CPSC 413: Assignment 2

The assignment is due on Dec. 9th at 4pm. You make work with others in the class, but you must write your solution up individually.

1. A biologist is looking at n samples of DNA. Finding the exact DNA code for each sample is quite a slow process. However, all the biologist wants to know is whether at least half the samples come from the same animal. He has a method to quickly test whether any pair of DNA samples comes from the same animal. Design a divide-and-conquer algorithm that takes as input a set of n DNA samples, and determines if at least half of the group come from the same animal by only testing $O(n \log n)$ pairs.

   For each of the next three problems, describe a polynomial-time dynamic programming solution to the problem, prove that it always returns the correct answer, and determine its running time.

2. Each week, you have the choice between heating your house with solar power, or with gas. For each the of the next n weeks, you can estimate how much each heating type will cost you: say it will cost s_i dollars if you heat with solar power in month i, and g_i dollars if you heat with gas power. Unfortunately, you can't simply switch back and forth between the two heating types easily; doing so costs p dollars each week you switch from solar to gas or vice versa. Give an algorithm that determines which weeks to switch from solar to gas and back so as to minimize your total costs (it should also determine which type of heating to start with). For example, if we had

   $$s_1 = \$10, s_2 = \$5, s_3 = \$7,$$

   and

   $$g_1 = \$5, g_2 = \$12, g_3 = \$4,$$

   and $p = \$4$, then the optimal solution would be to start with gas, then switch to solar for weeks two and three.

3. At your company, you've just found out that a number of documents have become corrupted - somehow, the documents have lost all spacing between words. What you need to do is design a program that can turn such a document into the best possible readable file, by restoring the

spaces in their most logical places. For example, the program should be able to take the sequence "Ihadanicedaytoday" and return "I had a nice day today", rather than "Ihad anice dayt oday". To help with this task, you have a function **quality** that takes in a string of characters, and returns a real number (that could be positive or negative), representing how close that string is to a real word. For example, it might return a negative value for "Ihad" and a positive one for "had". Supposing that quality runs in polynomial time, give an algorithm that takes a string and returns where to insert spaces so as to maximize the total quality of the strings seperated by the spaces.

4. In the United States, a process called "gerrymandering" is used to change electoral disctricts in a candidates favour. Suppose we know, in each of n precincts, how many people voted for party A (a_i), and how many voted for party B (b_i). The candidate in power, who runs for party A, wants to seperate these precincts into two different districts, so that his party has the majority in both of the districts. For example, suppose that we have four precincts, with

$$a_1 = 55, a_2 = 43, a_3 = 60, a_4 = 47$$

and

$$b_1 = 45, b_2 = 57, b_3 = 40, b_4 = 40.$$

If he grouped precincts 1 and 4 together, and 2 and 3 together, then he would have the majority in both of these new districts (102 to 85 in precincts 1 and 4, and 103 to 97 in precincts 2 and 3). Write an algorithm that, given the voting numbers for the n precincts, determines if it is possible to split the precincts into two districts, with A having the majority in both of the districts.

5. An ecologist has been observing the locations of where a number of animals have drunk water. She keeps this data in an array Water$[i, j]$, in which the (i, j)th entry of the array contains how many times the ith animal has drunk water at the jth location (suppose there are n animals and m locations). She would like to determine, given an integer $k \leq n$, if there is a set of k animals for which no two have drunk water from the same location. Show that this problem is NP-complete.