Data Structures (R1UC308B)
Practice Questions - MTE

Set 1

1. Define asymptotic notation and explain its importance in analyzing algorithm efficiency. (Low, K1, 3 marks)
2. Explain the difference between tail recursion and head recursion with examples. (Low, K1, 3 marks)
3. Derive the index formula for accessing elements in a 2-D array stored in row-major order. (Low, K2, 4 marks)
4. Explain the difference between linear search and binary search with their time complexities. (Low, K2, 4 marks)
5. Write the algorithm and explain the working of insertion sort with an example. (Medium, K3, 5 marks)
6. Discuss how sparse matrices are represented and explain any one representation method. (Medium, K3, 5 marks)
7. Explain the process of reversing a singly linked list with an example. (Medium, K4, 8 marks)
8. (a) Write a recursive algorithm for the Tower of Hanoi problem and explain it.
   (b) Compare the trade-offs between recursion and iteration. (Medium, K4, 8 marks)
9. Case study: Given an array of integers, design an efficient method using merge sort to sort it. Explain its time complexity and benefits over bubble sort. (High, K5, 10 marks)
   OR
   Case study: Explain the application of doubly linked lists in polynomial representation. Demonstrate insertion and deletion operations. (High, K5, 10 marks)

Set 2

1. Define time-space tradeoff with an example. (Low, K1, 3 marks)
2. Discuss the applications of multidimensional arrays with suitable examples. (Low, K1, 3 marks)
3. Derive the index formula for a 1-D array and explain its significance. (Low, K2, 4 marks)
4. Write and explain the linear search algorithm. What are its advantages and limitations? (Low, K2, 4 marks)
5. Explain quick sort algorithm and give an example of partitioning. (Medium, K3, 5 marks)
6. Discuss the different types of recursion with examples. (Medium, K3, 5 marks)
7. Explain insertion and deletion operations in doubly linked lists with diagrams. (Medium, K4, 8 marks)

8. (a) Write an iterative solution for calculating Fibonacci numbers.
   (b) Explain the removal of recursion with an example. (Medium, K4, 8 marks)
9. Case study: Given a sparse matrix, demonstrate how it can be stored using linked list representation. Explain the benefits of this approach. (High, K5, 10 marks)
   OR
   Case study: Discuss the role of quick sort in real-time applications. Compare it with merge sort in terms of efficiency. (High, K5, 10 marks)

Set 3

1. Explain the concept of algorithm efficiency and how it is measured. (Low, K1, 3 marks)
2. Differentiate between time complexity and space complexity with examples. (Low, K1, 3 marks)
3. Derive the index formula for accessing elements in a 3-D array in column-major order. (Low, K2, 4 marks)
4. Write the algorithm for binary search and explain its steps. (Low, K2, 4 marks)
5. Describe bubble sort and explain its working with an example. (Medium, K3, 5 marks)
6. Write a recursive method for calculating factorial of a number and explain it. (Medium, K3, 5 marks)
7. Explain circular linked lists and write the algorithm to traverse it. (Medium, K4, 8 marks)
8. (a) Discuss the trade-offs between recursion and iteration with reference to memory usage.
   (b) Write a recursive algorithm for the Fibonacci series. (Medium, K4, 8 marks)
9. Case study: Given an unsorted array, explain how merge sort sorts it efficiently. Also discuss real-world applications where merge sort is preferred. (High, K5, 10 marks)
   OR
   Case study: Describe the implementation and application of singly linked lists in managing polynomial expressions. (High, K5, 10 marks)

Set 4

1. Define algorithm and explain the importance of efficiency in algorithm design. (Low, K1, 3 marks)
2. Explain row-major and column-major representation of arrays with suitable diagrams. (Low, K1, 3 marks)
3. Derive the general index formula for multi-dimensional arrays. (Low, K2, 4 marks)
4. Compare insertion sort and selection sort with respect to their time complexity. (Low, K2, 4 marks)
5. Explain merge sort algorithm with an example. (Medium, K3, 5 marks)

6. Discuss removal of recursion and write an iterative algorithm equivalent to the recursive factorial function. (Medium, K3, 5 marks)
7. Explain insertion operation in singly linked list with an example. (Medium, K4, 8 marks)
8. (a) Write a recursive solution to Tower of Hanoi problem and explain the logic.
   (b) Discuss the advantages and disadvantages of recursion in problem-solving. (Medium, K4, 8 marks)
9. Case study: With an example, explain sparse matrix representation using a 2-D array and discuss its advantages and limitations. (High, K5, 10 marks)
   OR
   Case study: Discuss the use of doubly linked lists in complex data management scenarios, with examples of traversal and deletion operations. (High, K5, 10 marks)

Set 5

1. Define complexity and describe the different types of complexities used to evaluate algorithms. (Low, K1, 3 marks)
2. Write short notes on types of recursion with examples. (Low, K1, 3 marks)
3. Derive the index formulae for a 2-D array in row-major order. (Low, K2, 4 marks)
4. Write a brief note on searching techniques and explain binary search. (Low, K2, 4 marks)
5. Explain quick sort algorithm and give a real-life example where it can be used. (Medium, K3, 5 marks)
6. Write an iterative algorithm for calculating Fibonacci numbers and explain it. (Medium, K3, 5 marks)
7. Explain circularly linked list and its applications. Write the algorithm for deletion in a circular linked list. (Medium, K4, 8 marks)
8. (a) Compare recursive and iterative approaches with reference to stack usage.
   (b) Write a recursive program to compute factorial of a number. (Medium, K4, 8 marks)
9. Case study: Explain the application of linked lists in polynomial representation. Demonstrate addition of two polynomials using linked lists. (High, K5, 10 marks)
   OR
   Case study: Write the algorithm for merge sort and explain its divide and conquer approach. Discuss its significance in large dataset sorting. (High, K5, 10 marks)