Homework #2

Due Time: 2019/11/12 (Thu.) 13:00
Contact TAs: `ada-ta@csie.ntu.edu.tw`

Instructions and Announcements

- There are **four programming problems** and **three 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 **MUST** turn in **a printed/written version** of your answers to **the submission box at R217**. You can also upload your homework to the NTU COOL system as a backup; however, it will be marked **only when** you have turned in the printed/written answer but it is lost during the grading process.
  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.
- **Scoring policy.** Although the total score in this homework is 105, you'll get 100 if your original score exceeds 100.

Problem 1 - Line of Battle (Programming) (10 points)

Problem Description

WillyPillow is recently obsessed with a mobile game called *Azure Line*, which involves multiple ships lining up and forming fleets. In the game, each ship is characterized by two parameters: a leader*ship* factor and a team awareness factor. In addition, we define a fleet as a non-empty set of contiguous ships in which the ship that comes first is the flagship. For a fleet to be efficient, the leader*ship* factor of the flagship has to be no smaller than the sum of the team awareness factors of the other ships in the fleet.

Now, WillyPillow is given a line of N ships, and he wants to calculate how many combinations of fleets he can form such that all fleets are efficient and one ship is assigned to exactly one fleet. Two combinations are considered different if there exists a pair of ships that are in the same fleet in one combination but are in different fleets in the other.

Formally speaking, he has to count the number of ways to partition a list into segments, with every segment $[l, r]$ satisfying $a_l \geq \sum_{i=l+1}^{r} b_i$, where a_i and b_i are the leadership and team awareness factors, respectively, of the i-th ship.

Input

Due to the sheer size of the input data, you are required to use the *generator* in the link below to generate data on the fly:

`https://gitlab.com/snippets/1904475`

Specifically, you need to paste the given *generator* at the beginning of your source file, and use it as follows:

```
int n = ada::Init(); // Get N
for (int i = 0; i < n; i++) leadership[i] = ada::GetLeadership();
for (int i = 0; i < n; i++) team_value[i] = ada::GetTeamValue();
```

You **may not** interact with the standard input by any other means, e.g., `scanf` and `cin`.

Note that in all test groups, the leader*ship* and team awareness factors are in the range $[0, 10^9]$.

Test Group 0 (0 %)

- Sample Input

Test Group 1 (5 %)

- $1 \leq N \leq 500$

Test Group 2 (15 %)

- $1 \leq N \leq 5 \times 10^3$

Test Group 3 (80 %)

- $1 \leq N \leq 2 \times 10^6$

Output

Please output an integer (to the standard output) indicating the number of ways to form combinations of fleets, modulo the prime $10^9 + 7$.

Sample Input 1

```
5 20 20 1537688804 584589912 3972715898 2166415565
```

Sample Output 1

```
2
```

Sample Input 2

```
10 20 20 4041108152 584535659 2466603739 198973427
```

Sample Output 2

```
46
```

Problem 2 - Chunithm (Programming) (15 points)

Problem Description

You are playing a game "chunithm". As the most talented student at NTU CSIE, you want to make sure that you can win the game with the least effort.

In the beginning of the game, you are given a song. The song is N seconds long and there is exactly one note within one second. You have a keyboard with M different notes, and all notes in the song can be played with this keyboard. You play the song with two hands on the keyboard. If the song contains the note j at i-th second, one of your hands must put on the note j of the keyboard during that second. The distance between notes j_1 and j_2 on the keyboard is $|j_i - j_2|$.

The effort of playing the game is defined as follows:
At every second, you can move each of your hands for distance K without any effort. If you move a hand for more than distance K in one second, the effort of moving the hand is (the number of steps you moved$-K$). The effort will accumulate over time and the effort experienced by both hands will add up.

Given the song, the size of the keyboard M, and the distance number you can easily move K, please output the minimum effort you need for finishing playing the song (the hands can start from anywhere on the keyboard).

Input

The first line of the input consists of 3 integers N, M, and K. The second line consists of N numbers a_0, a_1, ..., a_{n-1}. a_i indicates the note of the song at ith second.

- $1 \leq N \leq 100000$
- $1 \leq M \leq 300$
- $1 \leq K \leq M$
- $0 \leq a_i < M$

Output

An integer indicates the amount of effort.

