Boat

In the city of Seoul, a river called the Han River flows in the east-west direction. On the northern shore of the river there are N boating schools numbered from 1 to N as you move from the western end to the eastern end of the shore. All boats from the same school have the exact same color and thus are indistinguishable. The boats from different schools always have different colors and thus are always distinguishable. The school numbered i may choose to not send any boats to the festival. If it chooses to send boats to the festival it may send any number of boats from a_i to b_i, inclusive. ($a_i \leq b_i$)

One key condition is that the number of boats sent by the school numbered i, if it has chosen to send any boats, should be *larger than* the number of boats sent by any school numbered less than i, if any such school have chosen to send boats.

Task

Given a_i's and b_i's for all schools, find the number of all possible ways the schools may send boats to the festival, under the condition that at least one school chooses to send boats.

Input

The first line of the input contains a single integer N -- the number of schools. The i'th of the next N lines contains two integers a_i and b_i. ($1 \leq a_i \leq b_i \leq 10^9$)

Output

The output should consist of a single line with the remainder when the number of all possible cases the schools may send boats to the festival is divided by $1,000,000,007$.

Example

Input	Output	Comments
2 1 2 2 3	7	There are 4 ways where only one school sends boats and 3 ways where both schools send boats and thus the answer is 7.

Scoring

Subtask 1 (9 points): $1 \leq N \leq 500$ and for all $1 \leq i \leq N$, $a_i = b_i$.

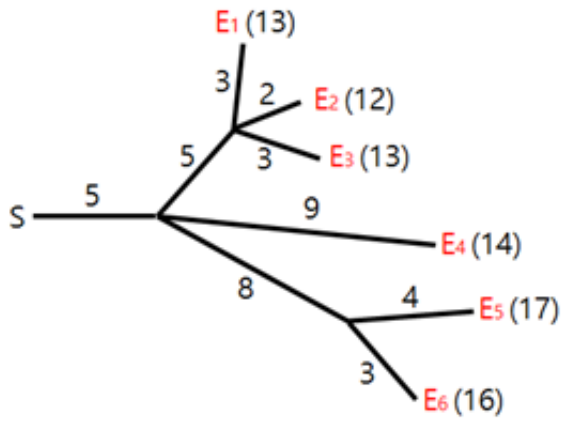
Subtask 2 (22 points): $1 \leq N \leq 500$ and $\sum_{1 \leq i \leq N} (b_i - a_i) \leq 10^6$.

Subtask 3 (27 points): $1 \leq N \leq 100$.

Subtask 4 (42 points): $1 \leq N \leq 500$.

