**ALTERNATIVE BONUS**

**Q1. Write a Gremlin command that creates the above graph [hint - you will also need a 'traversal' for it]. The command could be a multi-statement one, or a single line one (with function chaining).**

**Solution:**

gremlin> graph = TinkerGraph.open()

==>tinkergraph[vertices:0 edges:0]

gremlin> g = graph.traversal()

==>graphtraversalsource[tinkergraph[vertices:0 edges:0], standard]

gremlin> g.addV('CS101').property(id,'CS101').as('v1').addV('CS201').property(id,'CS201').as('v2').addV('CS220').property(id,'CS220').as('v3').addV('CS334').property(id,'CS334').as('v4').addV('CS420').property(id,'CS420').as('v5').addV('CS681').property(id,'CS681').as('v6').addV('CS400').property(id,'CS400').as('v7').addV('CS526').property(id,'CS526').as('v8').addE('requires pre-req').from('v2').to('v1').addE('requires pre-req').from('v3').to('v2').addE('requires pre-req').from('v4').to('v2').addE('requires pre-req').from('v5').to('v3').addE('is a co-req of').from('v5').to('v3').addE('requires pre-req').from('v6').to('v4').addE('requires pre-req').from('v7').to('v4').addE('requires pre-req').from('v8').to('v7').addE('is a co-req of').from('v8').to('v7')

**Output:**

==>e[8][CS526-is a co-req of->CS400]

**Explanation:** As above.

**Q2. Write a query that will output JUST the doubly-connected nodes.**

**Solution:**

gremlin> g.V().as("a").out().as("b").select("a","b").groupCount().unfold().filter(select(values).is(gt(1))).select(keys)

**Output:**

==>[a:v[CS526],b:v[CS400]]

==>[a:v[CS420],b:v[CS220]]

**Explanation:** As above

**Q3. Write a query that will output all the ancestors (for us, these would be prereqs) of a given vertex.**

**Solution:**

gremlin> g.V().has(id,'CS526').repeat(out().dedup()).emit()

**Output:**

==>v[CS400]

==>v[CS334]

==>v[CS201]

==>v[CS101]

**Explanation:** As above.

**Q4. Write a query that will output the max depth starting from a given node (provides a count (including itself) of all the connected nodes till the deepest leaf). This would give us a total count of the longest sequence of courses that can be taken, after having completed a prereq course.**

**Solution:**

gremlin> g.V().has(id,'CS101').repeat(\_\_.in().dedup()).emit().tail(1).path().count(local)

**Output:**

==>5

**Explanation:** As above.