 200$
- $0 \leq K \leq 600$
- $0 \leq D \leq 10^9$
- $1 \leq a_i \leq 10^9$
- $1 \leq c_i \leq 600$
- $1 \leq s_i \leq 3$

Test Group 4 (30 %)

- $1 \leq N \leq 200$
- $0 \leq K \leq 10^9$
- $0 \leq D \leq 10^9$
- $1 \leq a_i \leq 10^9$
- $1 \leq c_i \leq 10^9$
- $1 \leq s_i \leq 3$

Output

Please output $X + 4$ lines.

The first line contains one integer S denoting the maximum possible score.

The second line contains one integer X denoting the number of operations you want to do.

In the following X lines, the i-th line contains two integers pos_i, val_i. This indicates that in the i-th operation, YP changes a_{pos_i} to val_i.

In the next line, output an integer M denoting the length of the sequence b.

In the next line, output M integers denoting the indices YP chooses.

The output must satisfy the following restrictions:

- $0 \leq X \leq 1000$
- $1 \leq pos_i \leq N$
- $1 \leq val_i \leq 10^9$
- $\sum_{i=1}^{X} c_{pos_i} \leq K$
- $1 \leq M \leq N$
- $1 \leq b_i \leq N$
- $b_i < b_{i+1} \forall i \in [1, M-1]$
- After doing the K operations, $a_{b_1}, a_{b_2}, \ldots, a_{b_M}$ becomes an ADA sequence.
- $s_{b_1} + s_{b_2} + \ldots s_{b_M} = S$ is maximized.

If there are multiple solutions, you can output any of them.

It is guaranteed that there exists a solution satisfying the above restrictions.

Sample Input 1

```
6 0 0
4 2 3 2 3 4
1 1 1 1 1 1
1 1 1 1 1 1
```

Sample Output 1

```
5
0
5
1 3 4 5 6
```

Sample Input 2

```
8 0 0
8 3 1 3 4 4 1 8
1 1 1 1 1 1 1 1
1 3 1 3 1 1 1 1
```

Sample Output 2

```
9
0
5
1 2 3 4 8
```

Sample Input 3

```
6 3 1
4 8 17 1 6 12
1 2 2 3 1 2
1 3 1 2 2 1
```

Sample Output 3

```
9
2
1 12
5 8
5
1 2 4 5 6
```

Sample Input 4

```
10 1258 3
417 118 413 213 116 817 200 154 177 465
765 872 548 874 254 654 966 553 398 698
1 3 2 3 1 3 3 1 1 3
```

Sample Output 4

```
16
3
3 465
9 213
5 200
7
3 4 5 6 7 9 10
```

Hint

1. Solving the test groups in order (from 1 to 3) is helpful for solving Test Group 4.
2. *Longest Common Subsequence* (LCS), *Longest Increaing Subsequence* (LIS), and *Longest Palindromic Subsequence* (LPS) are classic dynamic programming problems.
3. GL & HF (Good Luck and Have Fun).

Problem 3 - Boook Arrangement (Programming) (15 points)

Problem Description

Boook is a librarian, and arranging books to make them look beautiful is his hobby. Boook thinks that a sequence of books is beautiful if the thickness of adjacent books differs by 1. Formally, if the thickness of the books in the sequence are $x_1, x_2, \ldots, x_k$, respectively, then it is beautiful if and only if $|x_i - x_{i+1}| = 1$ for all $i = 1, 2, \ldots, k-1$.

One day, Boook receives some books. More specifically, there are c_i books with the thickness i, for all $i = 1, 2, \ldots, n$. Boook would like to make them beautiful by arranging them into a beautiful sequence. However, he finds that sometimes doing so is impossible, so he wants to insert the least number of additional books with thickness also in the range $[1, n]$ so that he can make these books beautiful. Please help Boook complete the mission.

Input

The first line contains a positive integer n ($2 \le n \le 10^5$), which is mentioned in the description.

