Singapore Informatics League 2026

Task Statements (Online Contest)

July 2026

Contents

Level 1

Task 1: `Enclose Duck`

Authored by: shoryu386

Statement

There is an $n \times n$ grid containing some walls and some empty cells. A duck is located on an empty cell and can walk to an adjacent cell (it cannot walk diagonally). The duck can leave the grid if it manages to walk to a cell on the edge of the grid. To stop the duck from leaving, you must place walls to enclose it within the grid.

What is the minimum number of walls you need to place to trap the duck?

Input Format

The first line of input contains one integer n.

The following n lines of input each contain n characters. Each character will be either `'X'`, `'.'`, or `'D'`, representing a wall, an empty cell, or a duck, respectively.

Output Format

Output the minimum number of walls that need to be placed to enclose the duck within the grid.

Scoring

For all subtasks:

- There is only one duck in the grid.
- The duck will not be on the edge of the grid.

Subtask 1: $n = 7$

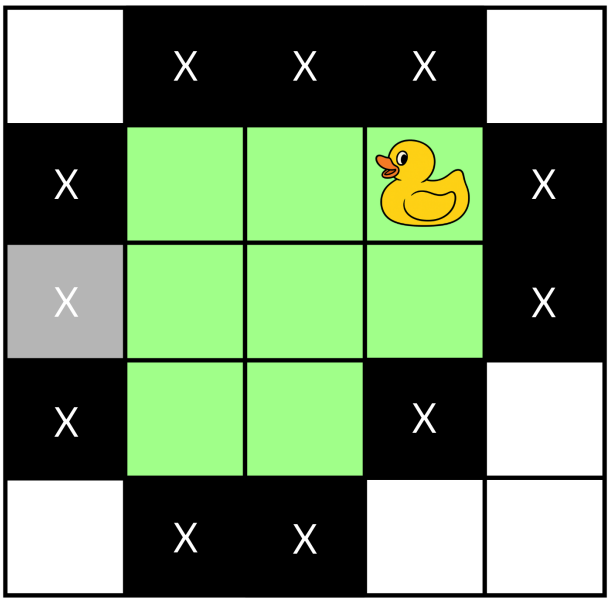
Subtask 2: $n = 10$

Subtask 3: $n = 10$

Examples

Input	Output
5 .XXX. X..DX ....X X..X. .XX..	1
4 ..X. ..XX XD.X ..XX	2

Note



In the first example, represented by the diagram above, by placing one wall at the leftmost position in the third row (marked in grey), the duck will be fully contained within the walls of the grid. Since you only need to place one wall to enclose the duck, the output is 1.

In the second example, you can place one wall directly above the duck and one wall directly below the duck.

Input File Link

Link to the Input Files

Task 2: `Tag`

Authored by: penguin133

Statement

There are n runners running around a circular track divided into l segments, labelled 1 to l in a clockwise manner. The i-th runner initially starts at a unique segment $p[i]$ and will run clockwise at a speed of $v[i]$ segments per second.

At any point, if a runner catches up to another runner in front of him (need not occur at integer points in time), he tags that runner. Subsequently, the tagged runner stops running and leaves the track.

After 10^{100} seconds, how many runners are still running on the track?

Input Format

The first line of input contains two space-separated integers n and l.

The second line of input contains n space-separated integers $p[1], p[2], \ldots, p[n]$.

The third line of input contains n space-separated integers $v[1], v[2], \ldots, v[n]$.

Output Format

Output a single integer, the number of runners still running on the track after 10^{100} seconds.

Scoring

For all subtasks:

- $1 \leq p[i], v[i] \leq l$ for all $1 \leq i \leq n$
- Each runner starts at a unique segment ($p[i]$ are all pairwise distinct).

Subtask 1: $n = 2$

Subtask 2: $n = 10$, $l = 20$, and $v[1] = v[2] = \cdots = v[n]$

Subtask 3: $n = 10$ and $l = 20$

Example

Input	Output
4 10 3 2 5 1 1 2 3 1	1

Note

There are $n = 4$ runners in the example.

- After 1 second, runner 2 is on segment 4, catching up with runner 1 who is also on segment 4. Runner 1 is now tagged out.
- After 3 seconds, runner 3 is on segment 4, catching up with runner 4 who is also on segment 4. Runner 4 is now tagged out.
- After 7 seconds, runner 3 is on segment 6, catching up with runner 2 who is also on segment 6. Runner 2 is now tagged out.

Note that the track is circular. This means that after running through segment l, a runner will run through the track again starting from segment 1.

Input File Link

Link to the Input Files

Task 3: `Evil`

Authored by: penguin133

Statement

You are an evil duck, and you have snuck into the Duck Olympiad in Informatics!

The caterer for the event has prepared n plates of food, numbered from 1 to n, where the i-th plate will be given to the i-th child.

The food on the i-th plate is determined by the character $s[i]$ in the binary string s. If $s[i] =$ `'0'`, it is ice cream, and if $s[i] =$ `'1'`, it is pizza.

You move from the first plate to the last plate, choosing to either eat the food on that plate or not. However, you cannot eat a plate of ice cream if the previous plate eaten was also ice cream, as you will get a stomachache.

A child is happy if his plate of food is not eaten by you.

As you are evil, you want to minimise the length of the longest contiguous row that consists only of happy children. That is, minimise the largest value of $r - l + 1$ for all pairs of l and r such that all children numbered from l to r still have their food, or 0 if there is no valid pair of l and r.

Determine the length of this longest contiguous row if you choose which plates to eat optimally.

Input Format

The first line of input contains one integer n.

The second line of input contains a binary string s of length n.

Output Format

Output a single integer, the length of the longest contiguous row that consists of only happy children if you choose which plates to eat optimally.

Scoring

For all subtasks:

- s is a binary string of length n.
- s contains only `'0'`s and `'1'`s.

Subtask 1: $n = 5$

Subtask 2: $n = 12$

Subtask 3: $n = 40$

Example

Input	Output
7 1000100	1

Note

In the example, one optimal way to eat is to eat from plates 1, 3, 5, and 6.

In this instance, the happiness of the children can be represented by the binary string `0101001`, where `'1'` represents happy and `'0'` represents unhappy. The maximum number of happy children in a row is 1. Note that this may not be the only way to eat that achieves a maximum row length of 1.

Eating from plates 1, 3, 4, 5, and 6 is not allowed: plates 3 and 4 are both ice cream, and eating plate 4 immediately after plate 3 would give you a stomachache.

Input File Link

Link to the Input Files

Task 4: `Six Seven`

Authored by: penguin133

Statement

The Scientific Committee for SIL 2026 is composed of many mature high school graduates. They think of many wonderful problems. Today, they are thinking of `67`.

They have a mysterious number, with some `6`s, some `7`s and no other digits. They first pick a `6` and then pick a `7` to the right of it. They realise that there are at least n ways to do this. Refer to the sample test case for an example.

What is the smallest possible number of digits the mysterious number could have?

Input Format

The first line of input contains one integer n, the number of ways to make `67`.

Output Format

Output the smallest possible number of digits the mysterious number could have.

Scoring

Subtask 1: $n = 6$

Subtask 2: $n = 36$

Subtask 3: $n = 67$

Example

Input	Output
5	5

Note

Consider the number 66767. There are exactly 5 ways to make 67.

- 66767
- 66767
- 66767
- 66767
- 66767

It can be shown that no number with fewer digits can make at least 5 67s.

