L: Flora and Beauty

Time limit per test: 1 second
Memory limit per test: 256 megabytes

Flora decided to give Bella a pair of flowers from the garden. There are m flowers in the garden, and the i-th flower has a beauty value of v_i. Bella has a peculiar preference—she doesn't necessarily want the two most beautiful flowers. Instead, she wants a pair of flowers that have the **maximum possible beauty difference**!

Your task is to write a program that calculates two values:

1. ***The maximum beauty difference*** that Flora can choose between two flowers.
2. ***The number of ways*** Flora can select these flowers. Two selections are considered different *if and only if* at least one flower is chosen in one case but not in the other.

Input

- The first line of input contains a single integer m ($2 \le m \le 2 \times 10^5$) — the number of flowers in the garden.
- The second line contains m space-separated integers v1,v2,...,v_m ($1 \le v_i \le 10^9$), where each v_i represents the beauty value of a flower.

Output

Print a single line containing two integers: the **maximum beauty difference** and the **number of ways** this can be achieved.

Example:

input
2 1 2
output
1 1

input
4 5 5 5 5
output
0 6

H: Journey by Bus

Time limit: 1 seconds

Memory limit: 256 megabytes

The journey by bus to our hometown feels like a return to roots, each mile bringing a sense of nostalgia and anticipation. As the bus drives through the roads and familiar landmarks, memories from the past flood back—childhood streets, old shops, and the quiet hum of family life. The closer we get, the more the rush of city life fades, replaced by the comforting sights of green fields and rustic houses. By the time we finally reach our destination, the bus ride has transformed into a peaceful transition, a reminder of where we come from and the comfort of returning home.

Assume you are waiting at the bus stand with a given number of passengers and your serial number. Your task is to determine the number of buses required to carry all the passengers, identify your bus number, and calculate how many empty seats the last bus will have.

Note that, A bus can transport 45 passengers at once.

Input:

A line containing two integer numbers, **n** ($1 \leq n \leq 1000$), the number of passengers waiting at the bus stop, and **s** ($1 \leq s \leq n$), your serial number.

Output:

Print a single line containing three integer numbers: the number of buses required to carry all the passengers, your bus number, and the number of empty seats on the last bus.

Sample Input:

60 15

Sample Output:

2 1 30

E: Battle of Brains

Time limit: 1 seconds

Memory limit: 256 megabytes

The "Battle of Brain" is a programming contest organized by the Institute of Science and Technology (IST), designed to bring together talented students to showcase their coding skills and problem-solving abilities. Participants compete by solving complex algorithmic challenges within a limited time frame, testing their knowledge of programming concepts, data structures, and logical reasoning. This contest encourages innovation and creativity, providing a platform for students to demonstrate their expertise and compete against their peers. Teams of three students to solve complex, real-world problems within a five-hour timeframe, testing their problem-solving and programming skills. Through events like the "Battle of Brain," IST aims to promote a culture of excellence in technology and coding among aspiring computer scientists.

Your task is to determine the total number of participants in this contest.

Input:

A single line containing an integer, n ($1 \leq n \leq 100$), representing the number of teams participating in the 'Battle of Brains".

Output:

Print a single line containing the total number of participants in this contest.

Sample Input:

5

Sample Output:

15

J: Generate the Sequence

Time Limit per test: 2 seconds

Memory Limit per test: 256 megabytes

You are given three positive integers m,x, and y. You need to construct a sequence S of length m consisting of lowercase Latin letters such that every contiguous subsequence of length x contains exactly y distinct letters. It is guaranteed that a valid sequence exists.

You need to solve p independent test cases.

Input

The first line of input contains an integer p ($1 \le p \le 2000$) — the number of test cases. Then, p test cases follow.

Each test case consists of three space-separated integers m,x, and y

($1 \le x \le m \le 2000$, $1 \le y \le \min(26,x)$), where:

- m is the total length of the required sequence.
- x is the length of the contiguous subsequence to be considered.
- y is the required number of distinct characters in each contiguous subsequence of length x.

