Fundamental Algorithms | **Instructor: Amir Shpilka**
Courant, Fall 2017 | **Homework 8** | **Due Date: Wed Nov 8, 11:55pm**

1. You are opening a new chain of hipster artisanal locally-sourced vegan coffe shops on First Avenue. You are given the expected revenue $r_i \geq 0$ of opening a shop on i-th street, for $1 \leq i \leq n$. Your only constraint is that you cannot open two shops on adjacent streets, i.e. on streets j and $j+1$ for some j. The task is to find the maximum expected revenue (forget the actual position of the shops for now) of opening a chain of shops subject to your constraints.
   (a) (2 points) First give an example to show that the best solution does not necessarily have to pick $\lfloor n/2 \rfloor$ streets. That is, it is not enough to consider only the solution of picking all the odd-numbered streets, and all the even-numbered streets.
   (b) (2 points) Give an example to show that this algorithm *does not* always find an optimal solution: pick the street with the maximum revenue r_i, set the value of r_{i-1}, r_i, r_{i+1} to zero and repeat.
   (c) (6 points) Give a dynamic programming algorithm to solve this problem in $O(n)$ time, by giving the recurrence equation along with a proof that it finds the optimal solution.
   (d) (6 points) Now suppose the city doesn't let you buy a shop on every street, and so you are instead given a list of street numbers $x_1, \ldots, x_n$ and the expected revenues $p_1, \ldots, p_n$ where $p_i \geq 0$ is the revenue of opening a shop on x_i-th street. You also have a number K which is the minimum number of streets you need between two of your shops (so the previous parts were using $K = 2$). Modify your recurrence equation from the previous subproblem to solve this problem.

2. You graduated from NYU and set up a fancy new start-up that is going to change the world. Your start-up grows rapidly, and soon you have two offices, one in the Upper East Side, and one in the Upper West Side. Every day, you need to decide which office to work from. You have with you the list of upcoming clients to each office, along with the extra money you'd make if you personally worked with them. That is, assume you are given $E_1, \ldots, E_n$ and $W_1, \ldots, W_n$ where E_i is the money you'd make by working in the UES office on day i and W_i is the money you'd make by working in the UWS office.

   Since it's a start-up you will also sleep in the office that you worked from, and so if you want to work from the other office the next day, you need to walk through Central Park in the morning. This takes ages, and so you expect to lose a fixed sum of C every time you switch offices. Your task is to calculate the maximum money you can make by scheduling your work optimally for the next n days.
   (a) (2 points) Give an example (i.e. values for E_i, W_i and C) where the following algorithm does not return the maximum value: each day, pick the office with the higher expected money made.
   (b) (6 points) Give a polynomial time algorithm to find the maximum money you can make. Briefly justify why it is optimal.

3. (Not all Greedy are Equal)
   (a) (3 points) Give an example to show that this strategy is not optimal for the activity selection problem: always picking the shortest (least duration) activity from the remaining compatible activities until none remain.
   (b) (3 points) Give an example to show that this strategy is not optimal for the activity selection problem: always picking the compatible remaining activity with the earliest start time.

4. You decide to do the Apalachian trail, which is long hike, with places to stop and rest at $x_1, \ldots, x_n$ miles from the beginning of the trail (x_n is also the end of the trail). You are super fit, but still can only hike a maximum of M miles a day (don't worry, the stops are never more than M miles apart). You're in a rush, so you want to find the minimum number of stops you need to make.

(a) (4 points) Give a dynamic-programming-style recurrence equation for the minimum number of stops you need to make. Briefly justify the correctness of this equation.

(b) (5 points) Give a greedy algorithm to find the minimum number of stops, and prove that it is correct.

(c) (1 point) What are the running times of both your above algorithms?

5. (10 points) Little Johnny is extremely fond of watching television. His parents are off for work for the period $[S, F)$, and he wants to make full use of this time by watching as much television as possible: in fact, he wants to watch TV non-stop the entire period $[S; F)$. He has a list of his favorite n TV shows (on different channels), where the i-th show runs for the time period $[s_i, f_i)$, so that the union of $[s_i, f_i)$ fully covers the entire time period $[S, F)$ when his parents are away.

   Little Johnny doesn't mind switching in the middle of a show he is watching, but is very lazy to switch TV channels, so he wants to find the smallest set of TV shows that he can watch, and still stay occupied for the entire period $[S, F)$. Design an efficient $O(n \log n)$ greedy algorithm to help Little Johny.