National Collegiate Programming Contest (NCPC) 2023

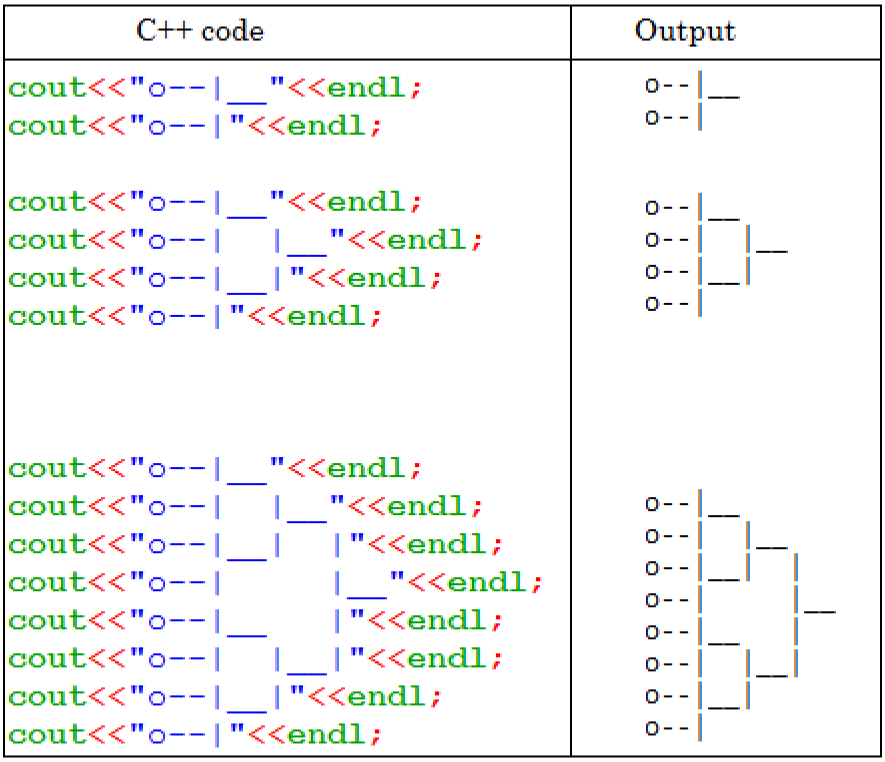
Department of Computer Science and Engineering, Jahangirnagar University

(Mock Contest)

Problem A

Proper Binary Tree

Have you heard the name of Pirate Captain Jack Dumb? He is the most wanted blacklisted pirate who lives in Black Iceland. Recently he has learned about proper binary tree (a binary tree where every node other than leaves has two children, also known as full binary tree). With his little knowledge in C++ he wrote code that can display proper binary tree with height 1, 2 and 3.



C++ code

```
cout<<"o--|__"<<endl;
cout<<"o--|"<<endl;

cout<<"o--|__"<<endl;
cout<<"o--|  |__"<<endl;
cout<<"o--|__|"<<endl;
cout<<"o--|"<<endl;

cout<<"o--|__"<<endl;
cout<<"o--|  |__"<<endl;
cout<<"o--|__|  |"<<endl;
cout<<"o--|     |__"<<endl;
cout<<"o--|__   |"<<endl;
cout<<"o--|  |__|"<<endl;
cout<<"o--|__|"<<endl;
cout<<"o--|"<<endl;
```

Output

```
o--|__
o--|

o--|__
o--|  |__
o--|__|
o--|

o--|__
o--|  |__
o--|__|  |
o--|     |__
o--|__   |
o--|  |__|
o--|__|
o--|
```

Figure: Proper/full binary tree with height 1,2 and 3

It's the night before ACM ICPC Regional Contest. Festive mode is going on amongst the organizer, participants and spectators. Rumours in the air-"Pirate Captain was seen in this city few hours ago". Just at that time, Pirate Captain Jack Dumb broke into the judge's room and hijacked one of your balloons. But Pirate Captain promised that, he will give back your balloon during contest if you can draw a proper/full binary tree with height **N** for him.

Input

The input begins with a single positive integer **T(≤13)** on a line by itself indicating the number of the cases. Each of the next **T** lines contains an integer **N(1≤ N ≤13)** denoting the required height of proper binary tree.

Output

For each test case, print the test case number in a single line followed by desired proper binary tree. You should follow the exact format as sample input/output. Note that, trailing spaces are not allowed.

Sample Input

```
2
1
2
```

Output for Sample Input

```
Case 1:
o--|__
o--|
2
Case 2:
o--|__
o--|  |__
o--|__|
o--|
```

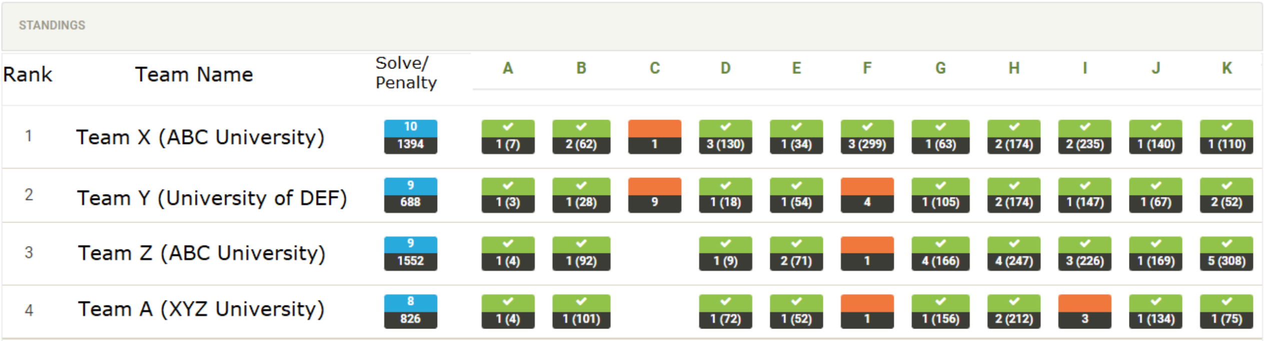
Problem B

NCPC Standings

Students get the chance to interact with different institutions while showcasing their problem-solving, programming, and teamwork abilities through the National Collegiate Programming Contest (NCPC).
The ranking of teams for this contest will be decided based on the following rules.

