QUIZ

1. Which of the following is a dynamic programming problem?

A. Optimal substructure
B. Overlapping subproblems
C. Distinct Subproblems
D. Greedy approach

2. A greedy algorithm can be used to solve all the dynamic programming problems.

A. False
B. True

3. In dynamic programming, the technique of storing the previously calculated values is called

A. Memoization
B. Storing value property
C. Saving value property
D. Mapping

4. Which of the following problems should be solved using dynamic programming?

A. Binary search
B. Longest common subsequence
C. Mergesort
D. Quicksort

5. What happens when a top-down approach of dynamic programming is applied to any problem?

A. It increases both, the time complexity and the space complexity
B. It decreases both, the time complexity and the space complexity
C. It increases the time complexity and decreases the space complexity
D. It increases the space complexity and decreases the time complexity.

6. If a problem can be broken into subproblems which are reused several times, the problem possesses _____ property.

A. Overlapping subproblems
B. Optimal substructure
C. Memorization
D. Greedy

7. What is optimal substructure in longest common subsequence Longest Increasing Subsequence problem?

A. L(i) = 1 + max(L(j)) where 0 < j < i and arr[j] < arr[i]; or L(i) = 1, if no such j exists.
B. L(i) = 1 + max(L(j)) where 0 < j < i and arr[i] < arr[j]; or L(i) = 1, if no such j exists.
C. L(i) = 1 + max(L(j)) where 0 <= j < i and arr[i] < arr[j]; or L(i) = 1, if no such j exists.
D. L(i) = 1 + max(L(j)) where 0 <= j < i and arr[j] < arr[i]; or L(i) = 1, if no such j exists.

8. which is not Increasing Subsequence in array = {10, 22, 9, 33, 21, 50, 41, 60, 80}
   A. {9,21}
   B. {10, 33, 41, 80}
   C. {10,21,22,41,60}
   D. {33,50, 80}

9. What is the time complexity of this algorithm?

```
def count(S, m, n):
    table = [0 for k in range(n+1)]
    table[0] = 1
    for i in range(0,m):
        for j in range(S[i],n+1):
            table[j] += table[j-S[i]]

    return table[n]
```

A. O(nlogn)
B. O(mn)
C. $O(n^2)$