Alex Asch
Assignment 4 Design
Description of Program
This program will begin by making a graph to track paths.
Notes

- Step 1: How we interact with the Graph struct
    - The graph struct is a struct with an integer number of vertices, a boolean for directed or undirected, a boolean array for if vertices are visited, and a matrix (2d array) which stores the vertices
    - The graph itself is an i by j array, and if there is a nonzero value at (i,j) this means that there is a path of length m(i,j) from i to j.
        - If the graph is undirected, that means for every path from i to j there is a path of the same length from j to i.
    - We will write creation and deletion functions for the graph struct.
        - The creation struct, called a constructor, takes two arguments, the number of vertices and the boolean if the graph is directed or undirected.
        - This allocates the appropriate amount of memory for a graph struct using calloc to make sure the memory is zeroed, it then initializes the number of vertices and the directedness create the graph
        - It returns a pointer value to the graph struct.
        - The deletion function.
            - The deletion function takes a pointer to a pointer of G, and frees the memory at point g, it also sets the pointer to null to prevent use after free errors.
    - The graph_vertices function.
        - This function will simply return the number of vertices a graph has. This is because we are creating opaque data types, which cannot be accessed outside of their implementation.
        - This function ensures there is a way for the user to access the values and data, but only when we want them to be able to.
    - The graph add edge function.
        - This adds an edge at (i,j) of weight k, taking all above values as integer values.
        - This will mirror this point across the value for undirected graphs.
        - For built-in error checking, this function checks if the vertices are in bounds and the edges are non negative, and returns a boolean value, which is based on successful creation of edges.
    - The graph has edge function.
        - This returns true if theres is an edge between the values i and j that are in the bounds of the matrix.
    - The graph_edge_weight function
        - This returns the weight of the graph at the vertices, if it is out of bounds or doesn't exist return zero.
    - The graph_visited function.

        - Return true if the vertex v has been visited false otherwise.
    - The graph_mark_visited function
        - If vertex v is within bounds, mark v as visited.
    - The graph_mark_unvisited function
        - Unvisit the vertex v.
    - graph_print
        - This is a debug function for making sure our implementation of the graph ADT is sound.