It is guaranteed that the sum of all m values across test cases does not exceed 2000 ($\sum m \le 2000$).

Output

For each test case, print a valid sequence S of length m consisting of lowercase Latin letters that satisfies the given constraints. If there are multiple valid sequences, print any one of them. It is guaranteed that at least one valid sequence exists.

Example:

input
4 7 5 3 6 1 1 6 6 1 5 2 2
output
abcabca aaaaaa aaaaaa ababa

F: An Outstanding Problem

time limit per test: 2 seconds
memory limit per test: 256 megabytes

Last evening, Ms Anika reminded his teacher Rahman about a contest problem due tomorrow. He hasn't noted anything yet, so Rahman decided to randomly copy-paste substrings from the prompt to make the problem set.

More formally, the prompt is a string s of initial length n. Rahman will perform the copy-pasting operation c times. Each operation is described by two integers l and r, which means that Rahman will append letters $S_lS_{l+1}...S_r$ to the end of string s. Note that the length of s increases after this operation.

Finally, Rahman needs to be able to see what has been noted. After copying, Rahman will ask q queries: given an integer k, determine the k-th letter of the final string s.

INPUT

The first line contains a single integer t — the number of test cases.

The first line of each test case contains three integers n, c, and q — the length of the initial string s, the number of copy-pasting operations, and the number of queries, respectively.

The second line of each test case contains a single string s of length n. It is guaranteed that s only contains lowercase English letters.

The following c lines describe the copy-pasting operation. Each line contains two integers l and r. It is also guaranteed that r does not exceed the current length of s.

The last q lines of each test case describe the queries. Each line contains a single integer k. It is also guaranteed that k does not exceed the final length of s.

It is guaranteed that the sum of n and q across all test cases does not exceed $2*10^5$ and 10^4, respectively.

OUTPUT

For each query, print the k-th letter of the final string s.

CONSTRAINTS

- $1 \le t \le 10$
- $1 \le n \le 2*10^5$
- $1 \le c \le 40$
- $1 \le q \le 10^4$
- $1 \le l \le r \le 10^{18}$
- $1 \le k \le 10^{18}$

EXAMPLE

2 4 3 3 pcis 1 4 5 7 3 8 11 10 12 7 3 3	p c i a n h

dhanmondi 2 3 3 4 2 9 9 11 12	
1 4 3 3 dstdhanmondi 1 4 5 7 3 8 1 10 12	i n i

Note

In the first test case, the copy-paste process is as follows.

- The first step is pasting string pcist at the end, yielding the string pcistpcis.
- The second step is pasting string pcistpcis at the end, yielding the string pcistpcistpc.
- The third step is pasting string pcistpcistpc at the end, yielding the string pcistpcistpcistpci.

K: Problem Calculate

time limit per test: 1 seconds

memory limit per test: 64 megabytes

Given a set of digits **S**, and an integer **n**, you have to find how many **n**-digit integers are there, which contain digits that belong to **S** and the difference between any two adjacent digits is not more than two.

Input

Input starts with an integer **T**, denoting the number of test cases.

Each case contains two integers, **m** and **n**. The next line will contain **m** integers **(from 1 to 9)** separated by spaces. These integers form the set **S** as described above. These integers will be distinct and given in ascending order.

Output

For each case, print the case number and the number of valid **n**-digit integers in a single line.

Constraints

- $T \leq 500$
- $1 \leq m < 10$
- $1 \leq n \leq 10$

EXAMPLE

3 3 1 3 5 3 1 1 2 3 3 3 1 4 7	Case 1: 7 Case 2: 3 Case 3: 3

Notes

For the first case the valid integers are

1. 11
2. 13
3. 31
4. 33
5. 66

D: The Data Pipeline Challenge

Time Limit : 1 second

Memory Limit : 256 megabytes

Imagine a large communication network represented as an **undirected** graph with `N` nodes and `M` edges. Each edge has a **capacity**, indicating the maximum amount of data that can flow through it. You are given a set of `Q` data transfer requests. Each request is defined by a source node `u`, a destination node `v`, and a required data volume `d`.