The second line contains n non-negative integers. The i-th integer c_i ($0 \le c_i \le 10^6$) represents the number of books with thickness i Boook initially has. It is guaranteed that at least one c_i is non-zero.

The third contains only one integer $flag$ ($flag \in \{0, 1\}$), indicating whether you should print an beautiful arrangement of the books or not.

It is guaranteed that the size of the output is less than 10^7 bytes.

Output

In the first line of the output, print an integer indicating the least number of additional books Boook needs to insert. Furthermore, if $flag = 1$ in the input, print a beautiful arrangement of the books using the least number of additional books in the second line. If there are multiple ways to arrange the books, you may print any of them.

Test Group 0 (10 %)

- Sample Input

Test Group 1 (10 %)

- $flag = 0$
- $c_i = 0, \forall\ 2 \mid i$, or $c_i = 0, \forall\ 2 \nmid i$

Test Group 2 (10 %)

- $c_i = 0, \forall\ 2 \mid i$, or $c_i = 0, \forall\ 2 \nmid i$

Test Group 3 (10 %)

- $flag = 0$
- $n \le 4$

Test Group 4 (10 %)

- $n \le 4$

Test Group 5 (20 %)

- Boook does not need to insert the additional books, that is, the first line of the output is 0

Test Group 6 (20 %)

- $flag = 0$

Test Group 7 (20 %)

- No other constraints

Sample Input 1

```
4
1 1 2 1
1
```

Sample Output 1

```
0
3 4 3 2 1
```

Sample Input 2

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

Sample Output 2

```
0
1 2 3 2 1 2 3 4 3 2
```

Sample Input 3

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

Sample Output 3

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

Sample Input 4

```
7
2 2 4 2 5 3 1
1
```

Sample Output 4

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

Problem 4 - Segments (Programming) (15 points + 5 points bonus)

Problem Description

Given N segment sets, each of them is located between $[0, M]$. The i-th segment set can be represented by (L_i, R_i, W_i), indicating that there are W_i segments covering the range $[L_i, R_i]$.

For any chosen subset of the segments and a real number x, we define the function $f(x)$ as the number of chosen segments that cover the position x. Your goal is to pick a subset S containing at most K segments that maximizes

$$V = \min_{0 \le x \le M} f(x).$$

Input

The first line contains three space-separated integers N, M and K, which represent the number of segment sets, the range of the segments, and the number of segments you can select respectively.

Each of the following N lines contains three space-separated integers L_i, R_i, and W_i, indicating the location of the segments and the number of segments in the i-th segment set.

- $0 \le K \le \sum W_i$
- $1 \le N, M \le 2 \times 10^5$
- $0 \le L_i < R_i \le M$
- $1 \le W_i \le 3 \times 10^{13}$

Test Group 0 (0 %)

- Sample Input

Test Group 1 (15 %)

- $K = 2$
- $W_i = 1$

Test Group 2 (25 %)

- $K = N$
- $W_i = 1$

Test Group 3 (25 %)

- $1 \le N, M \le 1000$
- $W_i = 1$

Test Group 4 (25 %)

- $1 \le N, M \le 10^5$

Test Group 5 (10 %)

- No other constraints

Output

In the first line of the output, print an integer indicating the maximum V.

In the second line, print N integers, where the i-th integer represents the number of segments you selected in the i-th segment set. If there are many possible answers, you can print any of them.

Sample Input 1

```
3 9 2
0 4 1
4 9 1
1 8 1
```

Sample Output 1

```
1
1 1 0
```

Sample Input 2

```
3 9 3
0 4 1
4 9 1
0 9 1
```

Sample Output 2

```
2
1 1 1
```

Sample Input 3

```
5 9 3
0 4 1
2 6 1
7 8 1
5 9 1
1 7 1
```

Sample Output 3

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

Sample Input 4

```
5 9 40
0 4 30
2 6 18
7 8 99
5 9 70
1 7 12
```

Sample Output 4

```
13
13 1 0 13 12
```

Origin & Bonus (5 points)

In homework 1 bridge, we discovered that a wrong solution is able to pass all test data generated randomly. The cool but **wrong** method is mainly comprised of the following two steps:

1. Rotate every point $P_i \in P$ by an angle randomly.
2. Sort all points $P_i \in P$ according to the X-axis, and find the closest pair of points from all pairs of points if the distance of their positions is less than $Magic$. Formally,

$$\min_{i=1}^{N}\left(\min_{j=i+1}^{\min(j+Magic,N)}(Dis(P_i,P_j))\right)$$

Now please generate a test data to make the following solution (the source code below) wrong.