Subtask 1 (30 %)

- $N \leq 100, M \leq 100$

Subtask 2 (30 %)

- $N \leq 3000$

Subtask 3 (40 %)

- No other constraints.

Sample Input 1

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

Sample Output 1

```
0
```

Sample Input 2

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

Sample Output 2

```
1
```

Problem 3 - Pokémon GO (Programming) (10 points)

Problem Description

WillyPillow is a Pokémon trainer, and he has N Pokémon. Each Pokémon has a unique ID from 1 to N, where the Pokémon with the ID i has the attack power A_i, an unknown parameter B_i, and an experience point E_i which is initially set to zero.

The battle between WillyPillow and his rival has K rounds, and both of them have to pick K Pokémon and arrange their attacking order. Let p_i be the ID of the Pokémon WillyPillow summons in the i-th round. The Pokémon will first cause $(A_{p_i} \times E_{p_i})$ damage to the opponent, then it will use its magic power to increase B_{p_i} experience points of all other WillyPillow's Pokémon which have not yet be summoned.

Because WillyPillow is busy preparing for the ADA exam, you, who live on Pokémon betting and have spent all money on this battle, decide to help him choose K Pokémon and arrange their attacking order such that the maximum total damage can be achieved.

Input

The first line of the input contains two integers N and K, indicating the number of Pokémon WillyPillow has and the number of rounds in the battle, respectively.

N lines follow, the i-th of which contains two integers A_i, B_i, denoting the attack power and the unknown parameter of the Pokémon with ID i.

- $1 \leq K \leq N \leq 100$
- $0 \leq A_i, B_i \leq 100$

Test Group 0 (10 %)

- $K = N$

Test Group 1 (10 %)

- $N \leq 10$

Test Group 2 (80 %)

- No other constraints.

Output

Print an integer indicating the maximum possible total damage that can be achieved if K Pokémon are chosen and arranged optimally.

Sample Input 1

```
9 4
6 8
0 13
14 14
14 1
14 4
```

```
13 5
3 11
5 6
3 12
```

Sample Input 2

```
6 6
10 7
0 8
5 3
10 7
2 9
1 3
```

Sample Output 1

```
994
```

Sample Output 2

```
673
```

Problem 4 - Weigh Anchor (Programming) (15 points)

Problem Description

After WillyPillow assembled his fleet in *Azure Line*, he can now fight enemies!

WillyPillow picks three most well-trained ships, he will only fight using those ships, with strengths s_1, s_2, and s_3, respectively.

To fight an enemy, he can select any subset of his ships. He wins a battle if and only if the sum of the strengths of his chosen ships are no smaller than the strength of the enemy.

However, each ship has a mood value that decreases when encountering a battle and recovers with time. Since WillyPillow is a good commander who does not want his ships to suffer, he can only use each ship once an hour.

For example, in a certain hour, he can fight two battles with ships $s_1 + s_2$ and s_3 respectively, but he can not fight three battles with ships $s_1 + s_2$, s_2, and s_3, since s_2 is used twice.

At the same time, since he needs to train Pokémon and prepare for the ADA exam (c.f. problem 3), he wants to spend as little time as possible on playing the game. Given a list of enemy strengths, he wants to find out how much time he needs to defeat all the enemies according to the rules above. Note that he can fight the enemies in any order.

Input

The first line of the input file contains an integer indicating N, the number of enemies.

The second line contains three integers separated by spaces, indicating s_1, s_2, and s_3, the strengths of WillyPillow's ships.

The third line contains N integers separated by spaces, with the i-th integer indicating the strength of the i-th enemy, denoted by a_i.

- $1 \leq N \leq 2 \times 10^5$
- $1 \leq s_1, s_2, s_3, a_i \leq 10^8$

Test Group 0 (5 %)

- $N \leq 4$

Test Group 0 (10 %)

- $N \leq 10$

Test Group 0 (85 %)

- No other constraints.

Output

Please output an integer indicating the minimum number of hours WillyPillow needs to defeat all the enemies.

