CPSC 335 – Project 01 Report

Algorithm 01: The Alternating Disk Problem

Members:

Name	Email
Andrew Lee	aslee@csu.fullerton.edu
Michael Bui	buimichael@csu.fullerton.edu
Natalia Garcia	natgarcia@csu.fullerton.edu

Algorithm Design & Implementation:

For the alternating disk problem, we decided to design an algorithm that implements a doubly linked list, which checks two concurrent disks at a time with the recursiveSort() function. In this case, disks will also be referred to as nodes that will use the function to check for four conditions. Other helper functions like swap() and deleteNode() were also implemented to the program.

To start, the first condition checks if there are no more nodes to the right, which then terminates the program. The second condition checks if a dark disk has a light disk to the left, in which case, will be swapped. The third condition checks for a light disk having a dark disk to the right and if there is, the disks are swapped. Lastly, the fourth condition is there for when the disks don't need to be swapped, so nothing is done. Except for the first condition, the function recursiveSort() is called recursively for the other conditions when the disks are swapped and when there isn't a need to have the disks swapped.

Algorithm 02: Matching Group Schedules

Members:

Name	Email
Andrew Lee	aslee@csu.fullerton.edu
Michael Bui	buimichael@csu.fullerton.edu
Natalia Garcia	natgarcia@csu.fullerton.edu

Algorithm Design & Implementation:

For matching group schedules, we were inspired by Alpha-Beta Pruning to design an algorithm that uses two variables to check the min and max values of availability windows. The input was standardized to an integer value and each intersecting interval of availability in the schedules was checked with the min and max variables. Valid results which were not shorter than the meeting time given were added to the output after converting the integers back into military time.

To start, the input is converted into integers as mentioned previously in order to work with the data in an accessible manner. The daily acts given by the users are used to create a new list of availability times. These new lists are compared, and whenever an interval intersected another interval, the min and max variables are used to determine eligibility and duration of availability. This information is used to filter out unwanted output, and only append the available times for all people which had enough time for the meeting. This output list is then converted back into military time.