Source Code: `https://www.csie.ntu.edu.tw/~b07902133/wa-bridge.cpp`

Hint

This bonus can be reduced to the **Segments** problem.

Problem 5 - Trick-or-Treating (Hand-Written) (30 points)

*In problem 5, please **briefly** explain your solution in text. **Do not** use pseudo code, or you will receive penalty. Note that if you use Greedy or Dynamic Programming in any subproblem of this problem, you should prove their properties (optimal substructure, greedy-choice property).*

Halloween is coming! As a starving student writing ADA homework 2 in the dorm all day long, you decided to go Trick-or-Treating with your friend Ada and ask for some candy from professors at midnight.

There are N professors participating in the event. After knocking on the i-th professor's door, he/she will chat with you by the door for t_i unit time and give you c_i candy.

However, to show respect to the professor, students should not do anything except chatting with the professor during the t_i unit time. Furthermore, the i-th professor needs to rest after r_i unit time from midnight. Suppose any student knocks on the professor's door or chats with the professor during his/her resting time, or visits the same professor more than once. In that case, the professor will be furious and ask the ADA instructor to give the student an F.

You are both great cyclists and can move from one door to another with your bicycle immediately (no time consuming). Please maximize the candies you can receive without getting an F on ADA.

Lastly, there are some assumptions shown as follows, which are used in the following problems.

Assumption 1 *Every professor can only give* 1 *candy, i.e., for all* $1 \leq i \leq N$, $c_i = 1$.

Assumption 2 *The longer a professor chats, the longer he/she needs to rest, i.e., for all* $1 \leq i, j \leq N$, *if* $t_i < t_j$, *then* $r_i < r_j$.

Assumption 3 *The professors' chatting time sequence is non-decreasing, i.e., for all* $i < j$, $t_i \leq t_j$.

Assumption 4 *The total amount of candy received is smaller than* M, *i.e.,* $\sum_{i=1}^{N} c_i < M$.

Please answer the following problems. **Note that if you answer (3) correctly, you will automatically get the score of (2).**

(1) (3%) Given the following information, please calculate the maximum number of candies you can get.

- $N = 5$
- $t = [2, 2, 3, 6, 10]$
- $r = [3, 4, 15, 16, 10]$
- $c = [8, 5, 1, 9, 8]$

(2) (5%) Please design an algorithm with time complexity $O(N)$ under all **Assumption 1, Assumption 2**, and **Assumption 3** to calculate the maximum candy you can get. Also, you need to explain why your algorithm meets the time complexity requirement.

(3) (6%) Please design an algorithm with time complexity $O(N \log N)$ under **Assumption 1** to calculate the maximum candy you can get. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

(4) (6%) Please design an algorithm with time complexity $O(NM)$ under **Assumption 4** to calculate the maximum candy you can get. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

On the next day of Trick-or-Treating, you and your friend Ada would like to visit and thank all N professors individually. You know the i-th professor would like to chat with you and Ada for f_i and g_i unit time, respectively. In order to save time, you want to come up with two visit plans a and b for you and Ada, separately.

A visit plan is a non-negative sequence p with length N, indicating that one visits the i-th professor at time p_i and starts chatting until the conversation is over. It is possible that $a_i \neq b_i$ since Ada and you could visit professors in a different order.

