NAMES: Bingcong Li, Alex Asare, Dakota Krogmeier

Worksheet 17: Work with the other students in your Zoom breakout room to complete this worksheet. You should only submit one paper for the group.

For this assignment, we will focus on Euler Paths and Euler Circuits. First, we need to define terminology. For this assignment, we will consider graphs that are connected, undirected, and un-weighted.

Given a graph G = (V, E), an Euler **path** on G is a path satisfying the following criteria: (0.5 points)

An Euler path is a path that uses every edge of a graph exactly once. starts and ends at diff vertex.

Given a graph G = (V, E), an Euler **circuit** on G is a path satisfying the following criteria: (0.5 points)

An Euler circuit is a Euler path which starts and stops at the same vertex and touches every edge once.

Given a graph G = (V, E), are there characteristics of G that make it possible to determine if G admits an Euler Circuit? If so, state this characteristic. How would you represent G in order to identify this characteristic? Explain your reason for choosing this data structure. (0.5 points)

A graph has an Euler circuit if and only if the degree of every vertex is even.

We decided an adjacency matrix would be easiest to check, as we could follow each row to count the number of edges.

Given a graph G = (V, E), are there characteristics of G that make it possible to determine if G admits an Euler Path? If so, state this characteristic. How would you represent G in order to identify this characteristic? Explain your reason for choosing this data structure. (0.5 points)

A graph has an Euler path if and only if there are at most two vertices with odd degree.

We landed on an adjacency matrix again, due to the ease in finding edges between vertices.

Given a connected, un-weighted, and undirected graph G = (V, E), describe an algorithm that can be used to determine if G admits an Euler Path that incorporates your KEY IDEA above, as well as the data structure you chose to represent your graph. (4 points)

Algorithm EulerPath(*G = (V, E)*):

Input: A graph, G = (V, E) represented as an adjacency matrix that is connected

Output: A Boolean True if G admits an Euler Path, and False if G does not admit an Euler Path

PROCESS:

```
oddDegree <- 0

countDegree <- 0

for vertex i in graph

        for vertex j in graph that are connected to i

                increase countDegree by 1

        if countDegree % 2 != 0

                increase oddDegree by 1

if oddDegree > 2

        return False

if oddDegree = 0

        return True

else

        return True
```