

## Dynamic Programming Algorithms - Assignment

### Task 1: Knapsack Problem

Write a function `int Knapsack(int W, int[] weights, int[] values)` in Java that determines the maximum value of items that can fit into a knapsack with a capacity `W`. The function should handle up to 100 items. Find the optimal way to fill the knapsack with the given items to achieve the maximum total value. You must consider that you cannot break items, but have to include them whole.

Code:

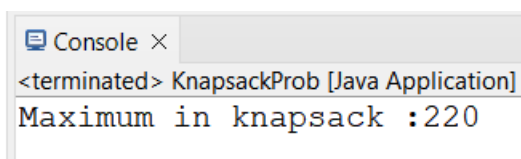
```
package knapsack;

public class KnapsackProb {

    public static int knapsack(int W, int[] weights, int[] values) {
        int n = weights.length;
        int[] dp = new int[W + 1];
        for (int i = 0; i < n; i++) {
            for (int w = W; w >= weights[i]; w--) {
                dp[w] = Math.max(dp[w], dp[w - weights[i]] + values[i]);
            }
        }
        return dp[W];
    }

    public static void main(String[] args) {
        int W = 50;
        int[] weights = { 10, 20, 30 };
        int[] values = { 60, 100, 120 };
        System.out.println("Maximum in knapsack :" + knapsack(W, weights, values));
    }
}
```

Output:



```
Console ×
<terminated> KnapsackProb [Java Application]
Maximum in knapsack :220
```

## Task 2: Longest Common Subsequence

Implement `int LCS(string text1, string text2)` to find the length of the longest common subsequence between two strings.

Code:

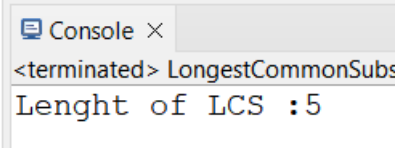
```
package longestcommonsubsequence;

public class LongestCommonSubsequence {

    public static int LCS(String s1, String s2) {
        int m = s1.length();
        int n = s2.length();
        int[][] dp = new int[m + 1][n + 1];
        for (int i = 1; i <= m; i++) {
            for (int j = 1; j <= n; j++) {
                if (s1.charAt(i - 1) == s2.charAt(j - 1)) {
                    dp[i][j] = dp[i - 1][j - 1] + 1;
                } else {
                    dp[i][j] = Math.max(dp[i - 1][j], dp[i][j - 1]);
                }
            }
        }
        return dp[m][n];
    }

    public static void main(String[] args) {
        String s1 = "abcde";
        String s2 = "acbacede";
        System.out.println("Lenght of LCS :" + LCS(s1, s2));
    }
}
```

Output:



```
Console ×
<terminated> LongestCommonSubs
Lenght of LCS :5
```