COMPSCI 300: Programming II – Comprehensive Test (Modules 7–9)

Instructions: Answer all questions. Part A is multiple choice (2 points each). Part B is short answer (4 points each). Topics covered: Algorithm Analysis, Recursion, Sorting & Searching.

Part A — Multiple Choice (2 pts each)

1. Purpose of algorithm analysis?

Answer: To measure how runtime changes as input size grows (B)

2. Runtime of for-loop printing 1000 numbers?

Answer: O(1) (constant number of iterations)

3. Definition of worst-case scenario?

Answer: When the algorithm performs the most operations (B)

4. What must every recursive function include?

Answer: A base case and recursive case (B)

5. Which statement about recursion is TRUE?

Answer: Every recursive call adds a new stack frame (C)

6. Runtime of factorial recursion?

Answer: O(N) (B)

7. When is linear search best used?

Answer: When dataset is small or unsorted (B)

8. Sorting algorithm with best average runtime?

Answer: Merge Sort (B)

9. When does Insertion Sort perform fastest?

Answer: When the array is nearly sorted (A)

10. Poor pivot in QuickSort causes runtime?

Answer: $O(N^2)$ (B)

Part B — Short Answer (4 pts each)

11. Why does recursion often require more memory than iteration?

Each recursive call adds a new stack frame to the call stack, consuming additional memory until the base case is reached.

12. Write a recursive method sumArray to return sum of array elements.

Answer:

```
public static int sumArray(int[] arr, int n) {
  if (n == 0) return 0;
  return arr[n - 1] + sumArray(arr, n - 1);
}
```

13. Describe when Linear Search is more efficient than Binary Search.

Linear Search is preferred when data is unsorted or very small, since sorting first would cost more time.

14. Explain first 3 merge steps in Merge Sort for array of 8 elements.

Array is split into halves repeatedly until single elements remain; first merges combine pairs of elements into sorted 2-element arrays.

15. Analyze runtime of nested loops printing i + j.

Outer loop runs N times; inner loop runs up to i times. Total $\approx N(N-1)/2 \rightarrow O(N^2)$.