Your task is to determine, for each data transfer request, whether it is possible to transfer the required data volume `d` from the source node `u` to the destination node `v` without exceeding the capacity of any edge in the network.

Input Format:

1. The first line contains **three integers** `N`, `M`, and `Q`, representing the number of nodes, the
   number of edges, and the number of data transfer requests, respectively.
2. The next `M` lines each contain **three integers** `a`, `b`, and `c`, representing an edge between nodes `a` and `b` with capacity `c`.
3. The next `Q` lines each contain **three integers** `u`, `v`, and `d`, representing a data transfer request from node `u` to node `v` with required data volume `d`.

Output Format:

For each data transfer request, output "**YES**" if the transfer is possible, and "**NO**" otherwise.

Constraints:

- 1 <= N <= 100
- 1 <= M <= N * (N - 1) / 2
- 1 <= Q <= 1000
- 1 <= a, b, u, v <= N
- 1 <= c, d <= 10^9

Sample Input	Sample Output
4 5 3 1 2 10 1 3 5 2 3 8 2 4 12 3 4 7 1 4 9 2 3 10 1 3 4	YES NO YES

B: Connecting the Nation

Time limit: 1 seconds

Memory limit: 256 megabytes

In the interim government of Bangladesh, Dr. Muhammad Yunus, the Chief Executive, is determined to improve the country's connectivity. The nation has N cities, some of which are already connected by bidirectional roads, while others remain isolated.

Driven by a vision of progress, Dr. Yunus wants to ensure that all cities are connected, enabling travel between any two cities. However, constructing new roads is costly, so he seeks to minimize the number of additional roads required to create a fully connected nation.

Your task is to assist Dr. Muhammad Yunus in calculating the **minimum** number of new roads that need to be built to connect all cities, ensuring there is a path between every pair of cities.

Input:

- First line contains two integers N and M — the number of cities (numbered 1 to N) and the number of roads.
- Next M lines each contain two integers u and v — representing a road between city u and city v.

Output:

- Print a single integer — the **minimum number of additional roads** required to connect all the cities.

Constraints:

- $1 \leq N \leq 300000$
- $0 \leq M \leq 300000$
- $1 \leq u, v \leq N$

Example:

Input 1:

5 2

1 2

3 4

Output 1:

2

Explanation:

• There are 3 connected groups: {1,2}, {3,4}, {5}.

• To connect all cities, you need **2 more roads.** ○ One to connect {1,2} and {3,4}

○ One to connect {5} to the rest.

Sample Input:

4 0

Sample Output 1:

3

G: The Queen's Gambit

Time Limit : 1 second
Memory Limit : 64 megabytes

Problem Statement:

You are given an N x N chessboard and a list of cursed (blocked) cells where queens cannot be placed. Your task is to place exactly N queens on the board such that:

No two queens attack each other — that is, no two queens share the same row, column, or diagonal.
No queen is placed on a cursed (blocked) cell.
Your job is to count how many valid arrangements of queens are possible on the given board.

Note:
Queens can attack through cursed cells. These cells do not block lines of attack. A cursed cell is simply a position where you cannot place a queen.

Input Format:

The first line contains two integers N and M ($1 \le N \le 12$, $0 \le M \le N*N$) — the size of the board and the number of cursed cells.

The next M lines each contain two integers r and c (0-based indexing), representing a cursed cell at row r and column c.

Output Format:

Print a single integer that indicates the number of valid ways to place N queens under the given conditions.

Constraints:

- $1 \le N \le 12$
- $0 \le M \le N*N$
- Cursed cells are guaranteed to be unique.

