Greedy

Q: What is greedy?
A: Some kind of strategy or rule, that you use in an optimization problem; Useful thought in most problems

Q: When to use greedy?
A: When some problem that ask you to optimize something or check something whether possible… and you cannot do DP or flow or iterate over all possible solutions; When you think something is intuitive and makes sense; When you don’t know the solution and time is running out

Greedy

Q: How to decide whether a greedy solution is correct or not
A: Typically, prove not doing the greedy strategy will not give better results; Sometimes induction is also useful for proof; Sometimes, when you cannot find counter example

A. Median of an Array

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array a of n integers.

The *median* of an array $q_1, q_2, \ldots, q_k$ is the number $p_{\lceil \frac{k}{2} \rceil}$, where p is the array q sorted in non-decreasing order. For example, the median of the array $[9, 5, 1, 2, 6]$ is 5, as in the sorted array $[1, 2, 5, 6, 9]$, the number at index $\lceil \frac{5}{2} \rceil = 3$ is 5, and the median of the array $[9, 2, 8, 3]$ is 3, as in the sorted array $[2, 3, 8, 9]$, the number at index $\lceil \frac{4}{2} \rceil = 2$ is 3.

You are allowed to choose an integer i ($1 \leq i \leq n$) and increase a_i by 1 in one operation.

Your task is to find the minimum number of operations required to increase the median of the array.

Note that the array a may not necessarily contain distinct numbers.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of test cases. Then follows the description of the test cases.

The first line of each test case contains a single integer n ($1 \leq n \leq 10^5$) — the length of the array a.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the array a.

It is guaranteed that the sum of the values of n over all test cases does not exceed $2 \cdot 10^5$.

B. Yet Another Coin Problem

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You have 5 different types of coins, each with a value equal to one of the first 5 triangular numbers: 1, 3, 6, 10, and 15. These coin types are available in abundance. Your goal is to find the minimum number of these coins required such that their total value sums up to exactly n.

We can show that the answer always exists.

Input

The first line contains one integer t ($1 \le t \le 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains an integer n ($1 \le n \le 10^9$) — the target value.

C. Theofanis' Nightmare

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Theofanis easily gets obsessed with problems before going to sleep and often has nightmares about them. To deal with his obsession he visited his doctor, Dr. Emix.

In his latest nightmare, he has an array a of size n and wants to divide it into non-empty subarrays[†] such that every element is in exactly one of the subarrays.

For example, the array $[1, -3, 7, -6, 2, 5]$ can be divided to $[1][-3, 7][-6, 2][5]$.

The Cypriot value of such division is equal to $\Sigma_{i=1}^{k} i \cdot \text{sum}_i$ where k is the number of subarrays that we divided the array into and sum_i is the sum of the i-th subarray.

The Cypriot value of this division of the array $[1][-3, 7][-6, 2][5] = 1 \cdot 1 + 2 \cdot (-3 + 7) + 3 \cdot (-6 + 2) + 4 \cdot 5 = 17$.

Theofanis is wondering what is the **maximum** Cypriot value of any division of the array.

[†] An array b is a subarray of an array a if b can be obtained from a by deletion of several (possibly, zero or all) elements from the beginning and several (possibly, zero or all) elements from the end. In particular, an array is a subarray of itself.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

Each test case consists of two lines.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the size of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^8 \le a_i \le 10^8$) — the elements of the array.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Archaeologists have discovered exciting clay tablets in deep layers of Alutila Cave. Nobody was able to decipher the script on the tablets, except for two symbols that seem to describe nested structures not unlike opening and closing parentheses in LISP. Could it be that humans wrote programs thousands of years ago?

Taken together, the tablets appear to describe a great piece of work — perhaps a program, or an epic, or even tax records! Unsurprisingly, after such a long time, the tablets are in a state of disorder. Your job is to arrange them into a sequence so that the resulting work has a properly nested parenthesis structure. Considering only opening and closing parentheses, a properly nested structure is either

- $()$, or
- (A), where A is a properly nested structure, or
- AB, where A and B are properly nested structures.

Input

The first line of input contains one integer n ($1 \leq n \leq 10^6$), the number of tablets. Each of the remaining n lines describes a tablet, and contains a non-empty string of opening and closing parentheses; symbols unrelated to the nesting structure are omitted. The strings are numbered from 1 to n in the order that they appear in the input. The input contains at most 10^7 parentheses.