Input File Link

Link to the Input Files

Task 5: `Circles`

Authored by: oolimry

Statement

There are n circles on a plane. The i-th circle is centered at $(x[i], y[i])$ and has a radius of $r[i]$.

Draw an axis-aligned rectangle such that all circles are contained within it. Here, axis-aligned means the edges of the rectangle are parallel to the x- and y-axes.

What is the minimum area of such a rectangle?

Input Format

The first line of input contains one integer n.

The following n lines of input each contain three space-separated integers. The i-th of these lines contains $x[i]$, $y[i]$, and $r[i]$.

Output Format

Output a single integer, the minimum area of the rectangle.

Scoring

For all subtasks:

- $-3 \leq x[i], y[i] \leq 3$
- $1 \leq r[i] \leq 3$

Subtask 1: $n = 1$

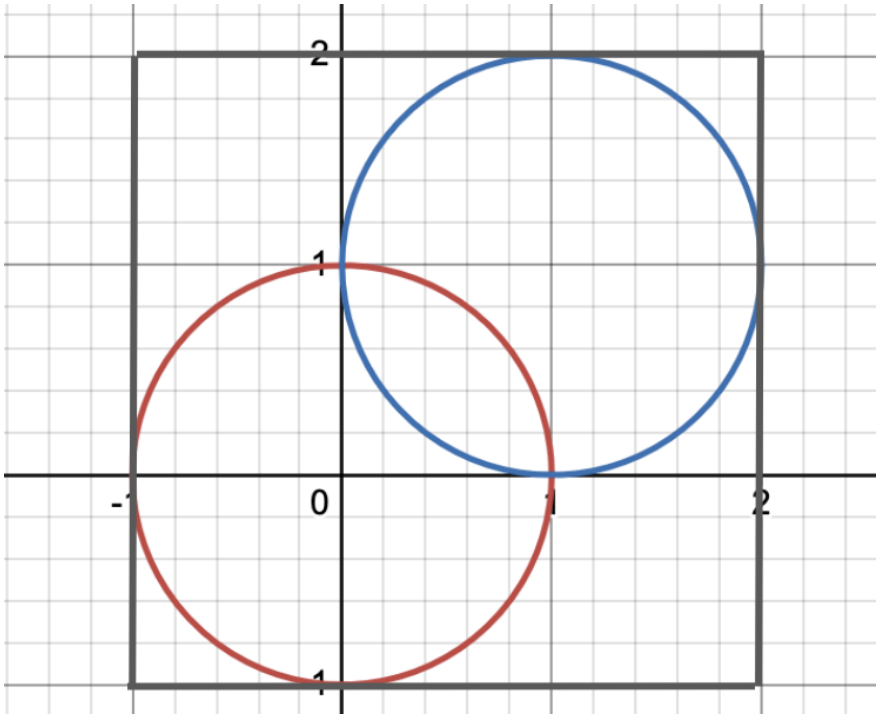
Subtask 2: $n = 3$

Subtask 3: $n = 5$

Example

Input	Output
2 0 0 1 1 1 1	9

Note



As shown in the diagram above, the smallest rectangle covering the circles in the example has area 9.

Input File Link

Link to the Input Files

Task 6: `Virus`

Authored by: oolimry

Statement

There are n people standing in a room, numbered from 1 to n from left to right.

Each person has some number of units of virus. Initially, person 1 has x units of virus, and every other person has 0 units. The value of x is unknown.

A sequence of sneezes then occurs. When a person sneezes, let v be that person's current number of virus units (the value at the moment of the sneeze). The sneezing person's own amount is unchanged, while every other person's amount increases by v.

Any person may sneeze any number of times, in any order, and a person with 0 units may sneeze (this has no effect).

You are given only the final state: after all sneezes, the i-th person has $a[i]$ units of virus.

You are not told who sneezed, how many sneezes occurred, or in what order.

Determine x, the number of units of virus person 1 initially had.

It is guaranteed that for each test case there exists exactly one value of x for which some sequence of sneezes produces the given final state a.

In this problem, there are multiple independent test cases. For each test case, you are to compute x, then output the sum of x over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output the sum of x over all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 3$
- $1 \leq a[i] \leq 10\,000$ for all $1 \leq i \leq n$
- $1 \leq n \leq 10$
- The number of sneezes is at most 6.

Subtask 1: The number of sneezes is at most 2

Subtask 2: $x = 2$ or $x = 3$

Subtask 3: No additional constraints

Example

Input	Output
1 5 252 126 216 180 252	18

Note

In the example, the initial amounts of virus of each person are $[18, 0, 0, 0, 0]$.

After person 1 sneezes, the virus amounts become $[18, 18, 18, 18, 18]$.

After person 4 sneezes, the virus amounts become $[36, 36, 36, 18, 36]$.

After person 3 sneezes, the virus amounts become $[72, 72, 36, 54, 72]$.

After person 4 sneezes, the virus amounts become $[126, 126, 90, 54, 126]$.

After person 2 sneezes, the virus amounts become $[252, 126, 216, 180, 252]$.

At the end, the virus amounts are equal to a, and therefore x must be 18, the amount of virus person 1 initially has.

Input File Link

Link to the Input Files

Task 7: ABBABBA

Authored by: oolimry

Statement

You are given two strings s and t, each consisting only of the characters `'A'` and `'B'`. You wish to turn s into t.

You start with the string s. In each step, you choose a contiguous substring of the current string, copy it, and append the copy to its end.

What is the minimum number of steps required to turn s into t?

Input Format

The first line of input contains the string s.

The second line of input contains the string t.

Output Format

Output a single integer, the minimum number of steps required to turn s into t.

Scoring

For all subtasks:

- $1 \leq |s| \leq |t| \leq 20$, where $|s|$ and $|t|$ represent the lengths of s and t respectively.
- There exists a way to turn s into t.

Subtask 1: s and t consist of only the character `'A'`, $|t| = 17$

Subtask 2: $|t| = 12$

Subtask 3: $|t| = 19$

Example

Input	Output
AB ABBABBA	3

Note

In the example, $s =$ AB can be turned into ABBABBA in 3 steps.

Step 1: $s =$ AB. Copy and paste the substring B to get ABB.

Step 2: $s =$ ABB. Copy and paste the substring A to get ABBA.

Step 3: $s =$ ABBA. Copy and paste the substring BBA to get ABBABBA.

It can be shown that 3 is the minimum number of steps required.

Input File Link

Link to the Input Files

Task 8: `Block Game`

Authored by: siewjh

Statement

Alice and Bob are playing a game on a board with n squares in a row. The rules are as follows:

- Alice and Bob take turns placing blocks on the board. Each block will cover exactly k consecutive squares on the board. Alice goes first.
- Blocks are not allowed to overlap.
- The game only ends when no more blocks can be placed.

Alice wants to maximise the number of empty squares remaining on the board, while Bob wants to minimise it. Assuming both players play optimally, determine the number of empty squares at the end of the game.

Input Format

The first and only line of input contains two space-separated integers n and k.

Output Format

Output one integer, the number of empty squares at the end of the game.

Scoring

Subtask 1: $n = 5, k = 2$

Subtask 2: $n = 6, k = 2$

Subtask 3: $n = 20, k = 4$

Examples

Input	Output
3 1	0
4 2	2

Note

In the first example, Alice and Bob will take turns placing blocks of size-1 until the board is filled. No matter their strategies, the game only ends when the number of empty squares reaches 0.

In the second example, the optimal strategy for Alice is to place a size-2 block in the middle of the board (covering the second and third squares). Bob will not be able to place a block, so the game ends with 2 empty squares.

