Solution 1   Solution 2

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

using System;
public class Program {
  // O(nk) time | O(nk) space
  public static int MaxProfitWithKTransactions(int[] prices, int k) {
    if (prices.Length == 0) {
      return 0;
    }
    int[,] profits = new int[k + 1,prices.Length];
    for (int t = 1; t < k + 1; t++) {
      int maxThusFar = Int32.MinValue;
      for (int d = 1; d <prices.Length; d++) {
        maxThusFar =
          Math.Max(maxThusFar, profits[t - 1,d - 1] - prices[d - 1]);
        profits[t,d] = Math.Max(profits[t,d - 1], maxThusFar + prices[d]);
      }
    }
    return profits[k,prices.Length - 1];
  }
}
```

Solution 1   Solution 2

```csharp
using System;
public class Program {
  // O(nk) time | O(nk) space
  public static int MaxProfitWithKTransactions(int[] prices, int k) {
    if (prices.Length == 0) {
      return 0;
    }
    int[,] profits = new int[k + 1,prices.Length];
    for (int t = 1; t < k + 1; t++) {
      int maxThusFar = Int32.MinValue;
      for (int d = 1; d <prices.Length; d++) {
        maxThusFar =
          Math.Max(maxThusFar, profits[t - 1,d - 1] - prices[d - 1]);
        profits[t,d] = Math.Max(profits[t,d - 1], maxThusFar + prices[d]);
```