

Assignment 1

1. What is the worst-case time complexity of Insertion Sort?

- A. $O(n)$
- B. $O(\log n)$
- C. $O(n^2)$
- D. $O(n \log n)$

2. What is the 'best-case' scenario for the Insertion Sort algorithm?

- A. The array is sorted in reverse order.
- B. The array contains many duplicate elements.
- C. The array is already sorted.
- D. The elements in the array are random.

3. In the worst-case scenario, where the value `valueToFind` is not present in the array, how many times will the comparison `if (a[i] == valueToFind)` be executed if the array size is $n = 20$?

```
boolean findValue(int a[], int n, int valueToFind) {  
    for (int i = 0; i < n; i++) {  
        if (a[i] == valueToFind) {  
            return true;  
        }  
    }  
    return false;  
}
```

- A. 1 time.
- B. 8 times.
- C. 16 times.
- D. 20 times.

4. Please count the total number of steps for the following code in the worst-case scenario, expressing your answer in terms of n .

```
int findMax(int a[], int n) {  
    int maxVal = a[0];  
    for (int i = 1; i < n; i++) {  
        if (a[i] > maxVal) {  
            maxVal = a[i];  
        }  
    }  
    return maxVal;  
}
```

5. In a standard linked list, also known as a "Chain", what does the link field of the very last node typically point to?
- A. A pointer to the first node.
 - B. A NULL (or 0) pointer.
 - C. A pointer to itself.
 - D. It is left undefined.
6. Assuming a linked list is pointed to by a first pointer, how would you access the third element (at index 2)?
- A. first->link
 - B. first->link->link
 - C. first
 - D. first->link->link->link