NCTU Competitive Programming I
2020 Spring
Final Exam 1

ID	Problem Name	Time Limit
A	Area	2 sec
B	Blockly	2 sec
C	Correctable Strings	2 sec
D	Delta? Lambda!	2 sec
E	Escape	2 sec
F	Fix the Sequence	2 sec
G	Grid Tiling	2 sec
H	Hacking	2 sec

NCTU Competitive Programming I
2020 Spring
Final Exam 1

Rules

Any violation of the rules is considered cheating and will result in a disqualification.

1. You should take place in the exam using your personal and only DOMjudge account given prior to the exam.
2. You must not discuss problems or share ideas/solutions with any other people.
3. Should you have a question about the problems or face any exam-related issue, please use the "request clarification" feature of DOMjudge. Do not ask or discuss with any other people.
4. Announcements made during the exam can be viewed using the "Clarifications" interface of DOMjudge.
5. You may use existing source code as parts of your solution. However, the code you use must either be in the lecture notes or be written by you prior to the exam.
6. You may search for and read documents or references online, but copying solution codes are not allowed.
7. Any malicious action interfering the exam is prohibited.

Scoring

1. You must submit your solutions via DOMjudge. The judge system will only respond to submissions that are submitted within the exam duration (300 minutes). The response to each run must be one of the following:
   - `Correct`: The judge accepts your code.
   - `Compiler-Error`: Your code cannot be successfully compiled.
   - `TimeLimit`: Your program consumes too much time.
   - `Run-Error`: Your program terminates with an non-zero return code, which often means your program is terminated by the operating system.
   - `Wrong-Answer`: The judge rejects the output of your program.
   - `No-Output`: Your program does not generate any output.

2. The score you get for final exams is based on the total number of problems to which a correct solution is submitted:

Solved	Score
0	20
1	30
2	35
3	39
4	43
5	46
6	48
7	49
8	50

Hint

The problems are sorted by their names (lexicographically) and not by increasing difficulty; problems that appear first are not necessarily easier. Because of this, it is recommended that you read every problem.

Almost blank page

Problem A
Area

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

This problem is modified from problem A in CodeForces Round 644 (Div. 3), see:
`https://codeforces.com/contest/1360/problem/A`

Now, we ask you to place a rectangular $a \times b$ house and a rectangular $c \times d$ house on a *square* land with the minimum area. The houses must not overlap. You may rotate the houses and the land, but you must keep the sides of the houses parallel to the sides of the desired *square* land. Please write a program to output the minimum area of the land satisfying the constraints above.

Note: a *square* is a rectangle with four equal sides.

Input Format

The first line contains an integer T indicating the number of testcases.

A test case is a line containing 4 space-separated positive integers a, b, c, and d. Your task is to find the minimum area of the square land such that you may put a rectangular $a \times b$ house and a rectangular $c \times d$ house on.

Output Format

For each test case, output the minimum area of the desired square land.

Technical Specification

- $0 < T \leq 10,000$
- $a, b, c, d \in [1, 10^9]$
- You might need to use a 64-bit integer to represent the answer.

Sample Input 1

```
10
1 2 3 4
5 6 7 8
9 1 2 3
4 5 6 7
8 9 1 2
3 4 5 6
7 8 9 1
2 3 4 5
6 7 8 9
1 2 1 2
```

Sample Output 1

```
16
144
81
100
81
64
81
36
196
4
```

Problem B
Blockly

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

Your lord, Lucifer, kidnaps Perly Princess and wants to occupy the kingdom. However, an epic hero, Blockly, will obstruct Lucifer's plan and rescue Perly Princess.

To prevent Blockly from saving Perly Princess. You have to create a magical Hydra with H heads and T tails. You can use four types of operations listed below.

1. Add 3 heads.
2. Exchange 2 heads for 3 tails. You may use this operation only when the magical has 2 or more heads.
3. Exchange 1 tails for 2 heads. You may use this operation only when the magical has at least 1 tail.
4. Cut off 2 tails. You may use this operation only when the magical has 2 or more tails.

