Source task and conventions

Implement recursive and dynamic-programming functions for $C_0 = 1$, $C_{n+1} = \sum_{i=0}^{n} C_i C_{n-i}$, and compare them with `timeit`.

Conventional solution and proof

The two required implementations are:

```
def catalan_recursive(n):
    if n < 0: raise ValueError("n must be nonnegative")
    if n == 0: return 1
    return sum(catalan_recursive(i) *
               catalan_recursive(n-1-i) for i in range(n))

def catalan_dp(n):
    if n < 0: raise ValueError("n must be nonnegative")
    c = [1] + [0] * n
    for k in range(1, n+1):
        c[k] = sum(c[i] * c[k-1-i] for i in range(k))
    return c[n]
```

Strong induction proves the recursive function: every recursive argument is smaller, and the returned sum is the defining recurrence. For the dynamic function, before iteration k, all stored entries below k equal the corresponding Catalan numbers; the update establishes entry k, preserving the invariant through termination. The code therefore returns the same exact integer.