Sample Input:	Sample Output:
Test Case 1: 4 2 1 1 2 2 Test Case 2: 5 3 0 0 2 2 3 4	Output: 1 Output: 6

Explanation:

Sample 1: The cursed cells block two positions, reducing options. There is only 1 valid configuration left.
Sample 2: There are 6 valid ways to place 5 queens such that none are on the 3 cursed cells and no two attack each other.

A: Problem: The Treasure Hunt

time limit per test: 1 seconds
memory limit per test: 256 megabytes

Problem Statement:

You are an adventurer on a treasure hunt in a mysterious land. The land is divided into n regions, each containing a certain amount of treasure (or possibly traps, represented by negative values). You have a magical compass that can guide you to the most profitable path, but it has a limitation: it can only guide you through a sequence of at most k consecutive regions.

Your goal is to find the maximum amount of treasure you can collect by traveling through a sequence of regions, but you cannot visit more than k regions in a row. The regions are arranged in a straight line, and you must move from one region to the next without skipping any.

Input:

- The first line contains two integers n and k ($1 \le k \le n \le 10^5$), the number of regions and the maximum number of consecutive regions you can visit.
- The second line contains n integers A[1], A[2], ..., A[n] ($-10^4 \le A[i] \le 10^4$), where A[i] represents the amount of treasure (or trap) in the i-th region.

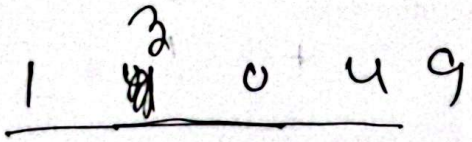
Output:

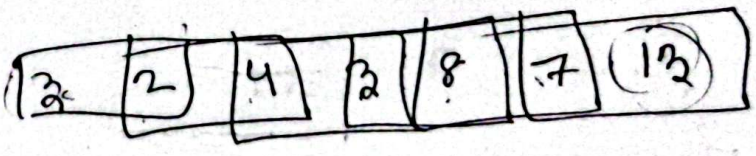
- Print a single integer, the maximum amount of treasure you can collect by visiting a sequence of at most k consecutive regions

Test Cases:

Test Case	Input (n, k)	Array A	Output	Explanation
1	5, 3	[1, 2, -3, 4, 5]	9	Best subarray is [4, 5] with sum 9. Length (2) ≤ k (3).
2	6, 2	[-1, -2, -3, -4, -5, -6]	-1	Best subarray is [-1] with sum -1. Length (1) ≤ k (2).
3	7, 4	[1, -2, 3, -4, 5, -6, 7]	7	Best subarray is [7] with sum 7. Length (1) ≤ k (4).
4	4, 4	[-1, -2, -3, -4]	-1	Best subarray is [-1] with sum -1. Length (1) ≤ k (4).
5	5, 2	[-1, 2, -1, 3, -1]	3	Best subarray is [3] with sum 3. Length (1) ≤ k (2).

6	5, 1	[1, 2, 3, 4, 5]	5	Best subarray is [5] with sum 5. Length (1) ≤ k (1).
7	3, 3	[10, -5, 20]	25	Best subarray is [10, -5, 20] with sum 25. Length (3) ≤ k (3).
8	4, 2	[0, 0, 0, 0]	0	Best subarray is any single 0. Length (1) ≤ k (2).
9	5, 5	[1, -1, 1, -1, 1]	1	Best subarray is any single 1. Length (1) ≤ k (5).
10	6, 3	[1, 2, 3, -2, -3, -4]	6	Best subarray is [1, 2, 3] with sum 6. Length (3) ≤ k (3).
11	7, 2	[3, -1, 2, -1, 5, -1, 6]	6	Best subarray is [6] with sum 6. Length (1) ≤ k (2).
12	8, 4	[-2, 1, -3, 4, -1, 2, 1, -5]	6	Best subarray is [4, -1, 2, 1] with sum 6. Length (4) ≤ k (4).
13	9, 3	[1, 2, 3, -5, 6, -7, 8, -9, 10]	10	Best subarray is [10] with sum 10. Length (1) ≤ k (3).
14	10, 5	[5, -4, -3, 8, -2, 7, -1, 9, -10, 11]	16	Best subarray is [8, -2, 7, -1, 9] with sum 16. Length (5) ≤ k (5).
15	5, 2	[-1, -2, -3, -4, -5]	-1	Best subarray is [-1] with sum -1. Length (1) ≤ k (2).





