The lab and homework this week center on graph algorithms for undirected graphs. The following definitions might be helpful/relevant.

- A path $P$ on a graph $G = (V, E)$ is a sequence of vertices $P = (v_1, v_2, \ldots, v_k)$ such that $(v_i, v_{i+1}) \in E$ for all $1 \leq i < k$.
- A path is simple if all vertices are distinct.
- The length of a path $P = (v_1, \ldots, v_k)$ equals $k - 1$. (Think of the path length as the number of edges needed to get from $v_1$ to $v_k$ on this path).
- A cycle is a sequence of vertices $(v_1, \ldots, v_k)$ such that $v_1, \ldots, v_{k-1}$ are all distinct and $v_k = v_1$. A cycle is odd (even) if it contains an odd (even) number of edges.

1. **Cycle Detection.** Design and analyze an efficient algorithm for finding a cycle in a graph. Your algorithm should take as input a graph $G = (V, E)$ and report a cycle (or output NO if no cycle exists). If there are multiple cycles in the graph, your algorithm should just output one.

2. **Testing Bipartiteness.** Call a graph $G = (V, E)$ bipartite if you can partition $V$ into sets $A$ and $B$ such that all edges $e \in E$ have one vertex in $A$, one in $B$. Design an analyze an algorithm to test a graph for bipartiteness.

   **Hint:** An alternate definition is that $G = (V, E)$ is bipartite if you can color vertices in $V$ by one of two colors so that each edge is bichromatic: for any $\{u, v\} \in E$, vertex $u$ is a different color from vertex $v$.

3. **Testing Tripartiteness.** Call a graph $G = (V, E)$ tripartite if $V$ can be partitioned into disjoint sets $A, B, C$ such that for any edge $(u, v) \in E$, the vertices $u, v$ lie in different sets. Design and analyze an algorithm to test a graph for tripartiteness.