Exercise 2

1. Dynamic programming can be used to solve optimization problems where the size of the space of possible solutions is exponentially large.

True

False

1. Dynamic programming can be used to find an approximate solution to an optimization problem, but cannot be used to find a solution that is guaranteed to be optimal.

True

False

1. Recall that sorting a list of integers can take O(nlogn) using an algorithm like merge sort. Dynamic programming can be used to reduce the order of algorithmic complexity of sorting a list of integers to something below nlogn, where n is the length of the list to be sorted.

True

False

1. Problem: I need to go up a flight of N stairs. I can either go up 1 or 2 steps every time. How many different ways are there for me to traverse these steps? For example:

3 steps -> could be 1,1,1 or 1,2 or 2,1

4 steps -> could be 1,1,1,1 or 1,1,2 or 1,2,1 or 2,1,1 or 2,2

5 steps -> could be 1,1,1,1,1 or 1,1,1,2 or 1,1,2,1 or 1,2,1,1 or 2,1,1,1 or 2,2,1 or 1,2,2 or 2,1,2

Does this problem have optimal substructure and overlapping subproblems?

It has optimal substructure and overlapping subproblems

It doe not have optimal substructure and does not have overlapping subproblems

It has optimal substructure and does not have overlapping subproblems

It does not have optimal substructure and it has overlapping subproblems

Explanation to #4:

Optimal substructure: "Optimal solution" in this case is the number of possible ways of traversing k steps -- it’s a somewhat trivial sense of “optimal”, with one good (correct) answer and infinite bad (incorrect) answers. But we can indeed figure out the solution for k+1 or k+2 steps using the solution to k.

Overlapping subproblems: Say there are 8 steps. The solution for 5 steps is required at least three times, as you can do [1,1,1] or [1,2] or [2,1] to reduce the solution to 5 steps.