C: The Watering Game

Time Limit : 1 second
Memory Limit : 256 megabytes

A gardener has **N** plants in a row, numbered from **1** to **N**. Each plant **i** needs a specific amount of water **W[i]** to thrive (**1 ≤ W[i] ≤ 10^6**). The gardener has a watering can with a total capacity of **C** units of water (**1 ≤ C ≤ 10^9**).
However, there's a catch: if two consecutive plants are both watered, they compete for sunlight and only the plant with the higher water requirement counts toward the gardener's score, while the other is wasted (**scores 0**). The gardener's score is the number of plants that thrive (receive at least their required water and aren't wasted due to competition).

Your task is to determine the maximum score the gardener can achieve by distributing the **C** units of water across the plants, ensuring:

Each watered plant gets at least its required water **W[i]**.
No two consecutive watered plants both contribute to the score unless one is wasted.
You can give a plant more water than required, but it doesn't increase the score beyond **1** per thriving plant.

Input Format

First line: **N C** (number of plants and water capacity).
Second line: **N** space-separated integers W[1], W[2], ..., W[N] (water requirement for each plant).

Output Format

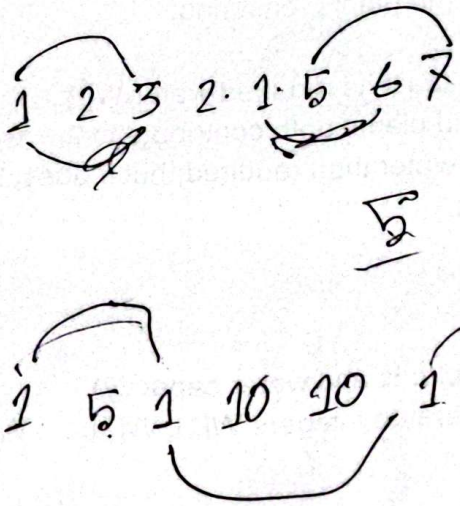
A single integer that denotes the maximum number of plants that can thrive (score).

Constraints

- 1 ≤ N ≤ 10^5
- 1 ≤ C ≤ 10^9
- 1 ≤ W[i] ≤ 10^6

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

Explanation:
Plants: [2, 3, 4, 5], Capacity = 10.
Water plant 1 (2), skip 2, water 3 (4), skip 4.
Total water = 2 + 4 = 6 ≤ 10, score = 2 (no consecutive competition).



I: Recursive Song Playlist

Time Limit: 2 seconds

Memory Limit: 256 MB

You are developing a recursive algorithm to generate valid playlists from a list of songs. Each song has a genre, and no two adjacent songs in the playlist can be of the same genre. Also, the playlist must be exactly **K** songs long.

Your task is to count how many distinct playlists of length **K** can be formed such that each song can be used at most once, and no two adjacent songs are from the same genre.

Input

The first line contains two integers, **N** and **K**, where **N (1 < N <= 10)** is the total number of songs available, and **K (1 < K < N)** is the required length of the playlist.

The next **N** lines contain two strings each: **ID** and **GENRE**, which consist of a minimum **of 1** to a maximum of **10** lowercase English letters.

All song IDs are unique. Genres may repeat.

Output

Print a single integer — the total number of valid playlists of length K following the above rules.

Sample

Input	Output
4 3 a1 pop a2 rock a3 pop a4 jazz	16