1. (10%) Write a recursive program for binary search. That is, given a list of elements \( b[0], b[1], \ldots, b[n-1] \), you need to search for a particular element \( X \) in the list using the binary search strategy.

2. (10%) Answer the following questions about complexity.
   a. (5%) Define the "Big-O" notation. What does \( f(n) = O(g(n)) \) mean?
   b. (5%) What is \( 6^2 + 4n^2 \) under the "Big O" notation?

3. (10%) Show how to define the data structure of a circular queue using an array. Show how to add an element and how to delete an element from the queue.

4. (10%) You are given the binary expression: \( a/b + c + d * (e + a + c) \)
   a. (5%) What is its postfix expression?
   b. (5%) What is its prefix expression?

5. (10%) Suppose we want to use a linked list to represent a polynomial (for example, \( P(x) = 3x^4 + 2x^3 + 5x^2 + 1 \)).
   a. (5%) Define its data structure.
   b. (5%) Write a program to multiply the polynomial by \( x^2 \). For example, \( x^2 * P(x) = 3x^6 + 2x^4 + 5x^4 + x^2 \).

6. (20%) Answer the following questions about trees.
   a. (5%) Show how to use "left-child-right-sibling" to represent a tree.
   b. (5%) What is a complete binary tree?
   c. (5%) What is a full binary tree?
   d. (5%) How to use an array to represent a complete binary tree?

7. (10%) Define a heap. Show how to insert a node to a heap. Show how to delete a node from a heap.

8. (10%) Show how to represent a graph by an adjacency matrix.

9. (10%) Define "depth-first search" and "breadth-first search".