Problem 1

You are given an array implementation of a binary max-heap containing \( n \) elements and a real number \( x \). Design an \( O(k) \) time algorithm to determine if the \( k \)th largest element in the heap is less than or equal to \( x \). (Hint: You are not asked to find the \( k \)th largest element itself)

Problem 2

You are given a directed acyclic graph \( G = (V, E) \) and two vertices \( s \) and \( t \). Design a linear time algorithm to count the number of directed paths from \( s \) to \( t \).

Problem 3

Let \( G = (V, E) \) be a connected, undirected graph. Design a linear time algorithm to find a vertex whose removal does not disconnect the graph.

Problem 4

Give an \( O(n \log k) \)-time algorithm to merge \( k \) sorted lists into one sorted list, where \( n \) is the total number of elements in all the input lists.

Problem 5

A graph \( G = (V, E) \) is called semi-connected if for all \( u, v \in V \) either there is a path from \( u \) to \( v \) or there is a path from \( v \) to \( u \). Design a linear time algorithm to test whether \( G \) is semi-connected. (Hint: Can you solve this problem easily on a DAG?)

Problem 6

When an adjacency matrix representation is used, most graph algorithms require time \( \Omega(n^2) \), but there are some exceptions. Show that determining whether a directed graph \( G \) contains a universal sink – a vertex with in-degree \(|V| - 1\) and out-degree 0 – can be determined in time \( O(V) \) given an adjacency matrix for \( G \).

---

1You may work with a partner, but you must submit your own solutions. You may NOT consult any textbook other than CLRS or the internet to aid in solving these problems.