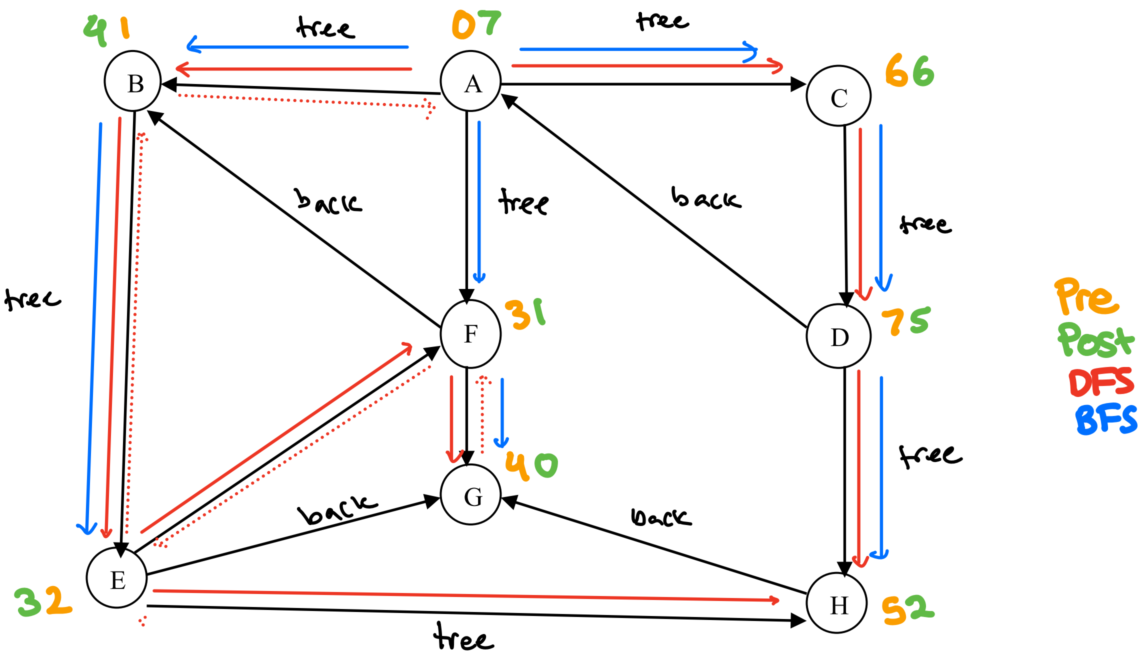
OCS 25 - Data Structures and Algorithms
Programming Assignment - 3
Directed graphs

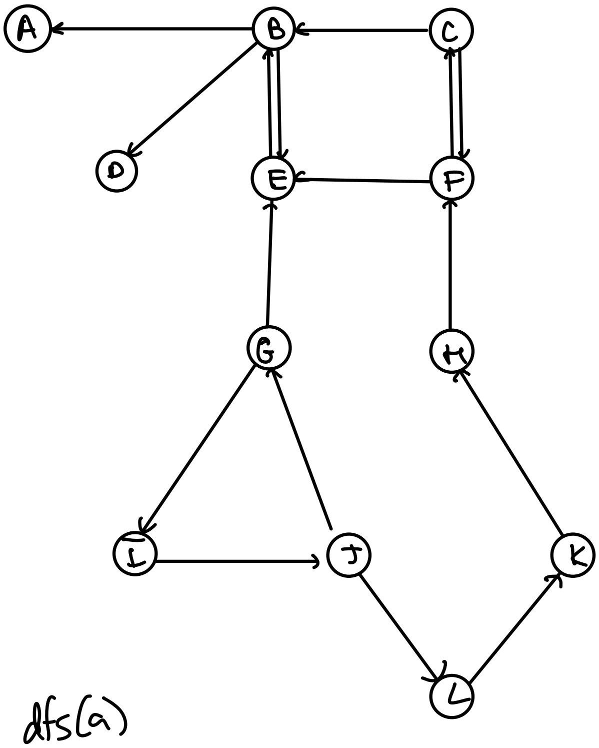
1. Depth First Search and Breadth First Search

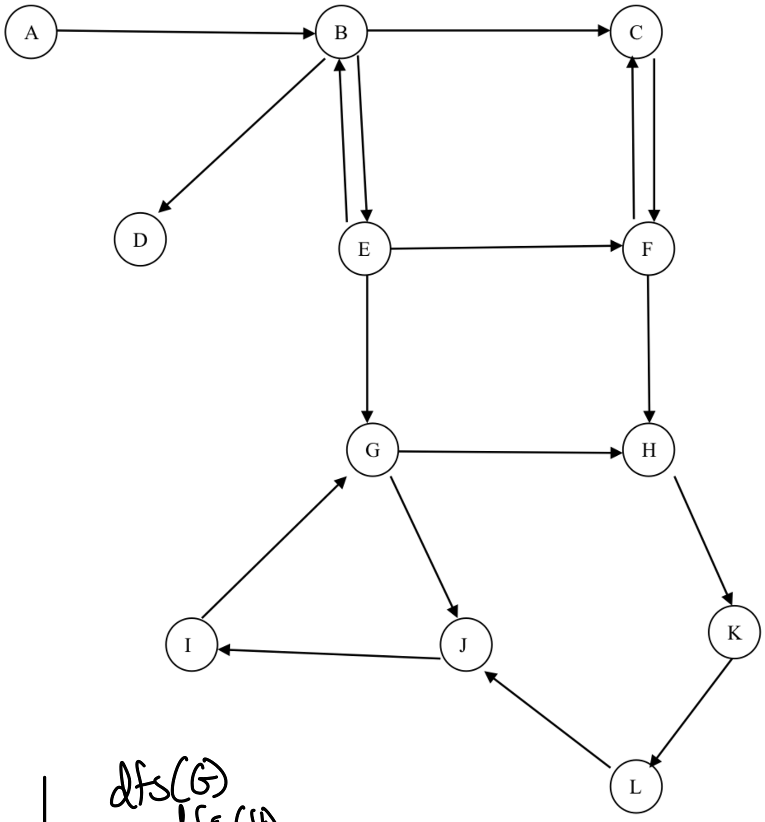
Perform a depth-first and a breadth-first search on the following graph; whenever there's a choice of vertices pick the one that is alphabetically first. Classify each edge as a tree edge or a back edge, and give **pre** and **post** number of each vertex. Start from vertex A.



2. Connected Components

Run a strongly connected components algorithm on the following directed G. Whenever there is a choice of vertices to explore, always pick the one that is alphabetically first. Start from vertex A. Show the trace at every step as shown in the textbook.





```
dfs(a)
a done
dfs (b)
   dfs(d)
    d done
   dfs (e)
    e done
dfs(c)
   dfs(f)
   f done
c done
dfs (G)
    dfs (I)
       dfs(J)
         dfs(L)
           dfs(K)
              dfs(H)
                H done
              K done
           L done
         J done
     I done
G done
```

Strong Component 1

```
dfs(G)
    dfs(H)
       dfs(K)
          dfs(L)
             dfs(J)
                dfs(I)
                I done
             J done
          L done
       K done
    H done
G done
```

Component 2 | dfs (f)
f done

Component3 | dfs(c)
c done

Component 4 | dfs (e)
e done

Component 5 | dfs(d)
d done

Component 6 | dfs (b)
b done

Component 7 | dfs (a)
a done

3. Programming Assignment

Write a program to determine the connected components in the graph shown on page 591 in the textbook. In case of multiple links between two nodes in the same direction, you can replace them with a single link.

I have attached my code in Tester.java.