You have an embryo of a magical Hydra with 0 heads and 0 tails. You want to know how many operations are needed to create a magical Hydra with H heads and T tails.

Input Format

The input contains only two non-negative integers H and T indicating that you need to create a magical Hydra with H heads and T tails.

Output Format

Output the minimum number of operations to create the desired magical Hydra.

Technical Specification

- $0 \leq H \leq 100$
- $0 \leq T \leq 100$

Sample Input 1

```
1 1
```

Sample Output 1

```
3
```

Sample Input 2

```
50 50
```

Sample Output 2

```
48
```

Almost blank page

Problem C
Correctable Strings

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

Billy types fast, and often makes typos when he texts other people. However, each time his phone's default auto correction system helps him correct typo, he finds that it just corrects it into another wrong word. Therefore, he decides to create his own auto correction system!

The first step he decides to do is to determine what kinds of mistake he makes. He assumes that any of he typos is a string "equivalent" to some word in his dictionary.

The two strings s and t are "equivalent" if they are the same length, and satisfy one of the following conditions.

1. The length is at most one, and the strings are the same.
2. The length is greater than one, and they have only one position with different character.
3. We split s into s_1 and s_2 where $|s_1| = |s_2|$ or $|s_1| = 1 + |s_2|$. Similarly, we also split t into t_1 and t_2 where $|t_1| = |t_2|$ or $|t_1| = 1 + |t_2|$. If s_1 is equivalent to t_1 and s_2 is equivalent to t_2, then we say s and t are equivalent.

To ensure that his assumption is correct, he first gives you the correct word that he wants to type, and then shows you what he typed. You must tell him whether the two words are equivalent according to his rules.

Input Format

The first line contains one integer n indicating the length of the strings. The second line contains a word of n characters in Billy's dictionary. And the third line contains the string of n characters typed by Billy.

Output Format

If the two words are equivalent, print "`True`" (without quotes). Otherwise, print "`False`" (without quotes).

Technical Specification

- $1 \leq n \leq 500,000$
- All characters are lowercase.

Sample Input 1

```
5
brick
bricl
```

Sample Output 1

```
True
```

Sample Input 2

```
5
asdfg
addgg
```

Sample Output 2

```
True
```

Sample Input 3

```
6
timber
itmber
```

Sample Output 3

```
False
```

Problem D
Delta? Lambda!

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

The shape of Δ (Delta) is a triangle. What is the shape of Λ (Lambda)? In this problem, (x, y, z) is a Λ-shape triple if $x < y$ and $y > z$.

You are given a sequence $a_1, a_2, \ldots, a_n$ of n integers. You find that there are many its 3-element subsequences forming Λ-shape triples. For example, the 4-element sequence $a_1 = 1$, $a_2 = 2$, $a_3 = 3$, $a_4 = 0$ has 3 subsequences which form Λ-shape triples: $(a_1, a_2, a_4) = (1, 2, 0)$, $(a_1, a_3, a_4) = (1, 3, 0)$, and $a_2, a_3, a_4 = (2, 3, 0)$. Please note that we do not count $(a_1, a_3, a_2) = (1, 3, 2)$, since a_1, a_3, a_2 is not a subsequence. Let a subsequence a_i, a_j, a_k be a Λ-shape subsequence if $i < j < k$, $a_i < a_j$ and $a_j > a_k$.

Let f_i be the number of Λ-shape subsequences that contains a_i. Please write a program to compute the sequence $f_1, f_2, \ldots, f_n$. The sequence f_1, f_2, f_3, f_4 corresponding to the example above is $2, 2, 2, 3$.

Input Format

The first line contains a positive integer n indicating the number of elements of the sequence. Then the second line contains n integers $a_1, a_2, \ldots, a_n$ representing the given sequence.