Input File Link

Link to the Input Files

Level 2

Task 9: `Pringles Sort 2`

Authored by: TheRaptor

Statement

There is a stack of n Pringles chips, each with a unique label. The chip at position i from the top of the stack has label $a[i]$, where $a[1], a[2], \ldots, a[n]$ is a permutation of $1, 2, \ldots, n$.

You have two hands. You will try to eat the chips by performing a sequence of the following actions:

- Grab the topmost chip of the stack with a hand (that hand must be empty).
- Eat the chip currently held in a hand (that hand must be holding a chip).

You wonder if it is possible to eat the chips so that their labels come out in the order $1, 2, \ldots, n$.

In this problem, there are multiple independent test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output a single binary string of length tc. The i-th character is `1` if it is possible to eat the chips of the i-th test case in the order $1, 2, \ldots, n$, and `0` otherwise.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 5000$
- $1 \leq a[i] \leq n$ for all $1 \leq i \leq n$

Subtask 1: $n = 5$

Subtask 2: $a[1] = n$

Subtask 3: No additional constraints

Example

Input	Output
2 4 3 1 2 4 4 2 3 4 1	10

Note

There are two test cases in the example.

In the first test case, one valid way is:

`grab 3 with the left hand`, `grab 1 with the right hand`, `eat 1`, `grab 2 with the right hand`, `eat 2`, `eat 3`, `grab 4 with the left hand`, `eat 4`.

The chips are eaten in the order 1, 2, 3, 4, so the output for this case is `1`.

In the second test case, there is no sequence of actions that lets the chips come out in the order 1, 2, 3, 4, so the output for this case is `0`.

Input File Link

Link to the Input Files

Task 10: `Duck Race`

Authored by: shoryu386

Statement

Two ducks, duck A and duck B, want to compete in a race.

The race has n different sections (e.g. running, swimming, flying).

Duck A and B take $a[i]$ and $b[i]$ minutes respectively to complete 1 km of section i.

For each section, the length is anywhere from $l[i]$ to $r[i]$ kilometres.

The section lengths are not yet fixed. Can you determine the winner, where each duck is evaluated under the most favourable section lengths for it? The winner is determined by the smaller total sum of time taken across all sections.

Let the answer be 0 if A always wins, 1 if B always wins, and 2 if either could win.

In this problem, there are tc independent test cases. For each test case, let ans$[i]$ represent the answer to the i-th test case. Output ans as a single string with no separators.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

The third line of each test case contains n space-separated integers $b[1], b[2], \ldots, b[n]$.

The fourth line of each test case contains n space-separated integers $l[1], l[2], \ldots, l[n]$.

The fifth line of each test case contains n space-separated integers $r[1], r[2], \ldots, r[n]$.

Output Format

Output ans. For example, if $tc = 3$ and A can always win in the first test case, either could win in the second test case, and B can always win in the third test case, output `021`.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 15\,000$
- $1 \leq a[i], b[i], l[i], r[i] \leq 10\,000$ for all $1 \leq i \leq n$
- $l[i] \leq r[i]$ for all $1 \leq i \leq n$
- Under the most favourable section lengths for each duck, the result will not be a tie

Subtask 1: $1 \leq n \leq 2$

Subtask 2: $1 \leq n \leq 15\,000$ and $l[i] = r[i] = 1$ for all $1 \leq i \leq n$

Subtask 3: $1 \leq n \leq 15\,000$

Example

Input	Output
2 3 4 1 7 9 2 3 1 2 4 9 2 9 4 10 20 30 40 20 30 40 50 1 1 1 1 1 1 1 1	20

Input File Link

Link to the Input Files

Task 11: `Heights`

Authored by: TheRaptor

Statement

There are n people, where the i-th person has an initial height of $h[i]$. Every day, you choose exactly one person and increase their height by 1.

A day is valid if, before the increase, there already exists another person whose height is exactly the height that this person will grow into.

Determine the maximum number of valid days that can pass.

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer for the i-th test case, i.e. the maximum number of valid days that can pass. Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $h[1], h[2], \ldots, h[n]$.

Output Format

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$

- $1 \leq n \leq 100\,000$
- $1 \leq h[i] \leq 10^9$

Subtask 1: $tc = 1, n = 5$

Subtask 2: $tc = 4$. $h[i] = h[i-1]$ or $h[i] = h[i-1] + 1$ for all $2 \leq i \leq n$

Subtask 3: No additional constraints

Example

Input	Output
3 5 1 2 2 3 5 3 10 10 10 4 100 101 100 101	10

Note

There are three test cases in the example.

In the first test case, the heights are $[1,2,2,3,5]$.

- Day 1: Increase the person of height 1 to 2. This is valid. The heights become $[2,2,2,3,5]$.
- Day 2: Increase a person of height 2 to 3. This is valid. The heights become $[2,2,3,3,5]$.
- Day 3: Increase a person of height 2 to 3. This is valid. The heights become $[2,3,3,3,5]$.
- Day 4: Increase the remaining person of height 2 to 3. This is valid. The heights become $[3,3,3,3,5]$.

No further valid days are possible, so $\text{ans}[1] = 4$.

In the second test case, the heights are $[10,10,10]$. Increasing any person produces height 11, which is not currently present. Hence no valid day is possible, so $\text{ans}[2] = 0$.

In the third test case, the heights are $[100, 101, 100, 101]$. The maximum number of valid days is 2, so $\text{ans}[3] = 2$.

Hence, the correct output for the example is

$4 \cdot 1 + 0 \cdot 2 + 2 \cdot 3 = 10$.

Input File Link

Link to the Input Files

Task 12: `Mahjong 2`

Authored by: siewjh

Statement

99% of gamblers quit right before they win big. Brian the chronic gambler is back at the mahjong table, determined to win back the money he lost from all his previous games.

As the dealer, he starts the game on his turn with $n \cdot m$ tiles. In this version of mahjong, he wins by forming n sets of m identical tiles. Different sets may use the same type of tile. Brian's hand currently consists of t distinct types of tiles, with $a[i]$ tiles of type i $(1 \leq i \leq t)$.

If Brian's hand is a winning hand at any point of the game (including his initial hand), he wins. Otherwise, he ends his turn by discarding a tile and draws a new tile at the start of his next turn.

Help Brian determine the minimum number of turns it takes for him to win!

In this problem, there are tc independent test cases. For each test case, you are to compute the answer for that test case, and output the sum of the answers across all test cases as a single integer.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains three space-separated integers n, m, and t.

The second line of each test case contains t space-separated integers $a[1], a[2], \ldots, a[t]$.

Output Format

Output a single integer, the sum of the answers across all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 5$
- $1 \leq n \cdot m \leq 10^{18}$
- $1 \leq t \leq 10^6$
- $1 \leq \sum t \leq 2 \cdot 10^6$ over all test cases
- $1 \leq a[i] \leq n \cdot m$ for all $1 \leq i \leq t$
- $\sum_{i=1}^{t} a[i] = n \cdot m$

Subtask 1: $a[i] = 1$ for all $1 \leq i \leq t$

Subtask 2: $m \leq 3$

Subtask 3: No additional constraints

Example

Input	Output
2 2 4 3 2 3 3 3 3 4 6 1 1 1	4

Note

There are two test cases in the example.

In the first test case, $n = 2$, $m = 4$, and $t = 3$. He has 2 tiles of the first type, 3 tiles of the second type and 3 tiles of the third type. To win he needs 2 sets of 4 tiles.