Sample Input 1

```
8
```

```
6 11 7
10 20 13 12 20 3 5 10
```

Sample Output 1

```
5
```

Sample Input 2

```
8
7 16 9
4 17 8 4 15 6 13 1
```

Sample Output 2

```
3
```

Problem 5 - Piepie's Pie Shop (Hand-Written) (16 points)

Piepie is a genius developer in Coming Spiorad International Enterprise and also a world famous chocolate-pie maker. People are willing to line up for 49 hours in order to purchase the chocolate pie he made, even he can make 1025 chocolate pies per hour. Nevertheless, Piepie felt board by making delicious chocolate pies after earning his first trillion dollars. Thus, he decided to open a brand new pie shop.

The new pie shop has only one table that can accommodate a group of N customers at the same time. A group of customers will come to the shop at the same time and leave together. Each customer i will have his/her customized pie that takes Piepie p_i minutes, and the customer will finish eating the pie exactly e_i minutes after the pie is served. Piepie only makes one pie simultaneously, and there is no time gap between two pies.

After opening the shop, he notices that the order of making pies affects the group's leaving time. In order to make the group leave as fast as possible and make great money, he is seeking the best strategy of arranging the order and the corresponding minimum time needed for finishing serving the group of customers with the given information. Although he is a genius, he is sometimes too lazy to think. Can you please finish the task for him?

For this problem, each customer can be represented by a number from 1 to N.

Note that if you use Greedy or Dynamic Programming in any subproblem of this problem, you should prove their properties.

(1) (2pts) When there are $N = 5$ people among a table of customers. The preparation time and eating time are as followed:

[(2, 13), (4, 2), (3, 4), (7, 3), (5, 5)]

Please show your arrangement and the minimum time needed.

(2) (4pts) Please design an algorithm to provide your arrangement and calculate the total time needed. Your algorithm should run in $O(N \lg N)$ time. Also, you need to explain why your algorithm meets the time complexity requirement.

(3) (4pts) Please prove the correctness of your algorithm in (2).

(4) (3pts) To reach the true power of making pie, he decided to learn more in the Ninja School. After he returned, Piepie is able to perform "SHADOW CLONE JUTSU". To enhance the productivity of his pie shop, he clones himself into piepie00 and piepie01. They believe that they can perform the best with the following same strategy as we perform in subproblem (2). Prove their daring thought or disprove with a counterexample.

(5) (3pts) In addition to "SHADOW CLONE JUTSU", Piepie also learned "CHIDORI" to kill at most one of the customer. He is wondering who he kills will minimize the time needed.
Please give an algorithm that runs in $O(N \lg N)$ time complexity and briefly explain the correctness and the time complexity of your algorithm.

Problem 6 - Mobile Diners (Hand-Written) (14 points)

y¿m2cm z¿m1cm

In an another world of strange dimension, there is a Natural Π University (will be called NΠU in the rest of this problem). The world is so strange that all the classrooms in NΠU lies on the only revenue, "Pineapple Revenue". NΠU is made up of N classes, to simplify the problem, we assume the each class i locates on x_i, indicating it's x_i units far from the entry of NΠU.

When it comes to lunch time, the students have no choice but to choose from the mobile diners. Each mobile diner has a delicious rate d, indicating that the students will only move at most d units from their classrooms to have their lunch on that mobile diner.

It's currently the recruiting stage and there are M mobile diner candidates and the delicious rate are $d_1, d_2, ..., d_M$, respectively. Due to the property of the strange world, the mobile diner with smaller index should always lie on the side with smaller coordinate. In order to reduce the personnel costs, you are asked to make a plan that make all the students be able to have meal with least mobile diners.

In this problem, please assume that x is monotonic increasing, that is , for any $i < j$, $x_i < x_j$.

Note that if you use Greedy or Dynamic Programming in any subproblem of this problem, you should also prove their properties.

(1) (2pts) Please show your plan when there are 5 classes and they locates on $x = \{1, 7, 11, 12, 17\}$ and there are 4 mobile diner candidates and the delicious rate are $d = \{3, 2, 5, 4\}$.