Output Format

Print n numbers $f_1, f_2, \ldots, f_n$ separated by blanks on one line.

Note: exactly one space between f_i and f_{i+1} for $i \in [1, n)$.

Technical Specification

- $3 \le n \le 2 \times 10^5$
- $1 \le a_i \le 2 \times 10^5$ for $i \in [1, n]$

Sample Input 1

```
4
1 3 2 1
```

Sample Output 1

```
3 2 2 2
```

Sample Input 2

```
6
5 1 4 6 4 1
```

Sample Output 2

```
2 4 3 6 4 5
```

Hint

You might need to implement some data structures based on binary trees.

Problem E
Escape

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

You are trapped in a grid maze of size $2 \times n$, and each grid cell is a room. The coordinates of a room can be represented by (r, c) where $r \in [1, 2]$ and $c \in [1, n]$. Room (r, c) is located at the row r and column c.

You are now at room $(1, 1)$ and want to escape. The exits are at $(1, n)$ and $(2, n)$, but there are locked now. To unlock them, you have to press the toxic gas button in every room. Once you press the button, the toxic gas will start to fill in a few seconds. Before releasing the gas, the room opens doors to adjacent rooms. The doors will automactically close when the concentration of toxic gas in the room is high enough to kill people. Fortunately, you can run really fast. You still can escape to an adjacent room before the toxic gas kills you. However, you may not return to the room with toxic gas released.

In this problem, we define two rooms (r_1, c_1) and (r_2, c_2) are adjacent if all following conditions hold.

1. $(r_1, c_1) \neq (r_2, c_2)$
2. $|r_1 - r_2| \leq 1$
3. $|c_1 - c_2| \leq 1$

You must press all toxic gas buttons, and the last button must be in either room $(1, n)$ or $(2, n)$. Otherwise, you cannot escape from the maze. It is better to have some plans to escape. You wonder how many different paths allow you to escape. The answer might be large, output the answer modulo $1,000,000,007$.

Input Format

The input contains a positive integer n indicating that the size of maze is $2 \times n$.

Output Format

Print the number of paths allowing you to escape modulo $1,000,000,007$ on a line.

Technical Specification

$1 \leq n \leq 10^{18}$

Sample Input 1

```
2
```

Sample Output 1

```
4
```

Sample Input 2

```
3
```

Sample Output 2

```
12
```

Sample Input 3

```
999990000099999
```

Sample Output 3

```
37422585
```

Hint

The following figure illstrates the first sample test case.

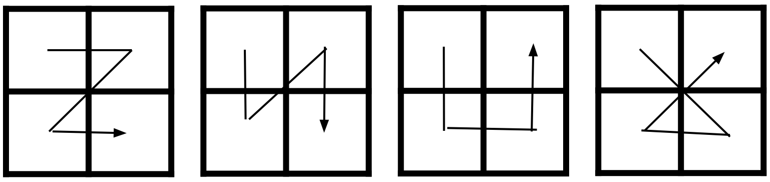

Figure 1: Four paths allow you to escape.

Problem F
Fix the Sequence

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

Frank loves numbers and music! The best way to show his love is to compose music using numbers. One day, he received a "music sequence" from the fourth dimension. The music sequence is a sequence $[a_1, a_2, \ldots, a_n]$ of n integers where each value is between 0 and m (inclusive). He also received n positive integers $b_1, b_2, \ldots, b_n$.

Frank can apply a series of operations to the sequence. There are three types of them:

- $inc(i)$: He changes the value of a_i to $a_i + b_i$. This operation is applicable only if $a_i + b_i \leq m$.
- $dec(i)$: He changes the value of a_i to $a_i - b_i$. This operation is applicable only if $a_i - b_i \geq 0$.
- $nop(i)$: He does nothing to a_i.