- The team with the higher number of solved problems will get a higher rank. Rank 1 is considered the highest rank.
- In case of a tie between two teams with the same number of problems solved, the higher rank is determined by penalty time. The team with a lower penalty time will get the higher rank.

The web page that shows all team's ranks in the contest from higher rank to lower rank, is known as the contest standings page. Usually, the contest standing page shows a live rank list consisting of the team name, number of problems solved, total penalty time, problem status (solved or unsolved), number of attempts on each problem, and so on. An example rank list page is shown in the following figure.



STANDINGS

Rank	Team Name	Solve/ Penalty	A	B	C	D	E	F	G	H	I	J	K
1	Team X (ABC University)	10 1394	✓ 1 (7)	✓ 2 (62)	1	✓ 3 (130)	✓ 1 (34)	✓ 3 (299)	✓ 1 (63)	✓ 2 (174)	✓ 2 (235)	✓ 1 (140)	✓ 1 (110)
2	Team Y (University of DEF)	9 688	✓ 1 (3)	✓ 1 (28)	9	✓ 1 (18)	✓ 1 (54)	4	✓ 1 (105)	✓ 2 (174)	✓ 1 (147)	✓ 1 (67)	✓ 2 (52)
3	Team Z (ABC University)	9 1552	✓ 1 (4)	✓ 1 (92)		✓ 1 (9)	✓ 2 (71)	1	✓ 4 (166)	✓ 4 (247)	✓ 3 (226)	✓ 1 (169)	✓ 5 (308)
4	Team A (XYZ University)	8 826	✓ 1 (4)	✓ 1 (101)		✓ 1 (72)	✓ 1 (52)	1	✓ 1 (156)	✓ 2 (212)	3	✓ 1 (134)	✓ 1 (75)

This year the organizer will not provide any rank list in the contest standing page.

The contest is running. There are still 3 hours remaining in this contest. As a contestant, you are only informed that there are **P** problems in the problem set, your team has already solved **S** problems, and your current team rank is **R**. You have to determine whether there is a chance by any means for your team to become champion after the end of the contest.

Input

The first line will contain a single integer **T (1 ≤ T ≤ 18150)**. Each test case will contain three integers **P (1 ≤ P ≤ 10)**, **S (0 ≤ S ≤ P)** and **R (1 ≤ R ≤ 165).**

Output

For each test case, Print the case number in the format “**Case X:**” in a single line (without quotes), where **X** is the case number, followed by “**Yes**” if there is a chance for your team, “**No**” otherwise.

Sample Input

```
2
10 0 165
10 10 1
```

Output for Sample Input

```
Case 1: Yes
Case 2: Yes
```

Problem C

Make A Beautiful Array

After a long break, we are having a programming contest in our country and it's the preliminary contest for ICPC Dhaka Regional, 2020. Alice and Bob are here as visitors on this beautiful evening. They brought some colorful balloons for Dhaka as a token of appreciation. The host of the contest decided to keep those balloons in quarantine for a couple of hours due to Covid19 issues. But, they did not want to go to the contest room with their bare hands and decided to present an array to the contestants.

So, Bob built an array with **N** integers within a very short time and wants to make it as beautiful as possible. He believes the beauty of an array is equal to its MCSS (Maximum Consecutive Subarray Sum). MCSS of an array is the largest subarray sum of any contiguous subarray indexed from ***i*** to ***j*** such that $1 \le i \le j \le N$.

Now, he asked Alice to perform a series of operations (if required) on the array to maximize its beauty. On each operation, she can select any two index ***i*** and ***j*** such that $1 \le i < j \le N$ and reverse the subarray (***i*** and ***j*** inclusive).

As Alice is in a hurry she wants your help to do so. You have to maximize the beauty of the array performing the given operation as many times as it is required. If there are several ways to get the maximum beauty, you have to choose the one that requires the minimum number of moves.

Input

The first line of the input contains an integer **T (1≤T≤100)**, denoting the number of test cases. Each of the next T test cases has two lines of input. First line of each test case contains an integer **N (1≤N≤30000)** denoting the size of the array and the second line contains **N** space separated integers $a_1, a_2, a_3, \ldots\ a_N$ **(-100000 ≤**$a_1, a_2, a_3, \ldots, a_N$**≤100000**) representing the array.

Output

For each test case print the test case number followed by the maximum consecutive subarray sum that can be achieved and the minimum number of moves. See the sample I/O section for output formatting.

Sample Input	Output for the Sample Input

2
4
1 -10 2 2
2
1 2
	Case 1: 5 1
	Case 2: 3 0

Explanation

Case 1: Initial array is {1, -10, 2, 2}, Consider i =1 and j =2 and reverse the subarray.
After performing the operation, the array will become {-10, 1, 2, 2}. So, Maximum consecutive subarray sum is 1 + 2 + 2 = 5.
Case 2: Initial array is {1, 2} and you do not need to perform any operation.
NB: Large dataset, use fast I/O methods.