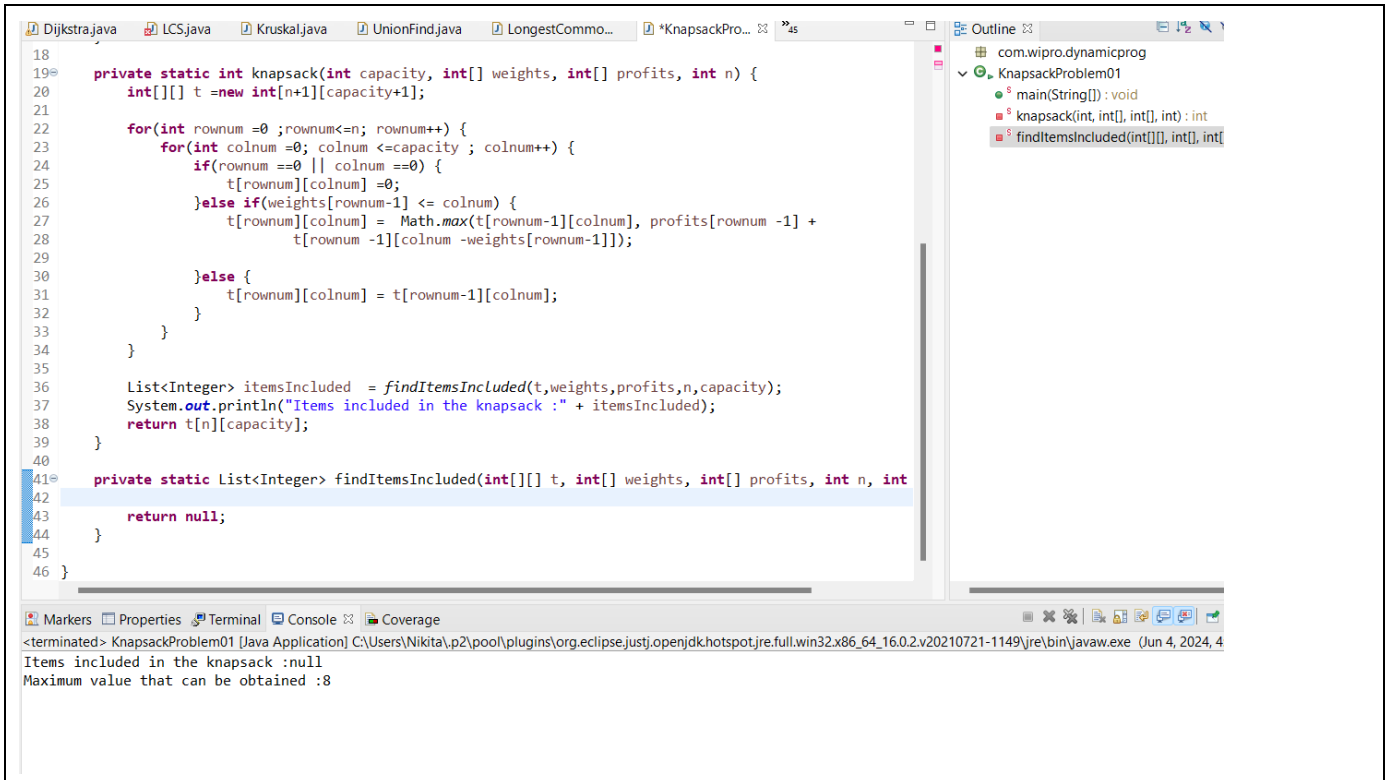
Knapsack Problem

```
import java.util.List;

public class KnapsackProblem01 {

	public static void main(String[] args) {

		int capacity =8;
		int[] values= {1,2,5,6};
		int[] weights = {2,3,4,5};
		int n = values.length;

		int maxValue= knapsack(capacity,weights,values, n);
		System.out.println("Maximum value that can be obtained :"+ maxValue);

	}

	private static int knapsack(int capacity, int[] weights, int[] profits, int n) {
		int[][] t =new int[n+1][capacity+1];

		for(int rownum =0 ;rownum<=n; rownum++) {
			for(int colnum =0; colnum <=capacity ; colnum++) {
				if(rownum ==0 || colnum ==0) {
					t[rownum][colnum] =0;
				}else if(weights[rownum-1] <= colnum) {
					t[rownum][colnum] =  Math.max(t[rownum-1][colnum],
profits[rownum -1] +
                                        t[rownum -1][colnum -weights[rownum-1]]);

				}else {
					t[rownum][colnum] = t[rownum-1][colnum];
				}
			}
		}

		List<Integer> itemsIncluded  =
findItemsIncluded(t,weights,profits,n,capacity);
		System.out.println("Items included in the knapsack :" + itemsIncluded);
		return t[n][capacity];
	}

	private static List<Integer> findItemsIncluded(int[][] t, int[] weights, int[]
profits, int n, int capacity) {

		return null;
	}


}
```




```
18
19      private static int knapsack(int capacity, int[] weights, int[] profits, int n) {
20          int[][] t =new int[n+1][capacity+1];
21
22          for(int rownum =0 ;rownum<=n; rownum++) {
23              for(int colnum =0; colnum <=capacity ; colnum++) {
24                  if(rownum ==0 || colnum ==0) {
25                      t[rownum][colnum] =0;
26                  }else if(weights[rownum-1] <= colnum) {
27                      t[rownum][colnum] =  Math.max(t[rownum-1][colnum], profits[rownum -1] +
28                              t[rownum -1][colnum -weights[rownum-1]]);
29
30                  }else {
31                      t[rownum][colnum] = t[rownum-1][colnum];
32                  }
33              }
34          }
35
36          List<Integer> itemsIncluded  = findItemsIncluded(t,weights,profits,n,capacity);
37          System.out.println("Items included in the knapsack :" + itemsIncluded);
38          return t[n][capacity];
39      }
40
41      private static List<Integer> findItemsIncluded(int[][] t, int[] weights, int[] profits, int n, int
42
43          return null;
44      }
45
46  }
```