Homework 4

Robbie McKinstry, Jack McQuown, Cyrus Ramavarapu

9 September 2016

Greedy Problems

Problem 12:

Problem 18:

Dynamic Programming

Problem 1:

A:

B:

C:

A $O(n)$ algorithm can be derived from the original recurrence relationship by first eliminating the summation by calculating $T(n+1)$ in the following manner.

$$T(n+1) = \sum_{i=1}^{n} T(i)T(i-1)$$

$$T(n) = \sum_{i=1}^{n-1} T(i)T(i-1)$$

$$T(n+1) - T(n) = \sum_{i=1}^{n} T(i)T(i-1) - \sum_{i=1}^{n-1} T(i)T(i-1)$$

$T(n+1)$ and $T(n)$ overlap for all values $i : 1 \leq i \leq n-1$, therefore subtracting the two sums leaves only the the final in the sum for $T(n+1)$.

$$T(n+1) - T(n) = T(n)T(n-1)$$

The values for n can be shifted by setting $n = m - 1$.

$$T(m) - T(m-1) = T(m-1)T(m-2)$$

However, the label m is without meaning, so label $m = n$.

$$T(n) - T(n-1) = T(n-1)T(n-2)$$

Equivalently,

$$T(n) = T(n-1)[1 + T(n-2)]$$

This expression is easily expressed in code.

$Array: \ T$
$T[0] = 2$
$T[1] = 2$
for $i \leftarrow 2$ *to* n **do**
| $T[i] = T[i-1] * (1 + T[i-2])$
end
$Output: \ T[n]$