Description: Program to find nth Fibonacci number using dynamic programming.
Author: Kulthum Lakha ~kyl0029
Programming Language: Python
Run: python fibonacci.py

The dynamic programming method is better than the naive method because it helps avoid re-calculation for the same number.
The naive method includes re-calculating the same result multiple times causing the algorithm to take exponential time.
However, the method I have used takes linear time by storing calculated results in a dynamic data structure (array in this case)
to avoid taking extra time by performing calculations for the same number multiple times.

In the dynamic programming approach, we are solving a subproblem once.
For fib(n) we have total n subproblems — fib(0), fib(1), fib(2) … fib(n) and we are solving each one of them just once.
So, the time complexity for the dynamic approach is O(n) [It is a linear complexity].

The graph below represents results of running my python script to get the nth Fibonacci number that I obtained by using the command "time python fibonacci.py" and recording the user time for various inputs.