He can win in 2 turns by discarding a tile of the first type, then drawing a tile of the second type, then discarding another tile of the first type, then drawing a tile of the third type. He will have 4 tiles each of the second and third types.

In the second test case, $n = 3$, $m = 3$, and $t = 4$. He has 6, 1, 1, 1 tiles of the first, second, third, and fourth types respectively. To win, he needs 3 sets of 3 tiles.

He can win in 2 turns by discarding a tile of the second type, then drawing a tile of the third type, then discarding a tile of the fourth type, then drawing a tile of the third type. He will have 6 tiles of the first type and 3 tiles of the third type.

Hence, the correct output for the example is $2 + 2 = 4$.

Input File Link

Link to the Input Files

Task 13: `Tokens`

Authored by: JustKitkat

Statement

Every day, Ket may collect 1 token or return 1 token for free. He can also decide not to do anything for that day. On the i-th day ($1 \leq i \leq n$), he must have at least $a[i]$ tokens if $b[i] = 0$, or at most $a[i]$ tokens if $b[i] = 1$ (b is a given binary array).

On any day, in addition to the one free collection or return, Ket may collect or return extra tokens, each of which costs 1. At all times, the number of tokens Ket has must be non-negative.

Starting from Day 1 with 0 tokens, find the minimum total cost he has to incur over all n days.

In this problem, there are tc independent test cases. For each test case, let ans$[i]$ represent the answer to the i-th test case. Output the sum of ans across all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

The third line of each test case contains n space-separated integers $b[1], b[2], \ldots, b[n]$.

Output Format

Output the sum of ans across all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 15\,000$
- $0 \leq a[i] \leq 100\,000$
- $0 \leq b[i] \leq 1$

Subtask 1: $1 \leq n \leq 10$ and $0 \leq a[i] \leq 100$

Subtask 2: $1 \leq n \leq 15\,000$ and $0 \leq a[i] \leq 1000$

Subtask 3: $1 \leq n \leq 15\,000$

Example

Input	Output
3 4 2 3 1 2 0 0 1 0 4 4 2 4 9 0 1 0 1 6 3 1 4 2 5 3 0 1 0 1 0 1	16

Note

There are three test cases in the example.

In the first test case, $n = 4$ with $a = [2,3,1,2]$ and $b = [0,0,1,0]$, so on days 1, 2, and 4 Ket needs at least 2, 3, and 2 tokens respectively, while on day 3 he may have at most 1 token.

On the first day, despite taking 1 free token, Ket still needs to collect 1 more token to reach the minimum of 2. However, on the second day he can just take his free token to reach 3 tokens, so no cost is incurred that day.

On the third day, he can return one token for free; however he still needs to incur one cost to decrease his number of tokens to 1, as he can have at most 1 token on that day.

Finally, he can take one more token on the last day for free to reach the minimum of 2. The total cost incurred is 2, and it is the minimum possible.

It can be shown that the minimum possible cost for the second and third test cases is 5 and 9 respectively. Hence, the output is $2+5+9=16$.

Input File Link

Link to the Input Files

Task 14: `Eat Sleep Repeat`

Authored by: shoryu386

Statement

Shor the Duck is presented with n meals, the i-th of which contains $a[i]$ units of food.

Starting with an empty stomach containing 0 units of food and a total energy of 0 units, Shor follows a specific routine for n days:

Every morning, he chooses whether to eat or sleep, while every night is reserved strictly for sleeping.

If he chooses to eat, he may consume any previously uneaten meal j, adding $a[j]$ units of food to his stomach.

If he chooses to sleep (either during the morning or at night), and his stomach contains at least x units of food, digestion occurs, decreasing the amount of food in his stomach by x units and increasing his total energy by 1 unit. Otherwise, nothing changes.

What is the maximum total energy Shor can accumulate by the end of the n days?

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer for the i-th test case, i.e. the maximum total energy Shor can accumulate in the i-th test case. Output $\text{ans}[1] + \text{ans}[2] + \cdots + \text{ans}[tc]$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains two space-separated integers n and x.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output a single integer, the sum of the maximum total energy Shor can accumulate over all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 200\,000$
- $1 \leq x, a[i] \leq 10^9$

Subtask 1: $a[i] = 1$ for all $1 \leq i \leq n$

Subtask 2: $x = 1$

Subtask 3: No additional constraints

Example

Input	Output
4 1 5 5 2 10 3 4 3 2 5 6 7 5 2 1 1 1 1 1	7

Note

There are four test cases in the example.

In the first test case, Shor has 1 day.

In the morning, he eats the meal containing 5 units of food, bringing his stomach to 5 units of food. At night, he sleeps and digestion occurs, leaving him with 0 units of food and 1 unit of energy.

In the second test case, the total food across both meals is $3+4=7$ units, which is less than $x=10$. No matter what Shor does, his stomach will never contain at least 10 units of food, so digestion never occurs and he ends with 0 units of energy.

In the third test case, Shor has 3 days.

On the morning of day 1, he eats the meal containing 5 units of food, and at night digestion occurs, leaving him with 3 units of food and 1 unit of energy.

On the morning of day 2, he eats the meal containing 6 units of food, bringing the amount of food in his stomach to 9 units. At night digestion occurs, leaving him with 7 units of food and 2 units of energy.

On day 3, instead of eating the remaining meal, he sleeps in the morning so that digestion occurs, leaving him with 5 units of food and 3 units of energy. He then sleeps at night for another digestion to occur, leaving him with 3 units of food and 4 units of energy.

In the fourth test case, Shor has 5 days.

On the morning of day 1, he eats a meal containing 1 unit of food, but since his stomach only has 1 unit of food which is less than $x=2$, digestion does not occur that night.

On the morning of day 2, he eats another meal containing 1 unit of food, bringing his stomach to 2 units, hence digestion occurs that night, leaving him with 0 units of food and 1 unit of energy.

He repeats this pattern on days 3 and 4, ending day 4 with 0 units of food and 2 units of energy. The remaining meal cannot contribute to any further digestion, so on day 5 he simply sleeps and ends with 2 units of energy.

Hence, the total energy accumulated is $1+0+4+2=7$.

Input File Link

Link to the Input Files

Task 15: `Bamboo 3`

Authored by: penguin133

Statement

Ziv needs to cut his bamboos to ensure that their heights are not overly tall.

He has n bamboos in his farm, the i-th bamboo having an initial height of $a[i]$. In one cutting operation, he selects a pair of bamboos i and j ($i \neq j$) where $a[i] + a[j] \geq k$, then decreases the height of **either** bamboo i or bamboo j by 1.

If Ziv can perform this cutting operation any number of times, help Ziv determine the minimum **total height** achievable across all his n bamboos.

In this problem, there are tc independent test cases. For each test case, you are to compute the answer for that individual test case, then output the sum of answers over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains two space-separated integers n and k.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output the sum of answers over all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $2 \leq n \leq 20000$
- $1 \leq k \leq 10^9$
- $1 \leq a[i] \leq 10^9$ for all $1 \leq i \leq n$

Subtask 1: $tc = 2$ and $n = 2$

Subtask 2: $tc = 5$ and all bamboos have the same initial height $(a[1] = a[2] = \cdots = a[n])$

Subtask 3: $tc = 10$

Example

Input	Output
2 3 6 3 4 2 2 100 1 1	8

Note

There are two test cases in the example.