(2) (2pts) In this subproblem, please assume that all the mobile diners are identical and share the same delicious rate d^*.
Please provide an algorithm to find the minimum number of mobile diners needed. Your algorithm should run in $O(N + K)$ time complexity. Briefly explain the correctness and why your algorithm meets the time complexity requirement.
Note: You can get full credit of this problem if you answer subproblem (3) correctly.

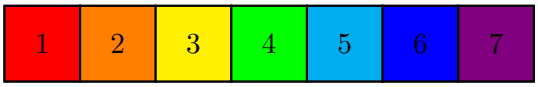
(3) (4pts) The president of NΠU collects dirty money from the mobile diners, there are unfair priority and the priority is exactly the index of the mobile diners!

Please consider the following scenario in this subproblem: *For any two mobile diners with indices $i < j$, j is picked only if i is picked.*. That is, if K mobile diners are selected, they must be the first K ones.
Please provide an algorithm to find the minimum number of mobile diners needed. Your algorithm should run in $O(N + K)$ time complexity. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

(4) (6pts) The corruption was revealed and the president is forced to quit. The new president is now open and enlightened. In subproblem (4), there are no priority among the mobile diners anymore. However, the property that mobile diners with smaller indices should locate on smaller coordinates still remains.
Please provide an algorithm to tell the minimum number of mobile cars. Your algorithm should run in $O(NM)$ time complexity. Briefly explain the correctness and why your algorithm meets the time complexity requirement.

Problem 7 - Rainbow Rarity Rally (Hand-Written) (20 points)

"Rainbow Rarity Rally" is an annual flying race held in Cloudsdale, Equestria. There are $N \geq 7$ magical rainbow candies floating in the sky on the same 2-dimensional plane, indexed from 1 to N. Each candy has a color from one of the seven colors of rainbow, ROYGBIV(red, orange, yellow, green, blue, indigo and violet, indexed from 1 to 7). For each color, there is at least one candy with that color. There is also a start point on the ground. A contestant needs to fly upwards from the start point on the ground to the sky, and then fly downwards back to the start point, collecting all candies in the process.



Let C be an integer greater than or equal to N. We use $N+1$ points $p_0, p_1, \cdots, p_N$ described in Cartesian coordinate system to represent a race. $p_i = (x_i, y_i)$, where y_i represents the vertical height of the point.$(x_i, y_i \in \mathbb{Z}, 0 \leq x_i, y_i \leq C)$. The start point is p_0, where $y_0 = 0$. The i-th candy is on the point p_i, and has color $c_i (c_i \in \mathbb{Z}, 1 \leq c_i \leq 7)$. In order to uphold the holy tradition of the great pegasi race, $y_i < y_{i+1}$ for all $i \in \{0, 1, \cdots, N-1\}$.

Since Equestria is a magical kingdom ruled by chromatic sapient ponies who speak English, law of physics from our world don't apply there. Therefore, it takes a pegasus $f(i,j) = (x_i - x_j)^2 + (y_i - y_j)^2$ units of time to fly from p_i to p_j.

Each contestant has to design his/her own route that satisfy the following conditions. A route consists of two parts.

- Part 1. Contestants start from p_0 and fly upwards in the sky, stopping at p_N.
- Part 2. Continued from part 1, contestants start from p_N and fly downwards, stopping at p_0.
- In order to collect all candies, for all $i \in \{1, 2, \cdots, N\}, p_i$ is visited exactly once in the route.

Rainbow Dash is an ambitious pegasus athlete who aims for the trophy. She has a special aerobatics called "Sonic Rainboom" that can only be performed if for each color she collects at least one candy with that color in "part 1" of her route. We call a route a "Sonic-Rainboomable route" if it allows Rainbow Dash to perform "Sonic Rainboom". Rainbow Dash is not an egghead! She doesn't have time to do the math! Please help her!

