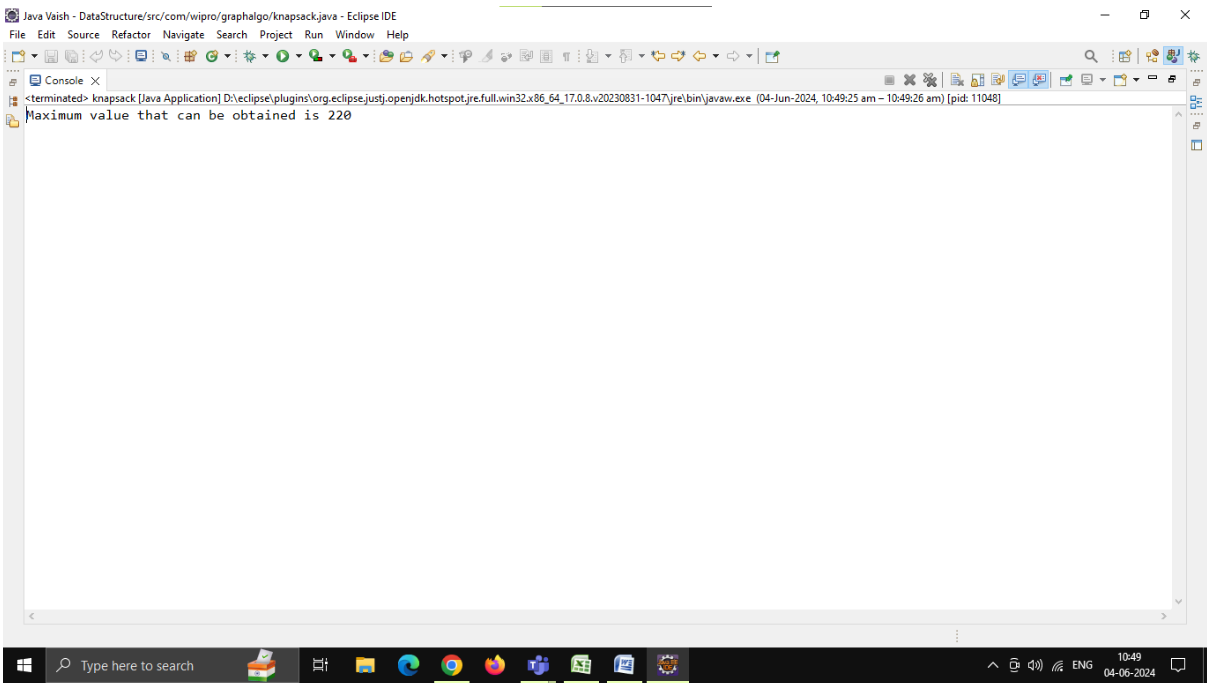
Day 15 and 16

Task 1: Knapsack Problem

Write a function int Knapsack(int W, int[] weights, int[] values) in C# 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.

Answer:

```
package com.wipro.graphalgo;

public class knapsack {

    static int knapsackk(int W, int[] weights, int[] values, int n)
{
        int[][] K = new int[n + 1][W + 1];

        for (int i = 0; i <= n; i++) {
            for (int w = 0; w <= W; w++) {
                if (i == 0 || w == 0)
                    K[i][w] = 0;
                else if (weights[i - 1] <= w)
                    K[i][w] = Math.max(values[i - 1] + K[i - 1][w -
weights[i - 1]], K[i - 1][w]);
                else
                    K[i][w] = K[i - 1][w];
            }
        }

        return K[n][W];
    }

    public static void main(String args[]) {
        int[] values = new int[]{60, 100, 120};
        int[] weights = new int[]{10, 20, 30};
        int W = 50;
        int n = values.length;
        System.out.println("Maximum value that can be obtained is "
+ knapsackk(W, weights, values, n));
    }
}
```


Java Vaish - DataStructure/src/com/wipro/graphalgo/knapsack.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Type here to search
ENG
10:49
04-06-2024