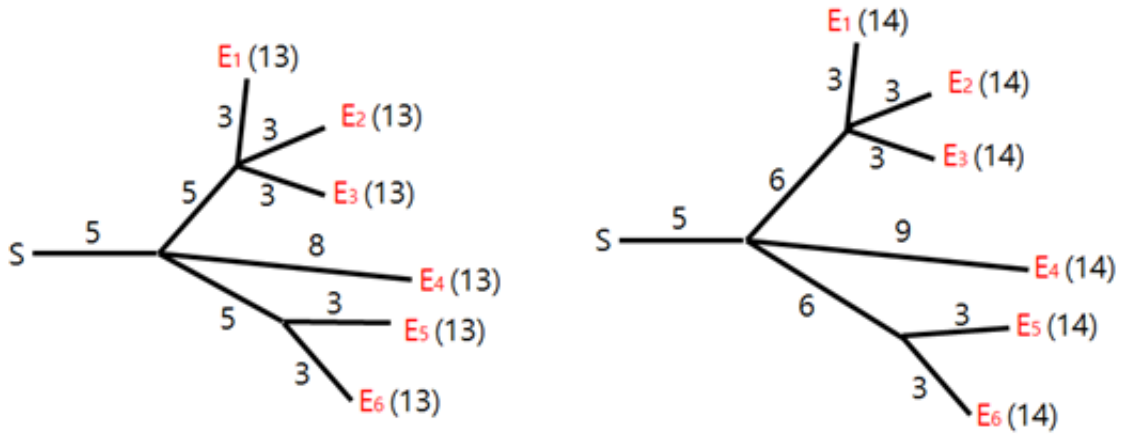
Fireworks

Fireworks display is one of the most exciting events in a festival. It is important in a fireworks display that every explosive connected to a switch by fuses explodes simultaneously at a planned time. Since the explosives used in the fireworks are very dangerous, they are set up far apart from the switch and are connected to the switch by some number of fuses. To connect several explosions to the switch fuses are connected as if edges are connected in a tree as shown in [Figure 1]. The spark starts from the switch, and moves along the fuses. When a spark reaches a junction, the spark spreads to all the fuses connected to the junction. The speed at which the sparks move is constant. [Figure 1] shows how six explosives $\{E_1, E_2, \ldots, E_6\}$ are connected and how long each fuse is. Also it shows the explosion time assuming that the starting time of a spark at the switch is 0.



[Figure 1] Connection Layout

Hyunmin, who participated in the fireworks display, made a connection layout. Unfortunately, in his layout, the explosives may not explode at the same time. We want to have all explosives explode at the same time by changing the lengths of some fuses. For example, to have all the explosives in [Figure 1] explode at time 13 lengths of fuses can be adjusted as shown in the left figure in [Figure 2]. Similarly, to have all the explosives in [Figure 1] explode at time 14 lengths of fuses can be adjusted as shown in the right figure in [Figure 2].



[Figure 2] Examples of fuse length changes that lead to simultaneous explosions

The cost of changing the length of a fuse is equal to the absolute value of difference in fuse length. For example, if the layout shown in [Figure 1] changes to the layout on the left in [Figure 2], the total cost is 6. If the layout shown in [Figure 1] changes to the layout on the right in [Figure 2] the total cost is 5.

The length of a fuse can be fully reduced to 0, retaining the connectivity among junctions.

Given a connection layout, you are to make a program which adjusts the fuse lengths so that all the explosives explode at the same time with minimum cost.

Input

All input values are positive integers. Let N denote the number of junctions, M the number of explosives. Every junction is identified by a number from 1 to N. The junction numbered 1 is where the switch is located. Every explosive is identified by a number from $N+1$ to $N+M$.

The input is given as follows:
$N\ M$
$P_2\ C_2$
$P_3\ C_3$
$\ldots$
$P_N\ C_N$
$P_{N+1}\ C_{N+1}$
$\ldots$
$P_{N+M}\ C_{N+M}$

P_i, $1 \leq P_i < i$, identifies the junction which is connected to either junction or explosive numbered i. C_i denotes the length of the fuse used to connect them ($1 \leq C_i \leq 10^9$). The number of fuses connected to a junction except the switch is more than 1 and that of fuses connected to a explosive is exactly 1.

Output

Print the minimum cost to adjust the lengths of fuses to have all the explosives explode at the same

time.

Example

Input	Output
4 6 1 5 2 5 2 8 3 3 3 2 3 3 2 9 4 4 4 3	5

Scoring

Subtask 1 (7 points): $N = 1, 1 \leq M \leq 100$.

Subtask 2 (19 points): $1 \leq N + M \leq 300$ and the longest distance between the ignition switch to an explosive is less than or equal to 300.

Subtask 3 (29 points): $1 \leq N + M \leq 5,000$.

Subtask 4 (45 points): $1 \leq N + M \leq 300,000$.

Gap

There are N non-negative integers $a_1, a_2, \ldots, a_N$ satisfying the following inequality $0 \leq a_1 < a_2 < \cdots < a_N \leq 10^{18}$. Jeehak wants to know the *largest possible* value of $a_{i+1} - a_i$ where i ranges from 1 to $N-1$. The input integers will not be given directly to Jeehak's program but will be accessible through a special funtion. See sections Implementation of your selected programming language for details.

Task

Help Jeehak to implement a function to return the largest possible value of $a_{i+1} - a_i$ where i ranges from 1 to $N-1$.

