Experiments Scheduling

A) The optimal substructure of this problem is to find the minimum number of switches between students and complete all N steps of the experiment. So, after all students assign the steps they can do, we have one student do some steps and within the remaining steps we have other students do it. We keep repeating the process until there are no more steps left. Then we will obtain the optimal solution of minimum switches to complete all steps between students.
B) The greedy algorithm that could find an optimal way to schedule the students is to find the student that can do the most consecutive steps at a time. So, the algorithm would be looping through all students to see who can do the steps in a row the longest, and then update the steps and look for the next students with steps in a row until all steps are done. If both students have the same number of consecutive steps, then the first student would do it.
C) Work with Ting Ting Ye.
D) O(m*n)
E) Assume there exists an opt sol. OPT using fewer switches
   ALG: $SW_1, SW_2, \ldots, SW_k$
   OPT: $SW_1', SW_2', \ldots, SW_l'$ $l < k$
   Let's say the ith SW_i or SW_i' is where ALG and OPT first diverge.
   By design our algorithm only chooses the student that can do the most consecutive steps. If we replace SW_i' with SW_i it won't worsen the OPT. Modify OPT and ALG, all the SWs same up to SW_{i+1}' and SW_{i+1}. We can use the same cut and paste logic up to SW_l and SW_l'. OPT solution claims SW_l' is the last switch. Our algorithm stops only when all steps have been schedule. Thus, there must be some student missing in the OPT solution. As a result, we reached contradiction that no valid OPT uses fewer switches. Therefore, our algorithm yields an optimal solution.

Public, Public Transit

A) An algorithm to this problem is incorporating Dijkstra's Algorithm with calculating amount of waiting time and then return the time difference between when arriving at destination and the time it starts. There are two cases where the first train haven't arrived or the other one is train arrives at least once.
B) The time complexity for such algorithm is $O(V^2$+waiting time). Waiting time would be insignificant. So, it would be $O(V^2)$.
C) The algorithm is implementing Dijsktra's algorithm.
D) I would be using the shortestTime in myShortestTravelTime. Also, adding the calculation for waiting time for whether the first train came or not.
E) The current complexity of "shortestTime" is $O(V^2)$. Through Professor's notes with Fibonacci Heap the time complexity can become O(VlgV + E).