Furthermore, the conversation should always be one-to-one; that is, one professor could only talk to one student at one time and vice versa. Thus, for all $1 \leq i \leq N$, either $b_i + g_i \leq a_i$ or $a_i + f_i \leq b_i$.

Now, your task is to construct two visit plans for you and Ada to minimize the ending time that both of you had visited the N professors individually.

To sum up, given the chatting time sequences f and g, please find out two visit plans a and b for you and Ada respectively such that $\max_{i=1}^{N}(\max(a_i + f_i, b_i + g_i))$ is minimum.

Please answer the following problems. **Note that if you answer (7) correctly, you will automatically get the score of (6).**

(5) (3%) Given the following information, please calculate the minimum ending time and construct a and b (if there are several possible answers, you could write down anyone arbitrarily).

- $N = 5$
- $f = [1, 2, 3, 5, 8]$
- $g = [13, 21, 1, 2, 3]$

(6) (3%) Prove that if it exists an x such that $(f_x + g_x) > \max(\sum_{i=1}^{N} f_i, \sum_{i=1}^{N} g_i)$, the minimum ending time is $f_x + g_x$.

(7) (4%) Please design an algorithm with time complexity $O(N)$ to construct two visit plans a and b. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

Problem 6 - String Problems (Hand-Written) (20 points)

*In problem 6, please **briefly** explain your solution in text. **Do not** use pseudo code, or you will receive penalty. Note that if you use Greedy or Dynamic Programming in any subproblem of this problem, you should prove their properties (optimal substructure, greedy-choice property).*

You're given two strings s1 and s2 ($1 \le |s1|, |s2| \le N$). Please determine whether it's possible to change the string from s1 to s2 with no more than K moves. (one addition, deletion, or replacement of one character counts as one move)

Please answer the following problems. **Note that if you answer (3) correctly, you will get the score of (1), (2) automatically.**

(1) (3%) If $K = 1$, Please design an algorithm with time complexity $O(N)$ to answer the questions. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

(2) (3%) If $K > 1$, Please design an algorithm with time complexity $O(NK^2)$ to answer the questions. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

(3) (4%) If $K > 1$, Please design an algorithm with time complexity $O(NK)$ to answer the questions. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

You're given two strings s1 and s2 ($1 \le |s1|, |s2| \le N$) over alphabet Σ. Consider all possible strings S over alphabet Σ, such that both of s1 and s2 are subsequence of S. Over all possible S, you want to find the one with minimum $D(S)$. If there is more than one possible S, anyone is acceptable.

- For any two characters c_1 and c_2 from Σ, we define $Dis(c_1, c_2)$ as the distance from c_1 to c_2.(You could consider $Dis(c_1, c_2)$ as a non-negative integer given by the problem)
- for any three characters c_1, c_2, c_3 from Σ, satisfied $Dis(c_1, c_2) + Dis(c_2, c_3) \ge Dis(c_1, c_3)$
- If $|S| = 1, D(S) = 0$
- If $|S| > 1, D(S) = \Sigma_{i=0}^{|S|-2}(Dis(S_i, S_{i+1}))$
- $|\Sigma| = K$
- A subsequence is a sequence that can be derived from another sequence by deleting some or no elements without changing the order of the remaining elements.

Please answer the following problems. **Note that if you answer (6) correctly, you will get the score of (5) automatically.**

(4) (3%) Given the following information, please find the S with minimum $D(S)$ and also the value of $D(S)$.
   - $\Sigma = \{a, b, c, d, e, f\}$
   - $Dis(c_1, c_2) = |ord(c_1) - ord(c_2)|$
   - ord(c) returns the ASCII value of the character
   - $s1 = "adefc"$
   - $s2 = "accfd"$

(5) (3%) Please design an algorithm with time complexity $O(N^2K)$ to find the S with minimum $D(S)$ and also the value of $D(S)$. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

(6) (4%) Please design an algorithm with time complexity $O(N^2)$ to find the S with minimum $D(S)$ and also the value of $D(S)$. Briefly explain the correctness and why your algorithm meets the time complexity requirement.