Homework 2

Problems:

1. Job scheduling: given n jobs where job i requires time t_i and has a weight w_i characterizing the importance of the job, we want to schedule the jobs on a single machine without overlap that minimizes
$$\sum_i w_i C_i,$$
where C_i is the finishing time of job i. Suppose we start at time 0.

2. Consider the following problem. The input consists of n skiers with heights $p_1, \cdots, p_n$, and n skies with heights $s_1, \cdots, s_n$. The problem is to assign each skier a ski to to minimize the average difference between the height of a skier and his/her assigned ski. That is, if the ith skier is given the $\alpha(i)$th ski, then you want to minimize
$$\sum_{i=1}^{n} |p_i - s_{\alpha(i)}|.$$
   (a) Consider the following greedy algorithm. Find the skier and ski whose height difference is minimized. Assign this skier this ski. Repeat the process until every skier has a ski. Prove of disprove that this algorithm is correct.
   (b) Consider the following greedy algorithm. Give the shortest skier the shortest ski, give the second shortest skier the second shortest ski, give the third shortest skier the third shortest ski, etc. Prove of disprove that this algorithm is correct.

3. The input to this problem consists of an ordered list of n words. The length of the ith word is w_i, that is the ith word takes up w_i spaces. (For simplicity assume that there are no spaces between words.) The goal is to break this ordered list of words into lines, this is called a layout. Note that you can not reorder the words. The length of a line is the sum of the lengths of the words on that line. The ideal line length is L. No line may be longer than L, although it may be shorter. The penalty for having a line of length K is $L - K$. There are two ways to define the total penalty as shown below. The problem is to find a layout that minimizes the total penalty.
   (a) In the first definition, the total penalty is the sum of the line penalties.
   (b) In the second definition, the total penalty is the maximum of the line penalties.

   Prove or disprove that the following algorithm gives the correct solution for each of the problems above.

```
For i = 1 to n
      Place the ith word on the current line if it fits
            else place the ith word on a new line
```

4. Given a graph G, each edge (i, j) has a weight $r_{ij} > 0$, the goal is to find whether there is a cycle such that the multiplication of the weights on the edges of the cycle is greater than 1. Find a polynomial time algorithm.

5. A shuffle of two strings X and Y is formed by interspersing the characters into a new string, keeping the characters of X and Y in the same order. For example, 'bananaananas' is a shuffle of 'banana' and 'ananas' in several different ways depending on the way the two strings are overleaved. Given three strings $A[1..m]$, $B[1..n]$ and $C[1..m+n]$, describe and analyze an algorithm to determine whether C is a shuffle of A and B.

6. Assume that you have a subroutine IsWord that takes an array of characters as input and returns True if and only if the string is a valid English word. Design efficient algorithms for the following problems and bound the number of calls to IsWord.

   (a) Given an array $A[1 \cdots n]$ of characters, compute the number of partitions of A into words. For example, given the string ARTISTOIL, your algorithm should return 2, for the partitions ARTIST·OIL and ART·IS·TOIL.

   (b) Given two arrays $A[1 \cdots n]$ and $B[1 \cdots n]$ of characters, decide whether A and B can be partitioned into words at the same indices.

   (c) Given two arrays $A[1 \cdots n]$ and $B[1 \cdots n]$ of characters, compute the number of different ways that A and B can be partitioned into words at the same indices.

7. **Distances between polygonal curves** A polygonal curve of n vertices $x_1, x_2, \cdots, x_n$ is the concatenation of $n-1$ line segments $x_i x_{i+1}$, for $i = 1, 2, \cdots, n-1$. Given two polygonal curves X, Y, each with n vertices $\{x_i\}$, $\{y_i\}$ respectively, a natural question is to define the distance between them.

   The Frechét distance is defined in the following way. Imagine that one person walks on the curve X and a dog walks on the curve Y. The dog is tied with a leash (or rope). Both the person and the dog can not walk backward. In addition, to simplify the problem, the person and the dog can only walk one at a time. When the person (or the dog) walks, the dog (or the person) needs to wait at a vertex. The Frechét distance is the minimum length of the leash for the person and the dog to possibly finish the walk.

   Give an $O(n^2)$ algorithm to compute the discrete Frechét distance of the two curves.