In the first test case, Ziv can perform the following sequence of cutting operations:

- Choose $i = 1$, $j = 2$. This is valid because $a[1] + a[2] = 3 + 4 = 7 \geq 6$. Choose to cut bamboo $i = 1$. Then, $a[1]$ decreases to 2. Now $a = [2, 4, 2]$.
- Choose $i = 2$, $j = 3$. Choose to cut bamboo $j = 3$. Then $a[3]$ decreases to 1. Now $a = [2, 4, 1]$.
- Choose $i = 1$, $j = 2$. Choose to cut bamboo $j = 2$. Then $a[2]$ decreases to 3. Now $a = [2, 3, 1]$.

At this point, no pair of bamboos yields a total height of at least 6. The total height across all 3 bamboos is $2 + 3 + 1 = 6$. It can be proven that this is the minimum sum possible.

In the second test case, the only operation possible is to choose the only pair of bamboos. However, their total height is $1 + 1 = 2$, which is below 100. Hence no operation can be performed and the sum of heights stays at 2.

Hence, the correct output for the example is $6 + 2 = 8$.

Input File Link

Link to the Input Files

Task 16: `Power Sum`

Authored by: sheep, shoryu386

Statement

Sheep was grazing around in her pasture mulling over SIL problems when she suddenly stumbled upon an array a of n distinct positive integers: $a[1], a[2], \ldots, a[n]$.

Pleasantly surprised by this discovery, Sheep decided to come up with her very own SIL problem! She wonders: given a positive integer x, in how many ways can she choose 3 elements from a (one to keep as it is, one to square and one to cube) such that they sum up to exactly x?

More formally, given an array a and integer x, define a valid triplet as an ordered triplet of indices (i, j, k) that are pairwise distinct $(i \neq j, j \neq k, i \neq k)$ and satisfy $a[i] + a[j]^2 + a[k]^3 = x$. Can you count the number of valid triplets in a?

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer for the i-th test case, i.e. the number of valid triplets in the i-th test case. Output $\text{ans}[1] + \text{ans}[2] + \cdots + \text{ans}[tc]$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains two space-separated integers n and x.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output a single integer, the sum of the number of valid triplets from each test case.

Scoring

For all subtasks:

- $tc = 20$
- $1 \leq n \leq 500\,000$
- $1 \leq x \leq 10^9$
- $1 \leq a[i] \leq 10^6$ for all $1 \leq i \leq n$
- All $a[i]$ are pairwise distinct ($a[i] \neq a[j]$ for all $i \neq j$).

Subtask 1: $n \leq 300$ and $x \leq 10^6$

Subtask 2: $n \leq 10\,000$

Subtask 3: No additional constraints

Example

Input	Output
3 3 8 1 2 3 3 33 2 3 10 8 31 1 2 3 4 5 14 21 26	5

Note

There are three test cases in the example.

In the first test case, only one triplet works: we can take 3 as the linear term, 2 as the squared term, 1 as the cubed term to get $3+2^2+1^3 = 3+4+1 = 8$. No other assignment is possible, so this test contributes 1 valid triplet.

In the second test case, although $2+2^2+3^3 = 33$, it would use the element 2 as both the linear and the squared term, which is not allowed. No assignment is possible, so this test contributes 0 valid triplets.

In the third test case, 4 different assignments are possible, so this test contributes 4 valid triplets:

$26+2^2+1^3 = 26+4+1 = 31$

$21+3^2+1^3 = 21+9+1 = 31$

$14+4^2+1^3 = 14+16+1 = 31$

$14+3^2+2^3 = 14+9+8 = 31$

Hence, the correct output for the example is $1+0+4 = 5$.

Input File Link

Link to the Input Files

Level 3

Task 17: `Keys`

Authored by: JustKitkat

Statement

Ket the Cat works at a food warehouse and has been asked to test whether a key fits the warehouse door.

The key consists of teeth 1 to n from left to right, where the i-th tooth has a height of $h[i]$ units. The keyhole consists of slots 1 to n, where the i-th slot has a depth of $d[i]$ units and the opening of the keyhole is on the right side.

Initially, the key is placed on the right side of the keyhole and is then pushed to the left into the keyhole as far as possible.

The key can continue moving inward only if every tooth currently passing through a slot has height less than or equal to the depth of that slot. If at any point a tooth is taller than the slot it is passing through, the insertion stops.

Let c denote the maximum distance the key can be inserted.

In this problem, there are multiple independent test cases. For each test case, you are to compute c, then output the sum of c over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $h[1], h[2], \ldots, h[n]$.

The third line of each test case contains n space-separated integers $d[1], d[2], \ldots, d[n]$.

Output Format

Output the sum of c over all test cases.

Scoring

For all subtasks:

- $1 \leq h[i] \leq 10^9$ for all $1 \leq i \leq n$

Subtask 1: $tc = 2$ and $1 \leq n \leq 5$

Subtask 2: $tc = 10$ and $n = 100$

Subtask 3: $tc = 10$ and $n = 10^5$

Examples

Input	Output
1 2 1 4 5 4	2
2 4 8 4 5 9 3 14 15 9 2 10 1 5 9	3

Note

There is one test case in the first example. In that test case, the key can be inserted all the way into the keyhole as all its teeth are shorter than the slots of the keyhole. Hence, it can be inserted to a depth of $c = 2$ units.

There are two test cases in the second example. In the first test case, the key can only be inserted to a depth of $c = 3$ units. In the second test case, the key cannot be inserted at all ($c = 0$). Therefore, the output is $3 + 0 = 3$.

Input File Link

Link to the Input Files

Task 18: `Reels`

Authored by: penguin133

Statement

You are watching Instagram reels. There are n reels in order and you know that the i-th reel is $t[i]$ seconds long. For each reel, you can either scroll past the reel or watch it fully. However, you are only allowed to scroll past reels k times.

There will be q queries. For the i-th query, you have $x[i]$ seconds total to watch reels. Find the maximum number of reels you can watch fully.

In this problem, there are multiple independent test cases. For each test case, you are to compute the sum of the answers to all of its queries, then output the sum of these values over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains three space-separated integers n, k, and q.

The second line of each test case contains n space-separated integers $t[1], t[2], \ldots, t[n]$.

The next q lines of each test case each contain one integer. The i-th of these lines contains $x[i]$.

Output Format

Output a single integer, the sum of the answers to all queries across all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n, q \leq 10^6$
- $0 \leq k \leq n$
- $1 \leq t[i], x[i] \leq 10^9$

Subtask 1: $k = 0$

Subtask 2: $1 \leq n, q \leq 5000$

Subtask 3: No additional constraints

Examples

Input	Output
1 5 1 3 3 1 4 2 1 1 5 15	8
2 2 0 1 3 2 4 3 2 2 5 3 8 10 8	5

Note

In the first example, there is only one test case with $q = 3$ queries. For the first query, 1 reel can be watched by skipping the first reel and watching the second. For the second query, 2 reels can be watched by simply watching the first two. For the third query, there is sufficient time to watch all 5 reels. Hence, the correct output is the sum $1 + 2 + 5 = 8$.

Input File Link

Link to the Input Files

Task 19: `Dictator`

Authored by: siewjh

Statement

Supreme dictator Pavement has gathered n of his political opponents, labelled $1, 2, \ldots, n$. Being inspired by his favourite algorithm, Stalin sort, he decides to play a game with them.

We define Stalin sort of an array as a procedure that scans its elements from left to right while remembering the most recently kept element. Each subsequent element is compared with the most recently kept element: if it is smaller it is removed from the array, else it is kept. All removed elements will form a new array in the order in which they originally appeared.