We define a sequence $[a_1, a_2, \ldots, a_n]$ to be *beautiful* if and only if there is an index j such that:

- $1 < j < n$,
- $a_1 < a_2 < \cdots < a_j$, and
- $a_j > a_{j+1} \cdots > a_n$.

For example, the sequences $[1, 2, 3, 0]$ and $[2, 222, 22]$ are beautiful, while $[1, 3, 3, 1]$ and $[2, 22, 222]$ are not. Frank's goal is to change the sequence he received into a beautiful one. For every minute, he chooses and performs one of the operations $inc(i)$, $dec(i)$, or $nop(i)$ for each $i \in [1, n]$, and stops as soon as the goal is reached.

Help Frank find out the minimum time required to make the sequence beautiful, or tell if it's impossible.

Input Format

The first line contains two integers n and m separated by a space.

The second line contains n space-separated integers $a_1, a_2, \ldots, a_n$.

The third line contains n space-separated integers $b_1, b_2, \ldots, b_n$.

Output Format

Print one integer indicating the minimum number of minutes required to make the sequence beautiful. If it is impossible to do so, print `-1` instead.

Technical Specification

- $3 \leq n \leq 3 \times 10^5$
- $2 \leq m \leq 10^9$
- $0 \leq a_i \leq m$ for $i \in [1, n]$.
- $1 \leq b_i \leq m$ for $i \in [1, n]$.

Sample Input 1

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

Sample Output 1

```
2
```

Sample Input 2

```
3 222
2 22 222
200 201 200
```

Sample Output 2

```
-1
```

Hint

In the first sample test case, the original sequence is $[1, 2, 1, 2, 4]$. Frank can make it beautiful in 2 minutes:

- In the first minute, he applies $nop(1)$, $nop(2)$, $inc(3)$, $inc(4)$, and $dec(5)$. The sequence becomes $[1, 2, 4, 4, 3]$.
- In the second minute, he applies $nop(1)$, $nop(2)$, $inc(3)$, $nop(4)$, and $nop(5)$. The sequence becomes $[1, 2, 7, 4, 3]$ which is beautiful.

In the second sample test case, it is not possible to make the sequence beautiful using the operations.

Problem G
Grid Tiling

Time limit: 2 seconds
Memory limit: 512 megabytes

Problem Description

You are given an n-by-n grid wall with m holes. Each grid cell is a square, and there is at most one hole on any grid cell. Your boss ask you to tile the wall with 2-by-1 tiles. Each tile can cover two adjacent grid cells. The tiling must satisfy the following conditions.

- The cells without a hole must be covered by a tile.
- The cells with a hole must not be covered by any tile.
- Each tile must exactly cover two cells.

For example, the following figure shows two ways to tile a 3-by-3 wall with 1 hole on the center grid cell.

