LEBANESE AMERICAN UNIVERSITY
School of Arts and Science
Department of Computer Science and Mathematics

CSC 310: Algorithms and Data Structures

Lab II

15. Sep. 2015

Important Note: All the problems in this lab should be solved using recursion. If your program produces a correct output, but you are not using recursion, then it will be considered a wrong answer. Moreover, input should be read from a file named after the problem name. For example, problem 1 has the input file “problem1.in”.

Problem 1

Given an integer *N*, write a program that outputs the factorial value of *N*.

Input

Your program will be tested against multiple test cases.
Each test case is an integer *N*. If *N* is negative, then your program should stop.

Output

For each test case, output the factorial value.

Sample Input	**Sample Output**
5	120
3	6
-1	

Problem 2

Given an integer *N*, write a program that outputs the N^{th} Fibonacci number. The Fibonacci sequence is a recurring sequence where each number is equal to the sum of its two predecessors.

1 1 2 3 5 8 13

Please note that the sequence is zero-based, meaning that *Fibonacci(0) = 1*.

Input

The first line of input is an integer *T* representing the number of test cases.
Each test case is made up of an integer *N* as described above.

Output

For each test case, output the Fibonacci value.

Sample Input

```
3
1
4
5
```

Sample Output

```
1
5
8
```

Problem 3

Given a binary search tree, write a program that outputs the values in the tree using a preorder traversal.

Input

The first line of input is an integer *T* representing the number of test cases.

Each test case is made up of two lines. The first line has an integer *N* representing the number of nodes in the tree. The second line contains *N* integers representing the values to insert.

Output

For each test case, output the tree using preorder traversal.

Sample Input

```
3
7
25 13 10 30 15 27 37
4
6 7 8 9
6
10 7 15 13 4 6
```

Sample Output

```
25 13 10 15 30 27 37
6 7 8 9
10 7 4 6 15 13
```

Problem 4

Given a string, write a program that checks if the string is a palindrome. A palindrome is a string that can be read the same way forwards and backwards.

Input

Your program will be tested against multiple test cases.

Each test case is made up of a string *str*. The program should stop if *str* has one or more ‘-‘ characters.

Output

For each test case, output “Yes” if the string is a palindrome, else output “No”.

Sample Input

```
tacocat
lannisters
rassar
---
```

Sample Output

```
Yes
No
Yes
```

Problem 5

Given an integer array, write a program that checks if it is possible to divide the integers into two groups, so that the sum of the two groups is the same, with these constraints: all the values that are multiple of 5 must be in one group, and all the values that are a multiple of 3 (and not a multiple of 5) must be in the other.

Input

The first line of input is an integer *T* representing the number of test cases.

Each test case is made up of two lines. The first line has an integer *N* representing the number of integers in the array. The second line contains *N* integers representing the values in the array.

Output

For each test case, output “Yes” if the grouping is possible, else output “No”.

Sample Input

```
3
2
1 1
3
1 1 1
4
3 3 5 1
```

Sample Output

```
Yes
No
Yes
```

Problem 6

Given a binary search tree, write a program that outputs the height of the tree. Note that the root is at height 0.

Input

The first line of input is an integer *T* representing the number of test cases.

Each test case is made up of two lines. The first line has an integer *N* representing the number of nodes in the tree. The second line contains *N* integers representing the values to insert.

Output

For each test case, output the height of the tree.

Sample Input

```
3
25 13 10 30 15 27 37
4
6 7 8 9
6
10 7 15 13 4 6
```

Sample Output

```
2
3
3
```

Problem 7

Given a sequence of numbers, write a program that finds the length of the longest increasing subsequence. The subsequence is not necessarily contiguous nor unique. For example, given the following subsequence,

10, 22, 9, 33, 21, 50, 41, 60

The longest increasing subsequence in the previous example is 10, 22, 33, 50, and 60. The length of the subsequence is 5.

Input

The first line of input is an integer *T* representing the number of test cases.

Each test case is made up of two lines. The first line has an integer *N* representing the number of integers in the sequence. The second line contains *N* integers representing the values in the sequence.

Output

For each test case, output the length of the longest increasing subsequence.

Sample Input

```
2
8
10 22 9 33 21 50 41 60
5
5 8 12 6 9
```

Sample Output

```
5
3
```