Programming Assignment #2

Programming assignments are to be done individually. You may discuss the problem and general concepts with other students, but there should be no sharing of code. You may not submit code other than that which you write yourself or is provided with the assignment. This restriction specifically prohibits downloading code from the Internet. If any code you submit is in violation of this policy, you will receive no credit for the entire assignment.

The goals of this lab are:

- To recognize algorithms you have learned in class in new contexts.
- Ensure that you understand graphs and greedy algorithms.

Problem Description

We have provided Java code skeletons that you will fill in with your own solution. Please read through this document and the documentation in the starter code thoroughly before beginning. For this program, you have three java files given to you, Driver.java, Robber.java and Graph.java. Do NOT add a constructor to Robber.java and do NOT change the signature of the constructor for Graph.java. The driver is provided to help you test your program. You can make changes to Driver.java, but we will not use them for grading your program. We will use our own driver for grading your program. Hence, the correctness of your algorithms should not be dependent on the code in Driver.java.

Fruitcake Frank is a professional burglar. These days he has been hanging around the neighborhood of Algs and has made some observations about the residents. He thinks he can use these observations to plan his burglaries to maximize his loot.

- A few residents in the neighborhood are very forgetful and forget to lock their doors when they leave the house.
- Many of the residents are worried about getting locked out of their house, so they have given their trusted friends in the neighborhood keys to their house.
- **Some houses have multiple keys, for example a key to the storm door lock and a key to the front door lock, which have been divided amongst different friends. Fruitcake must have all the keys for a house in order to rob the house. Assume that if a house has multiple keys and at least 1 key has been given to a neighbor, then all of its keys have been distributed amongst the neighbors.**
- No one in the neighborhood has a security system so Fruitcake only needs the keys to a locked house to be able to rob it. (Note: Fruitcake is weak so he cannot break open the door or window to a locked house.)

There are several parts to the problem.

Part 1: Represent the Neighborhood as a Graph [20 points]
Fruitcake has made a list of houses with all doors left unlocked, given in 'unlocked.txt'. He also has a list of which people have given keys to which of their neighbors given in 'keys.txt'. An example is given below.

`unlocked.txt`

- `House A`

`keys.txt`

- `House A: House B, House C`
- `House B:`
- `House C: House D`
- `House D: House B`

The above lists indicate that House A is unlocked and House A has the keys to House B and C, House B has no keys, House C has the key to House D, and House D has the key to House B. From the assumptions given at the beginning of the assignment, we also know that House B has exactly two keys and House D and C have exactly one key.

Fruitcake thinks his current representation of this situation is not as effective as a graph representation would be. A pictoral graph of the example above might look something like this:

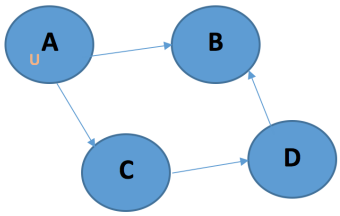

Figure 1: An example

Notice how the represenation distinguishes between locked and unlocked houses.

Help Fruitcake represent the houses and keys as a graph. In particular, fill in the template Graph.java. The constructor for Graph.java is `Graph(String unlocked, String keys)`, where unlocked is a String representing the path to unlocked.txt and keys is a String representing the path to keys.txt. You should implement the constructor and `boolean containsVertex(String node)`, `boolean containsEdge(String start, String end)`, and `boolean isLocked(String house)` where node, start, end, and house are all names of houses. For example, from the lists given above, `g.containsVertex("House E")` should return false, `boolean g.containsEdge("House C", "House`

`D")` should return true, and `g.isLocked("House A")` should return false. You may represent the graph however you choose as long as `containsVertex`, `containsEdge`, and `isLocked` function properly. Do not change the method signature for the constructor, `containsVertex`, `containsEdge`, and `isLocked`. Do not add an overloaded constructor as we will only create a graph using the given constructor.

In your report, describe breifly how you have represented the graph (adjancey list, adjancey matrix, directed, undirected, etc) and justify the choices you have made. You should consider the remaining parts of this assignment before making decisions on your graph representation.

Part 2: Determine Order to Burglarize the Houses [30 points]
Now that Fruitcake has a graph representation, he wants to use the information to attempt to rob all the houses in the neighborhood. Recall, in order to rob a locked house, he needs to be able to get inside the house. He can only get inside a house if 1) the house is unlocked or 2) the house is initially locked but he gets all keys to the locked house. Note that if House X is unlocked, he doesn't need to worry obtaining the keys to House X. Remember, for houses that have multiple keys, if at least 1 key has been given to a neighbor, then you may assume that all of its keys have been distributed amongst the neighbors.