First, he arranges them in a permutation $a[1], a[2], \ldots, a[n]$.

He does one round of Stalin sort on array a with the removed people forming array b.

He then does one round of Stalin sort on array b with the removed people forming array c.

This process repeats until no one is removed.

Pavement wants to know how many times Stalin sort is performed. Please help him!

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer for the i-th test case, i.e. the number of times Stalin sort is performed. Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

Output Format

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Scoring

For all subtasks:

- $1 \leq tc \leq 5$
- $1 \leq n \leq 10^6$
- The array a is a permutation of $1, 2, \ldots, n$.

Subtask 1: $1 \leq n \leq 5000$

Subtask 2: There exists exactly one value of i such that $a[i] < a[i+1]$

Subtask 3: No additional constraints

Example

Input	Output
2 3 1 3 2 5 5 4 3 2 1	12

Note

There are two test cases in the example.

In the first test case, $n = 3$ and $a = [1, 3, 2]$.

For the first Stalin sort on a: since 1 is the first element, we keep it. We then compare the next element 3 to the most recently kept element 1. Since 3 is greater than 1, we keep 3. We then

compare the next element 2 with the most recently kept element 3. Since 2 is smaller than 3, we remove 2. In the end $a = [1,3]$ and a new array $b = [2]$ is created.

For the Stalin sort on b: since 2 is the only element, it is kept and hence no elements are removed. Since no elements are removed, the process ends.

Stalin sort is performed a total of 2 times, hence the answer for the first test case is 2.

For the second test case, $n = 5$ and $a = [5,4,3,2,1]$.

After the first Stalin sort, $a = [5]$ and $b = [4,3,2,1]$.

After the second Stalin sort, $b = [4]$ and $c = [3,2,1]$.

After the third Stalin sort, $c = [3]$ and $d = [2,1]$.

After the fourth Stalin sort, $d = [2]$ and $e = [1]$.

In the fifth Stalin sort, no elements are removed from e and the process ends.

Hence, the answer for the second test case is 5.

Hence the final answer for the sample input is $2 \cdot 1 + 5 \cdot 2 = 12$.

Input File Link

Link to the Input Files

Task 20: `House Visit`

Authored by: siewjh

Statement

Jayden lives on a number line at position x. A day lasts y seconds. Jayden has the day off today and wants to complete some errands.

There are n errands, and errand i requires him to visit position $p[i]$ at least once in the day. Note that Jayden may need to complete multiple errands at the same position. There may also be errands at Jayden's house which he completes immediately.

Jayden starts the day at home and needs to return home at the end of the day at second y. Every second, he can move 1 position in any direction to complete his errands. Furthermore, he knows in advance that an inspector will visit his home at m times, with the j-th time being $t[j]$ seconds into the day. Jayden must be at home at the specified times. Note that the inspector may visit his home multiple times in the same second.

Find the maximum number of errands that Jayden can complete while still being home for every inspection.

In this problem, there are multiple independent test cases. For each test case, you are to compute the maximum number of errands, then output the sum of the maximum number of errands over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains four space-separated integers n, m, x, and y.

The second line of each test case contains n space-separated integers $p[1], p[2], \ldots, p[n]$.

The third line of each test case contains m space-separated integers $t[1], t[2], \ldots, t[m]$.

Output Format

Output a single integer, the sum of the maximum number of errands Jayden can complete over all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq x, y \leq 10^9$
- $1 \leq p[i] \leq 10^9$ for all $1 \leq i \leq n$
- $1 \leq t[1] \leq t[2] \leq \cdots \leq t[m] \leq y$

Subtask 1: $x = 1$ and $1 \leq n, m \leq 10^6$

Subtask 2: $1 \leq n, m \leq 5000$

Subtask 3: $1 \leq n, m \leq 10^6$

Example

Input	Output
2 3 2 5 5 2 4 6 1 3 4 2 3 10 1 1 7 8 10 10	4

Note

There are two test cases in the example.

Consider the first test case. Jayden can stay at home at seconds 0 and 1. At second 2, he can go to position 4 to complete the second errand, and take another second to return home at second 3. He can then go to position 6 at second 4 to complete the third errand and return home afterwards, completing a total of 2 errands.

In the second test case, Jayden can spend 2 seconds to go to position 1 and complete the first 2 errands, then spend another 2 seconds to return home by second 4. He can stay at home for the remaining time and hence completes 2 errands.

The total number of errands completed is $2 + 2 = 4$.

Input File Link

Link to the Input Files

Task 21: `Spinning Table`

Authored by: TheRaptor

Statement

You are in a Chinese restaurant with your n friends, seated around a spinnable table!

n dishes are placed around the edge of the table, evenly spaced with 1 unit of distance between adjacent dishes. Each dish has a unique dish label $x[i]$ $(1 \leq x[i] \leq n)$.

For simplicity's sake, your friends are indexed from 1 to n in clockwise order. Initially, person i sits in front of the dish with label $x[i]$. Person i wants the dish with label i, and takes the dish in front of them as soon as it is the one they want.

Every second, the table can turn one unit clockwise or one unit anticlockwise, where a unit is the distance required for a dish to travel between two adjacent people. Find the minimum number of seconds such that everyone will be able to get the dish they want at some point in time.

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer to the i-th test case. Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $x[1], x[2], \ldots, x[n]$.

Output Format

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 10^6$
- x is a permutation of $1, 2, \ldots, n$

Subtask 1: $1 \leq n \leq 10$

Subtask 2: $1 \leq n \leq 5000$

Subtask 3: $1 \leq n \leq 10^6$

Example

Input	Output
2 3 3 1 2 7 6 1 3 4 2 5 7	7

Note

There are two test cases in the example.

In the first test case, by rotating the table one unit in the anticlockwise direction, all 3 people can get their food at the same time.

In the second test case, the initial orientation of the dishes is as follows. In each layout, the top row gives the person index and the bottom row gives the dish label; `X` marks a dish that has already been taken.

Person	1	2	3	4	5	6	7
Food	6	1	3	4	2	5	7

Persons 3, 4, and 7 get their food immediately.

Next, the table moves 1 unit anticlockwise.

Person	1	2	3	4	5	6	7
Food	1	X	X	2	5	X	6

Persons 1 and 5 get their food.

Next, the table moves 1 unit anticlockwise.

Person	1	2	3	4	5	6	7
Food	X	X	2	X	X	6	X

Person 6 gets their food.

Lastly, the table moves 1 unit anticlockwise, and person 2 will get their food.

Person	1	2	3	4	5	6	7
Food	X	2	X	X	X	X	X

This takes 3 turns, or 3 seconds.

Hence, the correct output for the example is $1 \cdot 1 + 3 \cdot 2 = 7$.

Input File Link

Link to the Input Files

Task 22: A Certain Scientific Committee

Authored by: siewjh

Statement

A certain contest's scientific committee has proposed n problems. There are two testers who have tried all n problems and given their ratings for each of them, with the ratings for problem i being $a[i]$ and $b[i]$ for tester 1 and tester 2 respectively. It is guaranteed no two problems have the same values of $a[i]$ and $b[i]$.

The scientific committee will choose a certain real number weight x where $0 \leq x \leq 1$ to determine the problems' final ratings, with the final rating of problem i being $x \cdot a[i] + (1-x) \cdot b[i]$. All problems will be sorted and ranked based on their final rating. As the scientific committee hates resolving ties, x must be chosen such that no two problems have the same final rating.

As the one who proposed problem 1, siewjh wants problem 1 to have the best possible rank. Help siewjh find the best rank that problem 1 can achieve!

