Problem Set

Multiple Choice Questions

1. Which sorting algorithm has a worst-case time complexity of O(nˆ2)?

   A. Merge Sort
   B. Quick Sort
   C. Bubble Sort
   D. Heap Sort

2. The time complexity of selection sort is:

   A. O(n log n)
   B. O(nˆ2)
   C. O(n)
   D. O(log n)

Long Answer Questions

1. Explain how the insertion sort algorithm works. Provide an illustration and analyze its time complexity.
2. Compare and contrast merge sort and quick sort algorithms. Discuss their time complexity, stability, and space complexity.
3. Describe the steps involved in building a max-heap using the heap sort algorithm. Analyze its time complexity.

Solution Set

Multiple Choice Questions

1. B. Bubble Sort
2. B. O(nˆ2)

Long Answer Questions

1. Insertion sort is an algorithm that builds the final sorted array one item at a time. It iterates through the list, comparing each element with the ones before it and swapping them if they are in the wrong order. This process is repeated until the entire list is sorted.

   Illustration:

   ```
   Initial List: 5, 2, 8, 3, 1

   Pass 1: 2, 5, 8, 3, 1
   Pass 2: 2, 5, 8, 3, 1
   Pass 3: 2, 3, 5, 8, 1
   ```

```
Pass 4: 1, 2, 3, 5, 8

Sorted List: 1, 2, 3, 5, 8
```

Time Complexity: In the worst case scenario, when the list is reversed, the time complexity of insertion sort is O(n^2). However, in the best case scenario, when the list is already sorted, the time complexity is O(n).

2. Merge sort and quick sort are both divide and conquer algorithms used for sorting.

   Merge Sort:

   - Time Complexity: Merge sort has a time complexity of O(n log n) in all cases. It divides the list into smaller sublists, recursively sorts them, and then merges them back together.
   - Stability: Merge sort is a stable sorting algorithm, meaning it maintains the relative order of equal elements during the merge phase.
   - Space Complexity: Merge sort has a space complexity of O(n) because it requires additional space to store the sublists during the merge phase.

   Quick Sort:

   - Time Complexity: In the average and best cases, quick sort has a time complexity of O(n log n). However, in the worst case scenario, when the pivot is always the smallest or largest element, the time complexity can be O(n^2). This can be mitigated by choosing a good pivot.
   - Stability: Quick sort is not a stable sorting algorithm. The order of equal elements can change during partitioning.
   - Space Complexity: Quick sort has a space complexity of O(log n) due to the recursive calls on smaller sublists. However, it has a worst case space complexity of O(n) when the recursion is not balanced.

3. Building a max-heap using the heap sort algorithm involves the following steps:

   - Starting with the last non-leaf node, sift down each node to its correct position to reestablish the max-heap property.
   - Repeat this process for each non-leaf node, moving from right to left and top to bottom.
   - After this process, the array will contain a max-heap, with the maximum element at the root and the remaining elements in a partially sorted order.

   Time Complexity: The time complexity of building a max-heap is O(n), where n is the number of elements in the heap. This is because each node is sifted down to its correct position at most once.

Note: The solution set is provided for reference only. The actual answers may vary depending on the specific context and content of the class notes.