CS 2110, Summer 2016 Homework 5

You may work in groups of 1, 2, or 3, and may choose to work on either one of the following questions. This hw can be submitted **only** *via group demo to Prof Bailey (and subsequent uploading to cms).*

Question 1:

In the country of GnomenWald there lie many isolated villages connected by roads (roads only ever go from village to village, never going nowhere nor branching out partway). The gnomes are very organised, so name their villages by integers, starting incrementally from "1". In the days of the Grand Archgnome Zijphraagh, a complete road map was made of GnomenWald – some of the gnomes were surprised to see that some villages had many roads connecting to them, whereas some had only two, or even only one. They also noticed that some of the roads were only one-way (often due to the rigours of the terrain, but sometimes due to political sensitivities), so the map-makers were very careful to label all their roads as one-way (so that a two-way road between two villages would be drawn as a pair of one-way roads in each direction). Bear in mind that some roads will be cheap and others more expensive, and some roads will be slow and others faster (not necessarily related to the toll cost of using them). After several generations the population had increased so much that new villages started to spring up (though always connected via a road or roads to one or more villages in GnomenWald), and sometimes villages would disappear (though that made it very hard for the mapmakers since they insisted that roads only joined villages and never branched in the middle).

Build a data structure to represent this road map by having nodes hold villages and expanding on the idea of 'next' in order to connect villages via roads. Test your data structure by using it in a program which tries to move some gnomes around the villages 'along' the roads (at this stage you don't need to 'show' them going along a road -- simply show the succession of villages a given gnome visits). You should also test your ability to add new villages, you should also try to handle the case of deleting villages (though that's an intellectually amusing problem to think through). Do bear in mind that gnomes have been gknown to become disoriented, so there are times when it becomes very important to be able to find a gnome very quickly, so your data structure should also provide the ability to keep tabs on all the gnomes and allow very rapid location of any given gnome. However, you should also know that gnomes are notoriously secretive, so abhor any hints that any of their information can be accessed without proper authorization – as just two examples: gnomes only work in a cash economy (paying tolls, paying for accommodation), but don't want anyone knowing how much cash they have; also, locating a gnome has to be authorised by at least two or three different gnomes (we'll talk about tricks for this in class on Monday).

Now write a GUI for this with the following properties. It should display the graph of villages and roads, and allow the construction *ab initio* of any reasonable number of villages plus an easy way to build one or two way roads between villages. It should allow adding a new village to an existing setup, and permit deletion of individual villages (on deletion of a village, two options should be provided: (1) any roads that went through the village should also be deleted, or (2) any roads that went through the village *en route* to other villages should be made direct (for example, if there is a road from A to B and roads from B to C and from B to D, then in case (2) there will now be a road from A to C and A to D). There should also be a way to place a gnome in a specified village, and a button provided to (1) make the gnome move randomly amongst adjacent villages, and (2) a way to tell the gnome to move to a specified adjacent village. (The user should be able to see in which village a gnome is residing.) Be aware that you will want to handle multiple gnomes moving in their own threads, so do think carefully about your data structures and how to keep your code modular. You should also provide the capability of searching (efficiently) for any given gnome and so locating their current position and also acquiring a history of the places they've visited.

There should be intuitive ways to build gnomes, and to add/delete villages and roads. The roads (and perhaps even the villages) should have costs (tolls ... collected by trolls?) associated with them. Enhance your program

by having the gnomes know their current location and a desired location, plus a measure of urgency (ranging from lazy to desperate), so will choose their routes to their destinations accordingly. The multiple gnomes should move around in their own threads, and there should be reasonable limits on the number of gnomes permitted to rest in any one village and on the number of gnomes allowed to be in any given road at any one time (limits should vary by village and by road). Your program should be able to run a simulation of the gnomes travelling around. *You might like to explore ways of giving feedback to gnomes, either treating them as individuals or by devising a more global traffic control system, so that if they're not in lazy mode and so keen to get to their destination more quickly or more cheaply, they can reroute themselves to avoid being delayed when confronted with actual or potential traffic jams*. You will find yourself implementing a shortest path algorithm to resolve the above, but you should also implement

1. topological sort (for cases where a gnome would not be permitted to enter a particular village without having had its passport stamped by certain other villages)
2. minimal spanning tree (to allow the Gnomenwald government to minimise their investment in their total road network ... before a road is built from one village to another, the two villages will submit a building cost proposal to the country's government (you may assume that the cost of building a road will be 100 times the toll charged for using that road), and the government will then pick from the list of proposals those which minimise the total cost of building enough roads to connect all the villages (at least to connect all those which could be connected if all the proposed roads were built -- here, connected is allowed to mean connected via other villages; we're not requiring direct roads between all villages!!!). Notice that this raises the question of how to handle the case of a collection of villages being created and submitting a road-building proposal which connect to any of the old villages after a set of roads have already been built. Likewise, there are questions when villages or roads are deleted.

Add such additional features as you find interesting.

Question 2:

Recall the banking/economy simulation question from the second homework. You should revisit that question, refining it in natural ways. In particular, your program should be multi-threaded, allowing quasi-simultaneous transactions. All transactions must be assured to be atomic and thread-safe. For this model, every transaction has an associated cost (whether you call this a tax or a profit is up to you). There should be multiple currencies (so conversion between currencies is technically a transaction, and hence has a corresponding cost) – a consequence of this is that there may be multiple ways of conducting an international transaction: e.g., a direct currency exchange (dollars to yen), or several different sets of currency exchanges (dollar to sterling to euro to rand to yen, or dollar to rouble to renmimbi to yen), so it might be that there's an overall cheaper route to conducting the desired currency conversion depending on the transaction costs and differences in the buy-sell rates.

Your simulation should have an intuitive GUI, both to control the simulation as well as display its behaviour. Certain transactions, when over a given threshold, may require authorisation by more than one person or entity (we'll discuss nifty tricks to facilitate this in class on Monday).

(More detail might be added to this question as a result of discussion in class on Monday.)