In this problem, there are tc independent test cases. Let $\text{ans}[i]$ represent the answer for the i-th test case, i.e. the best rank that problem 1 can achieve. Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The second line of each test case contains n space-separated integers $a[1], a[2], \ldots, a[n]$.

The third line of each test case contains n space-separated integers $b[1], b[2], \ldots, b[n]$.

Output Format

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $2 \leq n \leq 10^6$
- $2 \leq \sum n \leq 5 \cdot 10^6$ over all test cases
- $1 \leq a[i], b[i] \leq 10^9$ for all $1 \leq i \leq n$

Subtask 1: $a[i] > a[1]$ for all $i > 1$

Subtask 2: $n \leq 5000$

Subtask 3: No additional constraints

Example

Input	Output
2 3 3 2 7 4 6 3 3 5 2 6 5 6 2	4

Note

There are two test cases in the example.

In the first test case, $n = 3$, $a = [3,2,7]$ and $b = [4,6,3]$. When $x = 1$, the final ratings are:

Problem 1: $1 \cdot 3 + (1-1) \cdot 4 = 3$

Problem 2: $1 \cdot 2 + (1-1) \cdot 6 = 2$

Problem 3: $1 \cdot 7 + (1-1) \cdot 3 = 7$

In this case, the rank for problem 1 is 2. It can be shown there does not exist a value of x such that problem 1 has the highest final rating, i.e. rank 1. Hence our answer for this test is 2.

In the second sample test case $n = 3$, $a = [5,2,6]$ and $b = [5,6,2]$. When $x = 0.4$, the final ratings are:

Problem 1: $0.4 \cdot 5 + (1 - 0.4) \cdot 5 = 5$

Problem 2: $0.4 \cdot 2 + (1 - 0.4) \cdot 6 = 4.4$

Problem 3: $0.4 \cdot 6 + (1 - 0.4) \cdot 2 = 3.6$

Since we are able to achieve rank 1 when $x = 0.4$, our answer for this test is 1.

Hence our final answer for this input is $1 \cdot 2 + 2 \cdot 1 = 4$.

Input File Link

Link to the Input Files

Task 23: `Bookshelf`

Authored by: TheRaptor

Statement

There is a row of n books on a bookshelf. Each book i has width $w[i]$ and height $h[i]$.

John Bookshelf is satisfied if and only if for every two adjacent books, the book to the right is either at least as wide or at least as tall as the book to the left. Formally, for every i between 1 and $n-1$ inclusive, $w[i] \leq w[i+1]$ or $h[i] \leq h[i+1]$.

In one operation, you can either insert a book of any dimensions, or remove a book from the bookshelf.

Find the minimum number of operations required to satisfy John Bookshelf.

In this problem, there are multiple independent test cases. For each test case, you are to compute the minimum number of operations, then output the sum of the minimum number of operations over all test cases.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains one integer n.

The following n lines of each test case each contain two space-separated integers. The i-th of these lines contains $w[i]$ and $h[i]$.

Output Format

Output the sum of the minimum number of operations over all test cases.

Scoring

For all subtasks:

- $1 \leq tc \leq 10^6$
- $1 \leq w[i], h[i] \leq 10^9$

Subtask 1: $1 \leq n \leq 300$, $w[i]$ and $h[i]$ are strictly decreasing

Subtask 2: $1 \leq n \leq 300$

Subtask 3: $1 \leq n \leq 10^6$

Example

Input	Output
2 4 1 2 2 2 4 4 3 3 4 4 44 3 33 2 22 1 11	4

Note

In the first sample test case, an optimal solution is to insert a book of $(5, 1)$ after book 3.

In the second sample test case, an optimal solution is to remove books 1, 3 and 4.

The final answer is $1 + 3 = 4$.

Input File Link

Link to the Input Files

Task 24: Pawns

Authored by: sheep

Statement

Brian and Ryan are playing a game on a board with n squares in a row, labelled from 1 to n. On these squares, there are m pawns, labelled from 1 to m. The i-th pawn is at position $d[i]$ and has colour $c[i]$: if $c[i]$ is `'B'`, it is a blue pawn belonging to Brian; if $c[i]$ is `'R'`, it is a red pawn belonging to Ryan. Initially, no two pawns are on the same square.

In this game, Brian moves first. On Brian's turn, he will pick a blue pawn at square p such that $p+1$ does not contain a red pawn and $p+1 \leq n$, then move it there. After that, it will be Ryan's turn.

Similarly, on Ryan's turn, he will pick a red pawn at square q such that $q-1$ does not contain a blue pawn and $q-1 \geq 1$, then move it there. After that, it will be Brian's turn.

The first player who cannot move loses. If Brian and Ryan play perfectly, who will win?

In this problem, there are tc independent test cases. For each test case, you are to determine who wins, then output a string of length tc representing who won in each test case.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains two space-separated integers n and m.

The following m lines of each test case each contain a character and an integer, separated by a space. The i-th of these lines contains $c[i]$ and $d[i]$.

Output Format

Output the string of length tc denoting who wins over all test cases. If Brian wins in the i-th test case, the i-th character is `'B'`. Otherwise, if Ryan wins in the i-th test case, the i-th character is `'R'`.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq n \leq 10^9$
- $1 \leq m \leq 200\,000$
- $c[i] \in \{\texttt{'R'}, \texttt{'B'}\}$ for all $1 \leq i \leq m$
- $1 \leq d[i] \leq n$ for all $1 \leq i \leq m$
- $d[1] < d[2] < \cdots < d[m]$

Subtask 1: For all $1 \leq i \leq m-1$, if $c[i] = \texttt{'R'}$, then $c[i+1] = \texttt{'R'}$

Subtask 2: For all $1 \leq i \leq m-1$, if $c[i] = \texttt{'R'}$, then $c[i+1] = \texttt{'B'}$

Subtask 3: No additional constraints

Example

Input	Output
3 4 2 B 1 R 4 6 3 B 1 B 3 R 6 9 6 R 2 B 3 B 4 R 6 B 7 R 9	RBB

Note

There are three test cases in the example.

In the first test case, Brian can only move his blue pawn at position 1 to position 2. Then, Ryan will move his red pawn from position 4 to position 3. Brian has no moves left, so Ryan wins.

In the second test case, Brian can move his blue pawn at position 3 to position 4. Then, Ryan can only move his red pawn at position 6 to position 5, and after Brian moves his blue pawn at position 1 to position 2, Ryan has no moves left, so Brian wins.

In the third test case, Brian can move his blue pawn at position 4 to position 5. Then, it can be shown that no matter how Ryan moves his pawns, Brian will always win.

However, if Brian chooses any other first move, Ryan can move his red pawn at position 6 to position 5. In that case, Ryan will win regardless of the moves Brian makes.

Input File Link

Link to the Input Files

Task 25: `Turbines`

Authored by: TheRaptor

Statement

Brian is the mayor of Penguinland! Penguinland is a city that can be modelled as a tree with n nodes (or estates) numbered from 1 to n, rooted at node 1. Brian wants to generate electricity for the residents of Penguinland. To do this, he can build at most m wind turbines across the nodes in the city.

Each turbine can be built on any node, but each node can contain at most 1 turbine. To generate electricity, a strong wind starts blowing from node 1 with strength k and flows downstream, away from the root, to every node in the city. When the wind reaches a node without a turbine, it passes through unchanged. When it reaches a turbine and arrives with strength s, that turbine generates s units of electricity for the city, and the wind continuing downstream into that node's subtree has its strength decreased by 1.

