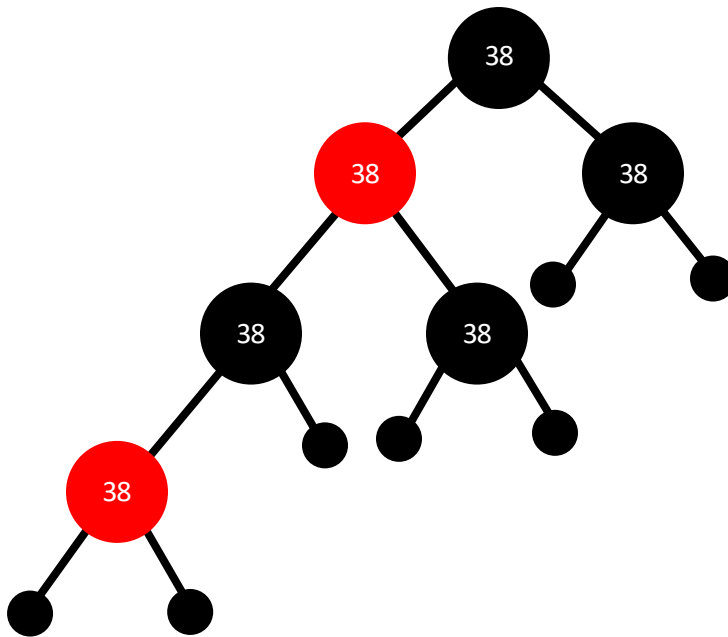Homework 8, CPSC 4100 – 01, Winter 2017

I have not received unauthorized aid on this assignment. I understand the answers that I have submitted. The answers submitted have not been directly copied from another source, but instead are written in my own words.

1) Show the red-black trees that result after successively inserting the keys 41, 38, 31, 12, 19, 8 into an initially empty red-black tree.

2) Rewrite the procedure DFS, using a stack to eliminate recursion.

```
DFS(G)
1 for each vertex u in G.V
2       u.color = WHITE
3       u.π = NIL
4 time = 0
5 stack = {}
6 for each vertex u in G.V
7       stack.push(u)
8       while stack isNot empty
9               u = stack.top()
10              if u.color == WHITE
11              time = time + 1
12                      u.d = time
13                      u.color = GRAY
14                      for each v in G.Adj[u]
15                              if v.color == WHITE
16                                      v.π = u
17                                      stack.push(v)
18              if u.color == GREY
19                      u.color = BLACK                         // blacken u; it is finished
20                      time = time + 1
21                      u.f = time
22                      stack.pop(u)
```

3) Design an efficient algorithm that gets a graph G (V, E) and returns "yes" if it is possible to have a Eulerian tour in G. Otherwise it returns "no".

1   If graph has an odd number of vertices, return "no"
2   Consider the graph represented as an adjacency matrix
3   If the graph is an undirected graph
        a. If the row totals are all even return "yes"
4   If the graph is a directed graph
        a. If the row total for each vertex = the column total for the vertex return "yes"

4) Assume that in each node of a graph an integer value has been stored. Design an efficient algorithm that given a graph G (V, E) and value X returns true if X is in G. Otherwise returns false.

1   Set a Boolean value to false
2   Traverse through the graph using a depth first search
3   As each node is visited, compare the integer value stored to x
4   If the integer value is equal to x, set the Boolean value to true

5   After traversing the graph, return e Boolean value.