COMPSCI330 Design and Analysis of Algorithms
Assignment 3

Due Date: Monday, February 11, 2019 at **5 pm**

Guidelines

- **Describing Algorithms** If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice. If the running time of your algorithm is worse than the suggested running time, you might receive partial credits.
- **Typesetting and Submission** Please submit the problems to GradeScope. You will be asked to **label your solution for individual problems**. Failing to label your solution can cost you 5% of the total points (3 points out of 60 for this homework).
- LaTeX is preferred, but answers typed with other software and converted to pdf is also accepted. Please make sure you submit to the correct problem, and your file can be opened by standard pdf reader. **Handwritten answers or pdf files that cannot be opened will not be graded.**
- **Timing** Please start early. The problems are difficult and they can take hours to solve. The time you spend on finding the proof can be much longer than the time to write. If you need more time for your homework please use this form and submit a STINF.
- **Collaboration Policy** Please check this page for the collaboration policy. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Clarifications on Grading Policy:

1. If your algorithm has a major problem (wrong greedy choices, have counter-examples), then the corresponding correctness proof will receive no credits.

2. If your algorithm is not the fastest possible (for P3), then you still receive partial credit as long as it is still a polynomial time algorithm.

3. If your algorithm is complicated and the proof is not correct, then you may not receive any credit on the algorithm question.

Problem 1 (Currency System)**.** As we all know, US dollar is the official currency in United States. There are different bills of value $100, 50, 20, 10, 5, 2, 1$, and coins of value $1, 0.50, 0.25, 0.1, 0.05, 0.01$ (although you rarely see \$2 bills and \$1,\$0.50 coins, they exist, see `https://en.wikipedia.org/wiki/United_States_dollar`). In this problem we consider the problem of buying something with the smallest number of bills/coins.

A simple way to pay x dollars is to use the following Greedy-Pay algorithm:

```
Greedy-Pay(x)
   While x > 0
      Let u be the largest value bill/coin such that u <= x
      Pay the bill/coin with value u
      Let x = x - u
```

(a) (12 points) Prove that if you have unlimited number of bills of $\{1, 2, 5, 10\}$ dollars (and you don't have any other bills/coins), no matter what x is (as long as it is a multiple of 1) the Greedy-Pay algorithm minimizes the number of bills/coins used for the payment. (Change is not allowed here. For example, you cannot pay \$9 by a \$10 bill and get \$1 back.)
(b) (8 points) Show an example for which the Greedy-Pay algorithm does not minimize the number of bills/coins used, when you have a limited number of bills/coins. (For example, you might not have a \$2 bill or might only have 2 \$20 bills.) Assume you have unlimited number of 1 cent coins so you have enough money to pay.

(Note: Your example should consist of an x - the amount you need to pay, and a set of bills/coins you have. Your example can use all bill/coin values mentioned in the problem (not limited to {1,2,5,10} as in (a)). You should be able to pay the amount x using fewer bills/coins than what Greedy-Pay uses, describe what bills you will use and what bills Greedy-Pay will use.)

Problem 2 (Physics Experiment)**.** (16 points) Professor X from Duke Physics Department is conducting a large experiment. This experiment has n crucial steps, and each step require one of Professor X's m Ph.D. students. However, these steps are very complicated. Each student only knows how to perform a subset of the steps. Professor X knows when the experiment switches hands from one student to another, there is a risk for failure due to limited communication. So he wants to finish the experiment with as few switches as possible. Since you are taking 330, Professor X hopes you can design an algorithm to help him.

Your input will be n, the number of steps, and m, the number of students. This is followed by m lists, where the i-th list contains all the steps that student i can perform.

You need to output a schedule: in particular, for each step you need to assign a student $a[i]$ to perform that step. Your schedule should minimize the number of switches (the number of steps where $a[i] \neq a[i-1]$).

As an example, suppose $n = 5$, $m = 3$, student 1 can do steps 1, 2, 5, student 2 can do steps 1, 3, 5 and student 3 can do steps 2, 3, 4. Then one valid solution is $a = [1, 3, 3, 3, 1]$. That is, student 1 will do step 1, student 3 will do steps 2, 3, 4, and then student 1 will finish step 5. The number of switch is 2. Notice that it is OK for student 1 to work on both 1 and 5.

(a) (6 points) Design the algorithm and analyze its running time.

(b) (10 points) Prove the correctness of the algorithm.

Your algorithm should run in time polynomial in n, m.

Problem 3 (24 points, Problem 5 in KT)**.** Let's consider a long, quiet country road with houses scattered very sparsely along it. (We can picture the road as a long line segment, with an eastern endpoint and a western endpoint.) More concretely, assume the road has length m, with n houses at coordinates $x_1, x_2, ..., x_n \in [0, m]$. Further let's suppose that despite the bucolic setting, the residents of all these houses are avid cellphone users. You want to place cellphone base stations at certain points along the road, so that every house is within 4 miles of one of the base stations.

(a) (6 points) Give an efficient algorithm that makes sure every house is within 4 miles of one of the base stations, using as few base stations as possible. Analyze its running time, and make your algorithm as fast as possible. (For this problem your input is $n, m, x_1, x_2, ..., x_n$. The x_i's are not necessarily sorted. All the inputs are positive integers.)

(b) (10 points) Prove the correctness of algorithm designed in part (a)

(c) (8 points) Now suppose the cellphone company can only construct t base stations, but they are willing to change the range of the base stations (so now instead of 4 miles, every house only needs to be within k miles). Design an algorithm that finds the smallest integer k that allow t base stations to cover all the houses. Analyze its running time, and make your algorithm as fast as possible. (For this problem your input is $n, m, t, x_1, x_2, ..., x_n$. The x_i's are not necessarily sorted. All the inputs are positive integers.)

(You don't need to prove correctness for this sub-problem.)

(Hint: You can use (a) as a subroutine.)