Given the graph that you built in Part 1, write an algorithm which prints out the order in which Fruitcake should rob the houses if he can rob all the houses in the neighborhood. In particular, implement the method, `public boolean canRobAllHouses(Graph neighborhood)` in Robber.java. The input of this method is the Graph you created in part 1. The method should return false if it is not possible to rob all the houses or true if it is possible to rob all the houses and print to the console the order in which the houses should be robbed with each house seperated by a comma. You do not need to print anything if all the houses cannot be robbed. Consider again the example in Figure 1. For that example, `canRobAllHouses` should return true and print to the console:

```
House A, House C, House D, House B
```

Note that it is very important that you format your output exactly as described above otherwise our grader will mark you answer as incorrect. Do not change the method signature for `canRobAllHouses`.
In your report describe briefly your algorithm, its runtime, and its space complexity. Justify your answers.

Part 3: Maximize Value of Loot [20 points]
Fruitcake is on his way robbing houses and hits the jackpot in one of the houses. Turns out one of the residents of Algs is a professional gourmet chef and just recieved a shipment of exotic and expensive ingredients such as saffron, truffles, and edible gold. Unfortunately, Fruitcake can only carry a limited amount of weight with him. Fruitcake makes a list of each ingredient and using a kitchen scale notes down the amount of each ingredient avaliable in pounds. He also uses his professional burglar knowledge to make an estimate on the black market price of each ingredient per pound. He also notes how much weight he can carry. This list is in ingredients.txt. This list is formatted as Item Name, Amount of Each Ingredient Available in Pounds, Price of Each Ingredient

Per Pound. The first entry in the text file is the maximum weight he carry. An example is given below.
`ingredients.txt`

- `1.4`
- `Saffron, 0.5, 5000`
- `Black Truffles, 3, 95`
- `Winter White Truffles, 1.6, 10000`

Fruitcake thinks he can use this information to maximize his loot. Given the maximum weight that Fruitcake can carry, tell him how much of each ingredient he should take in pounds. In particular implement the method, `public void maximizeLoot(String lootList)` in Robber.java. Note that he does not have to take all of an ingredient. For the above example, the maximum weight he can hold is 1.4 lbs, so he could take 1.2 lbs of Winter White Truffles and 0.2 lbs of Saffron. The input of `maximizeLoot` is a String representing the path to ingredients.txt. The method should print to the console the ingredient and the amount of each ingredient Fruitcake should steal. For the above example, a correctly formatted output should look as follows:

```
Winter White Truffles 1.2
Saffron 0.2
```

Note that the example above is only shown for formatting purposes and not the correct output of your algorithm.

In your report describe your algorithm and its runtime. Will your algorithm work if Fruitcake makes a list of the electronic devices (TVs, laptops, phones, etc) in the house, assuming he cannot steal a fraction of any of the devices? Why or why not?

Part 4: Plan Buyer Meetings [30 points]
Fruitcake has finally completed his burglaries and has lined up some potential buyers for his stolen items. The buyers want to come inspect the goods. Fruitcake wants to get out of town as fast as possible, so he has decided to meet as many buyers as he can in one day. However, as is often true in the criminal world, the buyers do not get along. So Fruitcake cannot schedule overlapping meetings for the buyers. In addition, he needs to keep a window of at least fifteen minutes between each of the meetings so that the buyers do not accidentally bump into each other on their way in and out. Given a list of potential buyers and the time the buyer has requested for the meeting, listed in buyers.txt, tell Fruitcake which meeting requests he should accept so that he maximizes the number of buyers he meets. In particular, implement `public void scheduleMeetings(String buyerList)` in Robber.java. The input of `scheduleMeetings` is a String representing the path to buyers.txt. Here is an example of buyers.txt:
`buyers.txt`

- `Cupcake Carl, 10am-11am`

- `Cookie Cory, 1pm-7pm`
- `Danish Danny, 11am-2pm`
- `Donut Dawn, 3pm-4pm`
- `Macaroon Mary, 3:30pm-6pm`
- `Pie Polly, 11:15am-3pm`

This method should print to console the buyers with whom Fruitcake should accept a meeting with each buyer seperated by a line. The order of the buyers listed does not matter. For the above example a correctly formatted output should look as follows:

```
Cupcake Carl
Pie Polly
Macaroon Mary
```

Note that it is very important you correctly format the output as specified above otherwise our grader will mark you answer as incorrect.
In your report describe your algorithm and its runtime. Prove that your algorithm is correct.

What To Submit

You should submit a single zip file titled `eid_lastname_firstname.zip` that contains all of your java files (Graph.java, Robber.java, and any other files you added) and pdf report `eid_lastname_firstname.pdf`. Do not put these files in a folder before you zip them (i.e. the files should be in the root of the ZIP archive). Your solution must be submitted via Canvas BEFORE 11:59 pm on March 28, 2017. Absolutely no late assignments will be accepted for any reason.

Other Tips and Instructions

- Make sure your program compiles on the LRC machines before you submit it.
- We will be checking programming style. A penalty of up to 10 points will be given for poor programming practices (e.g. do not name your variables foo1, foo2, int1, and int2).
- It is **highly recommended** that you do comment your code and follow good programming style. It will be very unlikely that you recieve partial credit if there are not descriptive comments and the TAs cannot understand what your code is/should be doing.