Formally speaking:

- A route is an integer sequence a of length $N+2$.
- $a_0 = a_{N+1} = 0$ and each $i \in \{1, 2, \cdots, N\}$ appears in a exactly once.
- Let q be the index of N in a.
- $\forall i \in [0, q-1], y_{a_i} < y_{a_{i+1}}$.
- $\forall i \in [q, N], y_{a_i} > y_{a_{i+1}}$.
- $a_0, a_1, \cdots, a_q$ is called "part 1" of the route.
- $a_q, a_{q+1}, \cdots, a_{N+1}$ is called "part 2" of the route.
- The time required to fly through a route is $\Sigma_{i=0}^{N} f(a_i, a_{i+1})$.
- A route is Sonic-Rainboomable if $|\{c_{a_i} \mid i \in [1, q]\}| = 7$

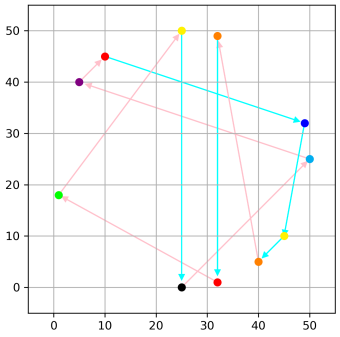


Fig 1.

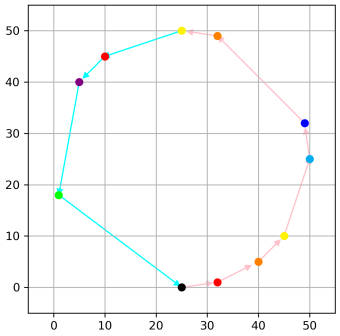


Fig 2.

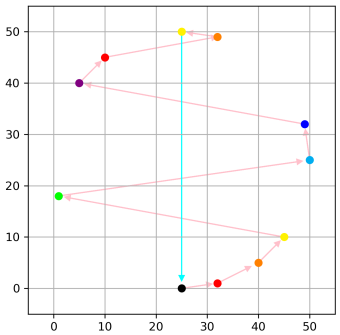


Fig 3.

(Task 1) (8 points) Please design a dynamic programming algorithm that finds a route that takes her the smallest amount of time with $O(N^2)$ time complexity.

(Task 2) (6 points) Please design a dynamic programming algorithm that finds a route that takes her the smallest amount of time with $O(N^2)$ time complexity and $O(N)$ space complexity.

(Task 3) (4 points) Please design a dynamic programming algorithm that finds a Sonic-Rainboomable route that takes her the smallest amount of time with $O(N^2)$ time complexity.

(Task 4) (2 points) Please design a dynamic programming algorithm that finds a Sonic-Rainboomable route that takes her the smallest amount of time with $O(N^2)$ time complexity and $O(N)$ space complexity.

Note

1. You have to prove the correctness and complexity of your algorithm!
2. **If you solved task 2, you get score from task 1 automatically. If you solved task 3, you get score from task 1 automatically. If you solved task 4, you get score from task 1,2,3 automatically. Start saving ink today! Save our Earth!**
3. $O(1) = O(2) = O(7) = O(127)$. We all love 127 :)

Time and Space Complexity and Computational Model

1. We use Word RAM (Wikipedia contributors, 2019b) as the computational model in this problem. It is a kind of Random-access machine(Wikipedia contributors, 2019a). If you don't want to read long articles or the following descriptions, just think of it as a single thread C program running on your personal computer.
2. Let $K = 2C^2(N+1)$. Let T be the set of integers in range $[-K, K]$. Let $w = 1 + \lceil \log_2(K+1) \rceil$.
3. A w-bits signed integer using the Two's complement representation is the basic data structure in this problem. Let's call it `word` . Obviously, we can store any $x \in T$ in an `word` .
4. In this problem, we assumed that an `word` takes $O(1)$ space.
5. Assignment of an `word` takes $O(1)$ time.
6. Arithmetic operations (addition, subtraction, multiplication, bitwise NOT, bitwise AND , bitwise OR , bitwise XOR , bitwise left shift , bitwise right shift) of two `word` take $O(1)$ time.
7. Comparison operations (equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to) of two `word` take $O(1)$ time.

References

Wikipedia contributors. (2019a). *Random-access machine — Wikipedia, the free encyclopedia.* `https://en.wikipedia.org/w/index.php?title=Random-access_machine&oldid=915665148`. ([Online; accessed 15-October-2019])

Wikipedia contributors. (2019b). *Word ram — Wikipedia, the free encyclopedia.* `https://en.wikipedia.org/w/index.php?title=Word_RAM&oldid=883343686`. ([Online; accessed 15-October-2019])