Our Solution(s) Run Code

Solution 1

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

using System;
using System.Collections.Generic;

public class Program {
  // O(nc) time | O(nc) space
  public static List<List<int> > KnapsackProblem(int[,] items, int ca
    int[,] knapsackValues = new int[items.GetLength(0) + 1,capacity +
    for (int i = 1; i < items.GetLength(0) + 1; i++) {
      int currentWeight = items[i - 1,1];
      int currentValue = items[i - 1,0];
      for (int c = 0; c < capacity + 1; c++) {
        if (currentWeight > c) {
          knapsackValues[i,c] = knapsackValues[i - 1,c];
        } else {
          knapsackValues[i,c] = Math.Max(knapsackValues[i - 1,c],
              knapsackValues[i - 1,
              c -
              currentWeight] +
              currentValue);
        }
      }
    }
    return getKnapsackItems(knapsackValues, items,
            knapsackValues[items.GetLength(0),capacity]);
  }

  public static List<List<int> > getKnapsackItems(int[,] knapsackValue
    int weight) {
    List<List<int> > sequence = new List<List<int> >();
    List<int> totalWeight = new List<int>();
    totalWeight.Add(weight);
```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
using System.Collections.Generic;

public class Program {
  public static List<List<int> > KnapsackProblem(int[,] items, int cap
    // Write your code here.
    // Replace the code below.
    List<int> totalValue = new List<int> {
      10
    };
    List<int> finalItems = new List<int> {
      1, 2
    };
    var result = new List<List<int> >();
    result.Add(totalValue);
    result.Add(finalItems);
    return result;
  }
}

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.