Assignment 01: makeAdjMatrixGraph Timing

In this assignment, students were asked to write two functions that will return a graph given a number of nodes and a probability that each new node will be connected to all previously added nodes. One function will return a graph using an adjacency matrix, the other returns one with an adjacency list. Students were then asked to test the function by timing a depth-first traversal of the returned graph with 5 different sizes.

In my repository, these were implemented in an hpp file labeled “GrpahMaker.hpp” and was tested with makegraph.cpp. It should also be noted that when trying to create a graph of 1024 nodes, a stack overflow exception is thrown. Remedying this requires rewriting the AdjMatrixGraph class to dynamically allocate the space on the heap for the graph, which I did not have time to do implement. For now, I have tested the timing with 1000 nodes instead.

The timings are recorded below for 5 trials. I believe the time complexity of my implementation is O(n logn). This is because of how the iterations are completed; when adding edges, the internal loop to do so only iterates from 0 to whatever number node is creating the outgoing edge. This steers away from quadratic behavior.

