# Project Description

This is a programming project that must be written in Java and turned in via Blackboard. The project has two parts. In the first part, you will implement a general graph interface. You may use any implementation method you wish that only utilizes the java.util library. Do not find a third-party graph library and use that.

In the second part, you will use your code from the first part to create a graph algorithm utility class that contains methods for topological sort, minimum spanning tree, and shortest path algorithms.

## *Part 1: (200 points)*

You must create a class that implements the `IGraph` interface provided to you on Blackboard. The name of your class must be `Graph` and reside in the `cs311.graph` package.

Your grade for this part will be based entirely on the correctness of your implementation of this interface. The names of the methods in the interface are self-explanatory, but if you are unsure of what a method should do, please ask on Blackboard.

At 15% of your grade will come from the documentation you write for this class. You may receive no credit if you do not follow the interface and its documentation. Do not change the `IGraph` interface since we will grade with the one in Blackboard. If there is an error in the interface that prevents you from completing part 2, it behooves you to post such problems early so that a correction can be made sooner rather than later. I did not intentionally place an error in the interface file this time, but errors are still possible.

## *Part 2: (300 points)*

Create a class that completes the GraphAlgorithm class skeleton provided in Blackboard. Again, documentation will be %15 percent of the grade. Do not change the method signatures of any of these static methods. Do not move this file from the graphalgorithms package. You may not receive any credit for the assignment if you do. You should test your implementations with your own JUnit tests or main methods. We will not provide any tests, but you may post tests to Blackboard for all to use. To encourage you to do this, we will utilize valid tests posted to Blackboard (at least half of the total number of tests we run) to test your projects after they are turned in.

*Turning in the assignment*

Turn in your assignment by creating a single zip file that unpacks to the two specified package directories defined in the project.