This repository contains 2 programs coded in python.

The first program (part A) can generate a digraph based on two user inputs as system arguments; the number of vertices and if the graph is connected or not. If there are no system arguments then the digraph will be generated with a random number of vertices and randomly generated connections. It creates the digraph using a dictionary and names the vertices numerically. Output for the digraph is in the form of each edge printed in the format “a b” meaning that there is an edge in the direction of vertex a to b.

The second program (part B) will take user inputs in the form of edges formatted as “a b” to create a digraph. It does this by splitting the vertices entered and counts the vertices while also checking if the edges entered are duplicates. Then the program will print out the number of vertices, number of edges, whether the graph is connected or not, if the graph is a Directed Acyclic Graph (DAG) or not, and can display a path from a starting vertex to end vertex if one can be found. The starting vertex and ending vertex are inputted as system arguments and if none are inputted the program will not find a path. The digraph is implemented using an adjacency list stored in a dictionary and uses three variations of Depth First Searches. The first variation is used to find a path from a starting vertex to the ending vertex. The second variation is used to check if there is a cycle in the graph and subsequently check if it is a DAG or not. Then the third variation is used to verify if the graph is connected or not.