Help Brian determine the maximum number of units of electricity that Penguinland can generate if he builds the turbines optimally.

In this problem, there are tc independent test cases.

For each test case, let $\text{ans}[i]$ represent the answer to the i-th test case.

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Input Format

The first line of input contains one integer tc, the number of test cases.

The first line of each test case contains three space-separated integers n, m, and k.

The next $n-1$ lines of each test case each contain two space-separated integers $u[i]$ and $v[i]$, describing the edges of the tree. This means that nodes $u[i]$ and $v[i]$ are connected by an edge.

Output Format

Output $\text{ans}[1] \cdot 1 + \text{ans}[2] \cdot 2 + \cdots + \text{ans}[tc] \cdot tc$.

Scoring

For all subtasks:

- $1 \leq tc \leq 10$
- $1 \leq k \leq 10^9$
- $1 \leq m \leq n \leq 200\,000$
- $1 \leq \sum n \leq 10^6$ across all test cases
- $1 \leq u[i], v[i] \leq n$ for all $1 \leq i \leq n-1$
- $u[i] \neq v[i]$ for all $1 \leq i \leq n-1$

Subtask 1: The tree is a line; $u[i] = i, v[i] = i+1$ for all $1 \leq i \leq n-1$

Subtask 2: $n \leq 2000$

Subtask 3: No additional constraints

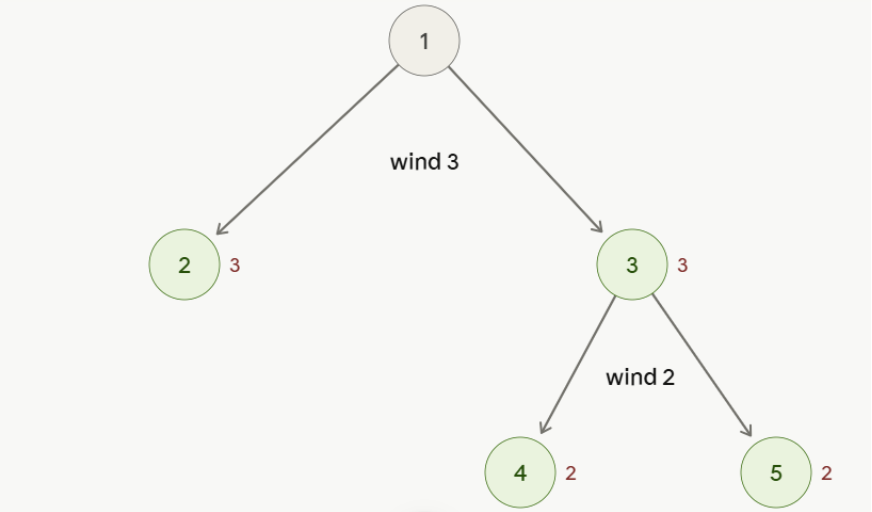
Example

Input	Output
3 6 3 5 1 2 2 3 3 4 4 5 5 6 5 5 3 1 2 1 3 3 4 5 3 12 8 4 1 2 2 3 3 4 4 5 5 6 1 7 1 8 1 9 1 10 1 11 1 12	125

Note

There are three test cases in the example.

Consider the second test case in the example. Penguinland is illustrated as the tree in the diagram below.



Nodes 2, 3, 4, and 5 (highlighted in green) are placed with a turbine.

Wind starts blowing from node 1 with strength 3. As there is no turbine on node 1, wind continues blowing down to nodes 2 and 3 with strength 3.

Nodes 2 and 3 then generate 3 units of electricity each. Wind then blows down from node 3 towards nodes 4 and 5, but with strength decreased to 2. Nodes 4 and 5 each then generate 2 units of electricity.

The total number of units of electricity generated is $3+3+2+2=10$. It can be shown that this is the maximum possible.

It can be shown that the answer for the first and third test cases is 12 and 31 respectively. Hence, the correct output for the example is $1 \cdot 12 + 2 \cdot 10 + 3 \cdot 31 = 125$.

Input File Link

Link to the Input Files

Optimisation Task

`Watercup Robot`

Authored by: shoryu386

Statement

There are $n+1$ points on a number line, numbered 0 to n. A robot with a water cup attached to its head starts at position 0. The cup is initially empty.

The cup can hold at most c units of water. **In subtasks 1 and 2, c is equal to the sum of all $a[i]$, so the cup is large enough to hold all available water.**

There are m water sources. The i-th source is at position $x[i]$ and initially contains $a[i]$ units of water. It can transfer at most $b[i]$ units of water per second to the robot, and only while the robot is at the same position as the source.

At position n there is a water tank of unlimited capacity. Your goal is to maximise the amount of water in the tank at the end of t seconds.

At the start of each second, the robot may move to any **integer** position from 0 to n (in either direction), or stay where it is. If the robot moves a distance of d, it spills d^2 units of water from its cup; if the cup contains fewer than d^2 units, the cup becomes empty instead.

At the end of each second:

- If the robot is at the tank (position n), it empties its entire cup into the tank.
- Otherwise, if the robot is at the same position as the i-th water source, which has r units of water remaining, and the cup currently holds w units, then $\min(r, b[i], c-w)$ units of water are transferred from the source into the cup.

This is an optimisation task: you are not expected to find an optimal solution. Better outputs receive higher scores, and sub-optimal outputs still receive partial scores (see Scoring).

Visualiser

We highly recommend using the visualiser, which you can open from this page. It replays your output on any of the subtask inputs and scores it exactly like the grader. It also has an **Edit Mode**, which lets you generate solutions interactively: click where the robot should go each second, then copy the resulting moves and submit them.

Input Format

The first line of input contains four space-separated integers n, m, t, and c.

The following m lines of input each contain three space-separated integers. The i-th of these lines contains $x[i]$, $a[i]$, and $b[i]$.

Output Format

Output t space-separated integers. The i-th integer is the position the robot moves to at the start of second i. Every position must be between 0 and n, inclusive.

Scoring

Your raw score for a subtask is the amount of water in the tank at the end of t seconds. Raw scores are then converted to normalised scores by comparison against all other teams: for each subtask, the team with the highest raw score receives the full 15 points, and teams with lower raw scores receive progressively fewer points.

Specifically, your team's normalised score for each subtask is computed according to the following formula.

$$10 \cdot \left(1 - \frac{\text{rank} - 1}{N}\right)^{1.25} + 5 \cdot \left(1 - \left(1 - \frac{\text{score}}{\text{best}}\right)^{0.4}\right)$$

N	The total number of teams, regardless of whether they attempted the subtask ($N = 177$)
rank	Your team's rank for that subtask by raw score (from 1 to N). Teams with equal raw scores share the best (smallest) rank of the tied group. Teams that did not attempt the subtask are also ranked.
score	Your team's raw score for that subtask
best	The highest raw score for that subtask across all teams

As a special case, a team that does not attempt a subtask (i.e. makes no valid submission for it) always receives 0 points for that subtask.

For all subtasks:

- $1 \leq x[i] < n$ for all $1 \leq i \leq m$
- $x[i] < x[i+1]$ for all $1 \leq i \leq m-1$ (sources are listed in increasing order of position)

In subtasks 1 and 2, c is equal to the sum of all $a[i]$.

Examples

Input	Output
12 3 12 36 3 15 4 5 11 6 10 10 2	3 3 5 5 7 9 10 10 10 10 10 12

Input File Link

Link to the Input Files