Problem 1  (a) Run BFS algorithm on the directed graph below, using vertex A as the source. Show all distances and the BFS tree.

(b) Run DFS algorithm on the directed graph below, using vertex A as the source. Compute the start time and the finish time of each vertex and draw the DFS tree.
Problem 2 (CLRS 22.1-6) Most graph algorithms that take an adjacency-matrix representation as input require time $O(n^2)$, but there are some exceptions. Show how to determine whether a directed graph $G$ contains a universal sink, i.e., a vertex with in-degree $n - 1$ and out-degree 0, in time $O(n)$ given an adjacency matrix for $G$. 
Problem 3 Suppose we have an undirected graph and we want to color all the vertices with two colors red and blue such that no two neighbors have the same color. Design an $O(n + m)$ time algorithm which finds such a coloring if possible or determines that there is no such coloring.

(a) Prove that if the graph has a cycle of odd length, there is no such a coloring.
(b) Assume the graph has no cycle of odd length. Use BFS algorithm to find an appropriate coloring.
Problem 4  (a) (CLRS 22.3-9) Give a counterexample to the conjecture that if a directed graph $G$ contains a path from $u$ to $v$, then any depth-first search must result in $v.start \leq u.finish$.

(b) (CLRS 22.3-8) Give a counterexample to the conjecture that if a directed graph $G$ contains a path from $u$ to $v$, and if $u.start < v.start$ in a depth-first search of $G$, then $v$ is a descendant of $u$ in the depth-first forest produced.
**Problem 5 (CLRS 22.3-7)** Implement DFS using a stack to eliminate recursion. Write a pseudo-code for your algorithm.
Problem 6 Given a directed acyclic graph $G$, design an $O(n + m)$ time algorithm which finds the length of the longest path of the graph.

(a) Find a topological sort of the given DAG and let $v_1, v_2, \ldots, v_n$ be a topological sort, i.e., each edge is from a vertex $v_i$ to another vertex $v_j$ with $j > i$. Let $A[i]$ be the longest path of the graph starting at $v_i$. Find a formula for computing $A[i]$.

(b) If we compute $A[1], A[2], \ldots, A[n]$, what would be the final solution?

(c) Write a dynamic program for filling array $A$. What is the running time of this algorithm?

(d) Run DFS and compute $A[i]$ during DFS($v_i$). How do you compute $A[i]$ during DFS($v_i$)? What is the running time of this algorithm?
Problem 7 (CLRS 22.4-5) Another way to perform topological sorting on a directed acyclic graph $G$ is to repeatedly find a vertex of in-degree 0, output it, and remove it and all of its outgoing edges from the graph. Explain how to implement this idea so that it runs in time $O(n + m)$.
Problem 8 (CLRS 22.2-8) The diameter of a tree $T = (V, E)$ is defined as the length of the longest path in the tree. Give an $O(n)$ algorithm to compute the diameter of a tree, and analyze the running time of your algorithm.

*Hint 1:* We have $m = n - 1$ in a tree, and thus $n + m = \Theta(n)$.

*Hint 2:* Pick an arbitrary vertex $r$ as the root. Run DFS algorithm from $r$. For each vertex $i$ define $A[i]$ as the length of the longest path in the subtree rooted at $i$, and $B[i]$ as the length of the longest path in the subtree rooted at $i$ which has vertex $i$ as one of its end points. Show how to compute $A[i]$ and $B[i]$ for each vertex $i$ during DFS$(i)$. 