Hang Bin Zheng
11/14/19
CSC 323
Tue/Thur 1:40-2:55
Prof. Yao

Problem 1: Running Trials and Tribulations

a) Describe the optimal substructure/recurrence that would lead to a recursive solution
The optimal substructure solution for this problem is there will be two base case. 1st case is where the runner is not injured and the 2nd case is the runner is injured and these cases will affect the number of days for runner to finish.

And then we going to look for the maximum speed for the runner that will not get injured.

b) Code your recursive solution under runTrialsRecur(int possible speeds, int days).
SEE CODE

c) Draw recurrence tree for given (#speeds = 6, # days =2)

d) How many distinct subproblems do you end up with given 6 speeds and 2 days?
12

e) How many distinct subproblems for N speeds and M days?
M * N

f) Describe how you would memorize runTrialsRecur.
First, need to define the base case
Then, use the for loop if applies, then do the recursive call.

g) Code a dynamic programming bottom-up solution runTrialsBottomUp(int possibleSpeeds, int days)

see code

Problem 2: Holiday Special

A) Looks for the best possible solution for the cook can do the that fit to the schedule, without moving the cooker schedule around.
B) Compare each number of steps with current steps.
C) See code
D) O(M*N^2)

Problem 3: League of Patience

a) The algorithm I use for this problem is “Dijkstra’s Shortest path algorithm” this algorithm is looking for the shortest path between nodes.
b) O(n^2)
c) Dijkstra’s Shortest path algorithm