CS 180 Homework 3

Due *Sunday, August* 1, 2021

Problem. Men, women and gupi live on the planet Alphaomega. Imagine you are travelling by a self-driving car on this planet and you need to pass the city Beth. The road goes through this city and you cannot bypass it. There are only two entrances to the city – one from the side you are coming and another from the opposite side. Besides, you know that an evil gupi is sitting at one of the intersections in Beth. When a human being comes close to him, he kidnaps this person and makes the human being his slave. Fortunately, this gupi is lazy and he kidnaps only those who come close to him. The distance in one block is safe for you because your self-driving car can feel the presence of the gupi one block ahead of it. At the same time, other inhabitants of Beth cannot help you because you don't know their language and they don't know your language. Besides, you don't have a map of the city Beth.

a) (20 pts). Is it always possible to go through the city? Prove that your answer is correct.

b) (80 pts). If it is always possible to go through the city, design an algorithm for your self-driving car for doing this.

c) (80 pts). If it is not always possible to go through the city, design an algorithm for your self-driving car that will allow you to go through the city when it is possible or to return back when it is impossible.

Homework 3

a) No, it’s not always possible to go through the city; Let’s assume that Gupi is sitting at the only node/intersection that allows you to pass the city (for example, at the last & only intersection that leads to the exit), then you must return since crossing is not possible.

b) Not always possible. SKIP

c) Let’s to backtrack when I am one edge away from Gupi and it’s not possible to go through the city; If possible, then we will find the possible path through the city. Again, if not, return to the starting point.

Pseudocode:

```
Search(G)
    For each vertex v ϵ G.V
        v.mark = unvisited            //mark all vertices un-visited
    For  each vertex v ϵ G.V
        If v.mark == unvisited.        //start with each unvisited, visit them
              Visit(G,v)
    For all vertex v ϵ G.V               //if all vertices are checked and explored
        If v.mark == completely explored
              If no exit is found
                    Return no exit found;
    End Search(G)
```

```
Visit(G,v)
    v.mark = found but not explored  //mark that vertex v found but not explored
    For each u ϵ G.Adj[v]          // for each edges adjacent to the vertex v, check
        If u.mark == unvisited
              If my car senses Gupi’s presence at u
                    Delete (v,u) from the graph G     //omit from graph due to Gupi
              Else
                    Visit(G,u)
    v. mark = completely explored
    If exit is found
        Return exit found;
```

```
End Visit(G, v)
```