Quiz 11 (Dec 6)

By taking this quiz, you agree to adhere to the honor code of the class.

Name: netid:

Name: netid:

Say you have an `EdgeListGraph<V, E>` with n vertices, m edges, and maximum degree d. Write a method `public boolean find_abc(V a, V b, V c)` for `EdgeListGraph` that returns true if there are three vertices A, B, C that hold values a, b, c and such that B is the child of A and C is the child of B (i.e. A -> B -> C). Ensure and justify the runtime is O(nd).

Simplified from github

```
public class EdgeListGraph<V, E>{
	public class Vertex<V> {
		V element;
	}
	public class Edge<E, V>{
		E element;
		Vertex<V>[] endpoints;
	}
	DoublyLinkedList<Vertex<V>> vertices;
	DoublyLinkedList<Edge<E, V>> edges;
	int n_vertices;
	int n_edges;
	List<Edge<E, V>> outgoingEdges(Vertex<V> v);
	List<Edge<E, V>> incomingEdges(Vertex<V> v);
	Edge<E, V> getEdge(Vertex<V> from, Vertex<V> to);
	Vertex<V>[] endVertices(Edge<E, V> e);
	Vertex<V> opposite(Vertex<V> v, Edge<E, V> e);
}
```

```
public boolean find_abc(V a, V b, V c) {
        for (Vertex v2: vertices) {
                if (v2.element.equals(b)) {
                        for (Edge e1: outgoingEdges(v2)) {
                                Vertex v3 = opposite(e1, v2);
                                If (v3.element.equals(c)) {
                                        for (Edge e2: incomingEdges(v2)) {
                                                Vertex v1 = opposite(e2, v2);
                                                If (v1.element.equals(a)) {
                                                        return true;
                                                }
                                        }
                                        break;
                                }
                        }
                }
        }
        return false;
}
```

The first for-loop will iterate through all vertices, so O(n). The inner two for-loops look like O(d) * O(d) = O(d^2), but it's actually O(2d) because of the "break". The innermost loop is guaranteed to run at most once. So the total runtime is O(nd);