Indian Institute of Science Education and Research Thiruvananthapuram
Data Structures (DSC 314)
Quiz 1 (January 23, 2025)

Time: 30 min Max. marks: 20

Answer all questions. Each question carries 2 marks.

1 For each of the following pieces of code, find the number of times **op()** is called as a function of the input size n. Express your answer in terms of the **Big-Theta** notation. Also, justify your answer.

(a) for $(i = 10; i < n + 5; i+ = 2)$
op();

Answer: $\mathcal{O}(n)$

(b) for $(i = 1; i < n; i* = 2)$
op();

Answer: $\mathcal{O}(logn)$

2 Express the function $f(n) = (n^3)/1000 - 100n^2 - 100n + 3$ in terms of θ notation.

Answer: $\theta(n^3)$

3 Which of the following functions provides the maximum asymptotic complexity?
a) $f(n) = n^{(3/2)}$ b) $f(n) = n^{(5/4)}$ c) $f(n) = nlogn$ d) $f(n) = 2^n$
Answer: $f(n) = 2^n$

4 Which of the following is a Divide and Conquer algorithm?
a) Bubble Sort b) Selection Sort c) Insertion Sort d) Quick Sort

Answer: Quick Sort

5 Which of the following sorting algorithms provides the best time complexity in the worst-case scenario?
a) Bubble Sort b) Merge Sort c) Insertion Sort d) Quick Sort

Answer: Merge Sort

6 Identify the sorting technique that compares adjacent elements in a list and switches whenever necessary.

a) Bubble Sort b) Merge Sort c) Insertion Sort d) Quick Sort

Answer: Bubble Sort

7 Among the following options which is the best sorting algorithm when the list is already sorted?

a) Bubble Sort b) Merge Sort c) Insertion Sort d) Quick Sort

Answer: Insertion Sort

8 What is the best case time complexity of the binary search algorithm? Justify your answer.

a) $\mathcal{O}(1)$ b) $\mathcal{O}(n)$ c) $\mathcal{O}(logn)$ d) $\mathcal{O}(n^{(3/2)})$

Answer: The best case time complexity of a binary search algorithm is $\mathcal{O}(1)$, which means constant time; this occurs when the target element is found at the middle index of the array, requiring only one comparison to identify it.

9 Show that the running time of Quick Sort is $\mathcal{O}(n^2)$ when the array A contains distinct elements and is sorted in decreasing order.

Answer: If the array is already sorted in decreasing order, then, the pivot element is less than all the other elements. The partitioning step takes $\theta(n)$ time, and then leaves with a subproblem of size $n-1$ and a subproblem of size 0. This gives us the recurrence $T(n) = T(n-1)+T(0)+\theta(n)$. Note that: $T(0) = \theta(1)$. So, in each level we will have n decreased by 1 and the partitioning cost is still $\theta(n)$, that leaves us with $\mathrm{T}(n-1) = T(n-2)+T(0)+\theta(n)$ again. Thus $T(n) = \mathcal{O}(n^2)$.

10 Consider the following sequence of push and pop operations on an initially empty stack S.

a) S = push(S,1);
b) S = push(S,2);
c) S = pop(S);
d) S = push(S,3);
e) S = push(S,4);
f) S = pop(S);
g) S = pop(S);
h) S = pop(S);

Which of the following is the correct order in which elements are popped?

a) 1, 2, 3, 4 b) 2, 1, 3, 4 c) 2, 3, 4, 1 d) 2, 4, 3, 1

Answer: 2, 4, 3, 1