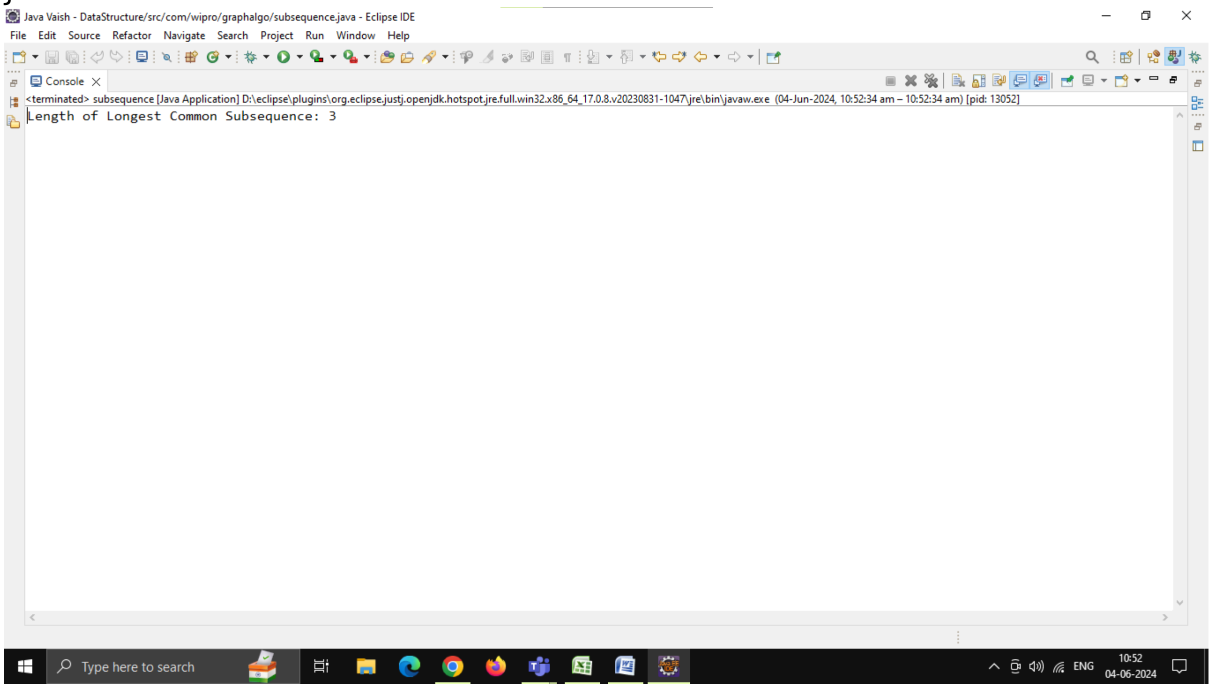
```
<terminated> knapsack [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (04-Jun-2024, 10:49:25 am – 10:49:26 am) [pid: 11048]
Maximum value that can be obtained is 220
```

Task 2: Longest Common Subsequence

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

Answer:

```java
package com.wipro.graphalgo;

public class subsequence {

    static int LCS(String text1, String text2) {
        int m = text1.length();
        int n = text2.length();
        int[][] dp = new int[m + 1][n + 1];

        for (int i = 0; i <= m; i++) {
            for (int j = 0; j <= n; j++) {
                if (i == 0 || j == 0)
                    dp[i][j] = 0;
                else if (text1.charAt(i - 1) == text2.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 text1 = "abcde";
        String text2 = "ace";
        System.out.println("Length of Longest Common Subsequence: "
+ LCS(text1, text2));
    }
}
```


Java Vaish - DataStructure/src/com/wipro/graphalgo/subsequence.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
Type here to search
ENG
10:52
04-06-2024

```
<terminated> subsequence [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe  (04-Jun-2024, 10:52:34 am – 10:52:34 am) [pid: 13052]
Length of Longest Common Subsequence: 3
```