Implementation for C and C++

You need to implement one function `findGap(T, N)` that takes the following parameter and returns an integer of type `long long`:

- `T` — the subtask number (1 or 2)
- `N` — the number of given integers

Your function `findGap` can call function `MinMax(s, t, &mn, &mx)` where the first two parameters `s` and `t` are integers of type `long long` and the last two paramters `&mn` and `&mx` are pointers to integer variables of type `long long`, i.e., `mn` and `mx` are integer variables of type `long long`. When `MinMax(s, t, &mn, &mx)` returns, the variable `mn` will have the value of smallest a_i larger than or equal to the value of `s` and the variable `mx` will have the value of largest a_j smaller than or equal to the value of `t`. In case there are no input integers between `s` and `t` (inclusive), then both `mn` and `mx` will have the value `-1`. The value of `s` should be no larger than the value of `t` when `MinMax` is called. If this condition is not met, program will be terminated with a non-zero exit code.

Implementation for Pascal

You need to implement one function `findGap(T, N)` that takes the following parameter and returns an integer of type `Int64`:

- `T` — the subtask number (1 or 2) (`Integer` type)
- `N` — the number of given integers (`LongInt` type)

Your function `findGap` can call procedure `MinMax(s, t, mn, mx)` where the first two parameters `s` and `t` are integers of type `Int64` and the last two paramters `mn` and `mx` are variables **called by reference** of type `Int64`, i.e., `mn` and `mx` are integer variables of type `Int64`. When `MinMax(s, t, mn, mx)` exits, the variable `mn` will have the value of smallest a_i larger than or equal to the value of `s` and the variable `mx` will have the value of largest a_j smaller than or equal to the value of `t`. In case there are no input integers between `s` and `t` (inclusive), then both `mn` and `mx` will have the value `-1`. The value of `s` should be no larger than the value of `t` when `MinMax` is called.

If this condition is not met, the program will be terminated.

Implementation for all

In addition to the standard requirements (time and memory limits, no runtime errors, etc), your submission has to achieve the following in order to solve a testcase:

- your function `findGap` must return the correct answer,
- the cost M associated with calls to function `MinMax` must not exceed the allowed limit (see section Scoring).

Example for C, C++

Consider the case where $N = 4$ and $a_1 = 2, a_2 = 3, a_3 = 6$, and $a_4 = 8$.

The answer, which is 3, can be calculated and thus returned by `findGap` if the following calls to `MinMax` are made:

- `MinMax(1, 2, &mn, &mx)` is called and `mn` and `mx` both have the value 2.
- `MinMax(3, 7, &mn, &mx)` is called and `mn` have the value 3 and `mx` has the value 6.
- `MinMax(8, 9, &mn, &mx)` is called and `mn` and `mx` both have the value 8.

Example for Pascal

Consider the case where $N = 4$ and $a_1 = 2, a_2 = 3, a_3 = 6$, and $a_4 = 8$.

The answer, which is 3, can be calculated and thus returned by `findGap` if the following calls to `MinMax` are made:

- `MinMax(1, 2, mn, mx)` is called and `mn` and `mx` both have the value 2.
- `MinMax(3, 7, mn, mx)` is called and `mn` have the value 3 and `mx` has the value 6.
- `MinMax(8, 9, mn, mx)` is called and `mn` and `mx` both have the value 8.

Scoring

In all subtasks the constraint $2 \leq N \leq 100,000$ holds.

Subtask 1 (30 points): Each call to `MinMax` will add 1 to M. You will receive the full score for the subtask if $M \leq \frac{N+1}{2}$ for all test cases.

Subtask 2 (70 points): Let k be the number of input integers larger than or equal to `s` and smaller than or equal to `t` in a call to `MinMax`. Each call to `MinMax` will add $k + 1$ to M. The final score will be calculated by the following rule: Final score for the subtask is the minimum score you received among all test cases. For a test case, the score is 70 if $M \leq 3N$ and the score is $\frac{60}{\sqrt{\frac{M}{N}+1}-1}$, otherwise.

Experimentation

The sample grader which can be downloaded from the scoring system will read data from standard input. The first line of input should contain two integers, subtask number T, and N. The next line should contain N integers in ascending order. The sample grader will write to standard output the value returned by `findGap` in the first line and the value of M appropriate for the subtask the input test case belongs to.

The following input describes the above example:

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