Problem 1 (7 points)
Make sure they have an overview (-1 if missing), abstraction function (AF) (-1 if missing), and represention function (-1 if missing) for the Graph class.

Make sure they have a spec for each non-obvious public method (-1 for each missing, no negative scores).

Here's a sample

```
public class Graph<String, String> {
      HashSet<String> nodes;
      HashMap<String,HashSet<Edge<String, String>>> edges;
    // AF (c) = < Nodes, Edges > where
    //     Nodes = { Graph.nodes }
    //     Edges = { {Graph.node1, Graph.node2 } }
    // i.e. nodes represents the set of graph nodes and edges
represents the set of edges connecting
    //      connecting two nodes.
    //
    // Rep invariant: nodes contains no nulls and no duplicates
(trivially enforced by Hashset)
    //                edges contains no nulls and no duplicate
edges, according to definition of "equals"
    //                all sources and targets of edges must be in
nodes
    // Note: Edge is defined in Edge.java
```

See https://www.cs.virginia.edu/~evans/cs201j/lectures/graph/impl1/Graph.java for a good example of AF and RI.

Give points for answers that make sense.

Problem 2. (2 points)
Explain the testing strategy

Give points for anything that makes sense.

Problem 3. (10 points)
They should describe the their rep and give advantages of each of the three reps.

Deduct 2 points for each description missing or that is not clear. Some judgment required. They should discuss efficiency or ease of implementaion for each case. (6 points total)

Deduct 1 point if they don't explain why they chose their represenation. Their own representation description should match their AF and rep from problem 1.

Check the code for their Graph. Deduct points if the code is missing an AF (-1 point), RI (-1 point), and checkRep (-1 point)

Collaboration and Reflection (1 point)
Deduct a point if missing.