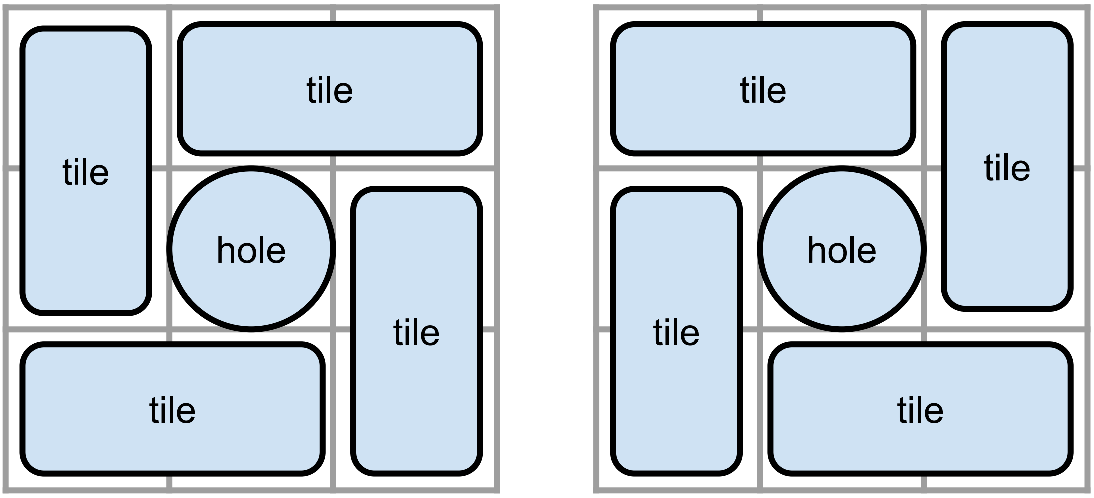

Figure 2: Two ways to tile

Please write a program to compute the number of ways to tile the given grid wall without violating the conditions. The number can be large, if it is greater than $1,000,000,006$, please output the number modulo $1,000,000,007$.

Input Format

The first line contains two integers n and m. The wall is an n-by-n grid and has m holes.

The followin m lines describes the holes. The i-th of them contains two positive integers r_i and c_i indicating there is a hole on the cell lying on row r_i and column c_i.

Output Format

Output the number of ways to tile the wall without violating the conditions modulo $1,000,000,007$ on a line.

Technical Specification

- $1 \leq n \leq 16$
- $0 \leq m \leq n^2$
- $r_i, c_i \in [1, n]$ for $i \in [1, m]$.
- There is at most one hole on any grid cell.

Sample Input 1

```
3 1
2 2
```

Sample Output 1

```
2
```

Sample Input 2

```
2 2
1 1
2 2
```

Sample Output 2

```
0
```

Sample Input 3

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

Sample Output 3

```
0
```

Sample Input 4

```
16 0
```

Sample Output 4

```
378503901
```

Hint

This problem is hard.

Problem H
Hacking

Time limit: 2 seconds

Memory limit: 512 megabytes

Problem Description

Recently, Hank starts to play a new game. Hank hacks the game and learns how its lottery system works. There are P pools numbered from 1 to P, and gamers can only draw super super rare (SSR) cards from one specific pool every day. Hank extracts the parameters $p_1, p_2, \ldots, p_N$ and $s_1, s_2, \ldots, s_N$ from the game server. He discovers that they are parameters for the lottery system in the next N days. On the i-th day, a gamer can only draw s_i SSR cards in total from pool p_i.

The parameters are very helpful for obtaining SSR cards. But another rule troubles Hank. Hank may change his pool before he start to draw cards on each day. He may change as many time as he wants, but he can only change his pool from pool i to pool j where $i < j$. That is, Hank can change his pool from pool 1 to pool 3, but he cannot change his pool from pool 3 to pool 2.

Assume that at the beginning, Hank's pool is pool 1. Hank wonders how many SSR cards can he draw from the pools in the next N days. Please write a program to compute the answer.

Input Format

The first line contains two positive integers P and N where P is the number of pools and Hank has parameters of the next N days. There are N lines following. The i-th of them contains two positive integers p_i and s_i. Hank can draw s_i SSR cards from pool p_i on day i.

Output Format

Print the maximum number of SSR cards drawn by Hank on one line.

Technical Specification

- $0 < P \leq 100$
- $0 < N \leq 100,000$
- $0 < p_i \leq P$ for $i \in [1, N]$.
- $0 < s_i \leq 10,000$ for $i \in [1, N]$.

Sample Input 1

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

Sample Output 1

```
9
```

Sample Input 2

```
5 5
5 100
4 4
3 3
2 2
1 1
```

Sample Output 2

```
100
```

Hint

The first sample test case has 5 pools and 10 days. On day 2, Hank draws 2 SSR cards from pool 1. On day 4, Hank changes his pool to pool 2 and draw 4 SSR cards. On day 9, Hank draws 3 SSR cards from pool 2.

The second sample test case has 5 pools and 5 days. On day 1, Hank changes his pool to pool 5 and then draw 100 SSR cards.