Problem 3

For my representation, I chose to represent the graph as two parts. The first is a list that contains all of the nodes/vertices in the graph. The second part is a list of Edges (edge as a class). There are also counter to keep track of the size of these lists. I chose this representation by expanding on the idea of just a collection of edges. By adding the list of vertices to the implementation it will allow me to keep better track of what is in the graph for future calculations and methods for the graph. The counters were also added for this reason, the made it easier to test and examine the graph in more complex methods/tests.

The adjacency list implementation is more advantages with edge calculations because the node data structure would be centered around the edges, so if you were looking for more information about specific edges in the graph it might be more optimal.

The adjacency matrix would be advantageous in that the access time for getting edge labels would be fast, and also more optimal for representing a graph of a known size. To make it the most optimal using a dynamically scaleable data structure for the list would be better as something like an ArrayList would need to copy over all the data which could be problematic.