CSE421: Design and Analysis of Algorithms	May 16, 2019
Homework 6	
Shayan Oveis Gharan	Due: May 23, 2019 at 5:00 PM

P1) Drogo, Daenerys's Dragon started practicing free climbing. The wall he is practicing on is $n+1$ squares tall and $2k+1$ squares wide. The bottom left square is at $(-k, 0)$ and the top right is at (k, n)

Little Drogo starts from the block $(0, 0)$. At each step Little Drogo climbs one block and either moves left one block, right one block or stays in the same line of climbing. So from (a, b), Little Drogo can go to $(a-1, b+1)$, $(a, b+1)$ and $(a+1, b+1)$, as long as the destination square exists.

Design a polynomial time algorithm that outputs in how many ways can little Drogo climb to the top level? For example, if $n = 2, k = 1$ the answer is 7.

P2) Suppose we have a path with $n+1$ vertices numbered $0, \dots, n$. We want to take a package from vertex 0 to vertex n. There are m mailmen on this line, where the i-th mailman is located at $p[i]$, i.e., array p has the location of all mailmen. For each mailman, i, let $v[i]$ be the speed of i, e.g., if $v[i] = 3$ it means that mailman i goes from vertex a to $a+1$ or $a-1$ in 1/3 of a second. To move the package, a mailman should pick it up at point 0 and move to a vertex a_1, at that point another mailman can move the package to a vertex a_2 and so on until the package reaches point n. The goal is to minimize the time that it takes to take the package to vertex n. You can assume all entries of p, v are integers. We want to design a polynomial time algorithm that outputs the minimum number of seconds needed to do this job.

Here is an example: Suppose $n = 10, m = 2, p[1] = 2, p[2] = 6, v[1] = 1, v[2] = 2$. One strategy is that the second mailman goes to 0 (in 3 seconds) picks up the package and goes to n (in 5 seconds). This would take 8 seconds. But, the optimum strategy is that mailman 1 goes to 0 (in 2 seconds) picks up the package and goes to 1 (in 1 second) meanwhile mailman 2 goes to 1 (because it takes him only 2.5 seconds to go to 1. He grabs the package and takes it to n in 4.5 seconds. So by this strategy the package will reach n in $2 + 1 + 4.5 = 7.5$ seconds.

a) Prove that if in the optimum solution, the package is handed from mailman i to mailman j at some point, we must have $v[j] \geq v[i]$. Use this fact in designing your algorithm.

b) Design a polynomial time algorithm that outputs the minimum number of seconds needed to do the job.

Hint: Sort the mailmen based on their speed and assume $v[1] \leq v[2] \leq \dots v[m]$. For all i, j let $p(i, j)$ be the minimum number of seconds needed to pick up the package from 0 and move it to vertex j using only mailmen $1, \dots, i$.

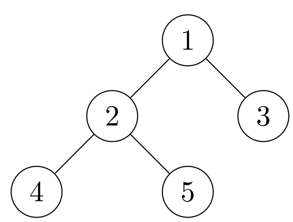
P3) You are given an $n \times n$ array A where for all $1 \leq i, j \leq n$, $A[i, j]$ is an integer that may be negative. For a rectangle $(x_1, y_1), (x_2, y_2)$ where $x_1 \leq x_2$ and $y_1 \leq y_2$, the value is the sum of all numbers in this rectangle, i.e.,

$$\sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} A[i, j]$$

Design an algorithm that runs in time $O(n^3)$ and outputs the value of the rectangle of largest value. Note that the value of the empty rectangle is zero. For example, if A is the following array, the optimum rectangle has value 8.

0	4	3
3	1	-5
-2	1	3

P4) You are given a tree T where every node i has weight $w_i \geq 0$. Design a polynomial time algorithm to find the weight of the largest weight independent set in T. For example, suppose in the following picture $w_1 = 3, w_2 = 1, w_3 = 4, w_4 = 3, w_5 = 6$. the maximum independent set has nodes $3, 4, 5$ with weight $4 + 3 + 6 = 13$.



P5) **Extra Credit:** Given a sequence of positive numbers $x_1, \dots, x_n$ and an integer k, design a polynomial time algorithm that outputs

$$\sum_{S \in \binom{n}{k}} \prod_{i \in S} x_i,$$

where the sum is over all subsets of size k.