Word Cloud
Given T data sets, word clouds are generated based on the frequency that each listed word appears on a given website, with one cloud pertaining to one data set. A word is output larger or smaller based on its frequency.
The point (pt.) size of each word within the cloud is determined by the formula P = 8 + [40(c*w*-4)/c*max*-4)], where c*w* is the number of occurrences of the current word and c*max* is the number of occurrences of the data set's most frequent word. Words are placed into rows of maximum width W, in which the height of a given row is the maximum font size of any word rendered in that row.
Output the total height for each cloud, using the height of each of its rows.

Turbo
An algorithm is given an array of integers between 1 and N (inclusive), with each of those appearing exactly once. It runs in N phases, repeatedly swapping consecutive elements as its sorting method. In odd-numbered phases, the algorithm chooses the smallest number not yet chosen and moves it to its final position. In even-numbered phases, it chooses the largest number not yet chosen and moves it to its final position. For instance, in the first phase, number 1 is moved to position 1. In the second, the number N is moved to position N.
Given the initial array and the value of N, output the number of swaps used in each phase of this algorithm.

Bilateral
A company consists of many two-person teams, where each team is assigned to one project. One employee can be a member of several teams, but there cannot be several teams consisting of the same pair of employees.
Given a list of two-person teams, compute the smallest number of people that must be included to get at least one person from each project.
Output the number of people on this list as well as the ID of each person on it. If possible (while keeping the list as small as possible), this list should include employee ID 1009.

Towers
Given a data set of M integers, sort them (arrange from smallest to largest) and output the resulting list. If two numbers are equal, the one that was first in the input should also be first in the output.

Autocompletion
An algorithm is needed to provide options to auto-complete text input into a search bar based on a predefined dictionary of N words. If the search bar contains the text W, then

the auto-completion will cycle forward through the dictionary starting at the lexicographically smallest word that has W as its non-trivial prefix. For instance, if the input W is “app” and the dictionary contains “albert”, “app”, “apple”, and “appletree”, then the algorithm will start at “app”.
Each time that the user presses the tab key, the algorithm finds the next smallest word in the dictionary. For instance, if the earlier user presses tab twice, then they will get “appletree”, the second lexicographically smallest word with “app” as prefix. If the algorithm reaches the end of the list, it will circulate back to the smallest word with the W prefix. If there are no words that have W as a prefix, pressing tab has no effect.
When the user types a letter, the letter is directly appended to the text.
Given the predefined dictionary and a list of keystroke sequences, output the text that the algorithm would return for each keystroke sequence. Assume that each sequence of consecutive tab presses only triggers the auto-completion one time.