OJ 10303

How Many Trees?

Prof. Edson Alves - UnB/FGA

OJ 10303 – How Many Trees?

A binary search tree is a binary tree with root k such that any node v reachable from its left has $\text{label}(v) < \text{label}(k)$ and any node w reachable from its right has $\text{label}(w) > \text{label}(k)$. It is a search structure which can find a node with label x in $O(n \log n)$ average time, where n is the size of the tree (number of vertices).

Given a number n, can you tell how many different binary search trees may be constructed with a set of numbers of size n such that each element of the set will be associated to the label of exactly one node in a binary search tree?

Entrada e saída

Input

The input will contain a number $1 \leq i \leq 1000$ *per line representing the number of elements of the set.*

Output

You have to print a line in the output for each entry with the answer to the previous question.

Exemplo de entrada e saída

Entrada	Saída
1 2 3	1 2 5

Solução em $O(N)$

- Este é o tipo de problema que fica simplificado se o competidor conhecer os números de Catalan
- Os três exemplos dados correspondem as valores $N = 1, 2, 3$ e as respostas correspondentes são os três primeiros números de Catalan
- É possível resolver manualmente ainda o caso $N = 4$, e a resposta associada, 14, confirma a suspeita da contagem corresponder aos números de Catalan
- Como o N-ésimo número de Catalan pode ser computado diretamenta a partir de N por meio do cálculo de dois fatoriais, a complexidade da solução é $O(N)$ para cada caso de teste

Solução em $O(N)$

```python
import sys
import math


def catalan(n):
    return str(math.factorial(2*n)//((n + 1)*math.factorial(n)**2))


def solve(ns):
    return map(catalan, ns)


if __name__ == '__main__':
    xs = sys.stdin.readlines()
    ns = map(int, xs)
    ans = solve(ns)

    print('\n'.join(ans))
```