Fundamentals of Algorithms

Grade of Computer Science. Group I

Examination Extraordinary Call, July 5, 2021.

Rules for conducting the exam

1. You have to program solutions for each of the three exercises, test them and submit them to the automatic judge accessible at the address `http://exacrc`. For the submission evaluation, the judge has only the test data from the problem statement.

2. Write comments that explain your solution, justify why it has been done this way and help to understand it. Calculate the complexity of all the functions you implement.

3. At the judge you will identify yourself with the username and password you received at the beginning of the exam. The username and password you have been using during the continuous evaluation are **not** valid.

4. Write your **first and last name** in a comment on the first line of each file you upload to the judge.

5. Your solutions will be evaluated by the teacher independently of the verdict of the automatic judge. For this purpose, the teacher will **only** take into account the last submission you have made for each exercise.

1. **(4 points)** Given a vector v of $n \geq 0$ positive integers and two natural numbers $k, l \geq 0$ so that $k \leq n$ and the first k positions of v contain even numbers, it is required to count how many segments of k length satisfy that they they have less or equal($\leq$) than l positions with odd numbers.

    1. (0.25 points) Define a predicate $noMore(v, i, j, l)$ that is true if and only if the quantity of odd numbers of the vector v between the positions i (included) y j (excluded) is less or equal than l (assuming $0 \leq i \leq j \leq v.size()$).
    2. (0.5 points) Using the predicate $noMore$, specify a function that given v, n, k y l so that $k, l \geq 0$, $k \leq n$ and the first k positions of v contain even numbers, it returns the number of segments of k length that has less or equal($\leq$) than l positions with odd numbers. If $k = 0$ the function should return $n + 1$.
    3. (2.5 points) Design and implement an efficient iterative algorithm that solves the proposed problem.
    4. (0.5 points) Write the invariant of the loop to prove the correctness of the loop and provide a coordinate function.
    5. (0.25 points) Indicate the asymptotic cost of the algorithm in the worst case and adequately justify your answer.

Input

The input starts with a line containing the number of test cases. Each test case will contain the value of the number of n elements, the value of k and l, and then in another line the elements of the sequence.

Output

For each test case the program will write a line with the number of segments requested in the instructions.

Input Example

```
5
8 4 2
2 6 4 8 1 10 3 2
8 4 0
2 6 4 8 9 3 1 7
4 2 5
2 4 1 3
3 0 2
1 3 5
3 1 1
2 3 6
```

Output Example

```
5
1
3
4
3
```

2. (2.5 points) A digit of a natural number n is said to be *respectful* if it is greater than or equal to all the digits of n that are more significant than it. The most significant digit is therefore always respectful. For example, in $n = 1234$ all digits are respectful, while in 3159 only 1 is not respectful. It is desired to count how many digits of n are respectful.

It is requested:

1. (1.5 points) Write an efficient recursive algorithm to solve the problem for a given number n. It is not allowed to store in an auxiliary vector the digits of the number.
2. (1 point) Write the recurrence corresponding to the cost of the recursive function using the number of digits of n as the size of the problem. Also indicate to which order of asymptotic complexity this cost belongs.

Input

The input starts with a line containing the number of test cases. Each test case will contain the number n in a line.

Output

For each test case, the program will write the number of respectful digits in that case.

Input Example

```
5
1234
0
32
303
4329
```

Output Example

```
4
1
1
2
2
```

3. (3.5 points) Given an integer M and a vector v of $n \geq 0$ natural numbers, design a backtracking algorithm that determines whether there is a way to insert **between** the n numbers of the vector (as they are placed in the vector) sum or subtraction operations in such a way that obtains the number M as the final result. If, for example, $M = 9$ y $v = [1, 3, 2, 1, 4, 3, 2, 1]$, the algorithm should return "yes" since $1+3+2-1+4+3-2-1 = 9$. However, if $v = [2, 1, 3]$ and $M = -1$, the result should be "no" since $2+1+3 = 6$, $2+1-3 = 0$, $2-1+3 = 4$ y $2-1-3 = -2$. Implement a backtracking algorithm that solves the problem. The algorithm should stop as soon as it finds a way to obtain M. Branching that allows to reduce the number of explored nodes will be valued.

Input

The input starts with a line containing the number of test cases. Each test case will contain the integer M, the number n of natural numbers, and in another line the sequence of those numbers.

Output

For each test case the program will write NO on one line if M cannot be obtained, and otherwise it will write YES.

Input Example

```
5
9 8
1 3 2 1 4 3 2 1
-1 3
2 1 3
4 2
1 5
0 0
0 1
3
```

Output Example

```
YES
NO
NO
YES
NO
```