Ali Baba and the N Houses

Description

Ali Baba has scoped out a circle of N houses that you would like to rob tonight. However, he is realized that he wants to avoid robbing two houses that are adjacent to one another because this will minimize the chance that he gets caught. He has also scouted out the hood, so for every house i, he knows that the net worth he will gain from robbing that house is vi.

The problem is:
Given the number of houses (N) and a list of the net gained value for each consecutive house (V), find the following:

1. The maximum amount of gain that Ali baba can get.
2. The indices of the robbed houses (1-based and ordered from left to right)
    - if there're two or more combinations that have the same max total gain, return **ANY of them**

Trace Example: N = 5, V = [5, 2, 1, 3, 1],
Output:

1. Max gained value = 8
2. Robbed Houses = [1, 4]

Complexity

With an algorithm less than $O(N^2)$

Input: Already Implemented

The first line of input is an integer T (T = 10), that indicates the number of test cases. Each case consists of two lines: first one contains one integer represents the number of houses (N). Second line contains the N integers representing the values of each house separated by space.

Output: Already Implemented

The maximum amount of money the Ali Baba can get from the input N houses.

Function: Implement it!

```
long[] GetMaxGainedValue(long[] values, long N, ref long res)
```

It takes the array of integers (`values`), number of them (N). This function returns

1. The max gain that Ali Baba can get: as a reference param (i.e. set the "res" to it)
2. The indices of the robbed houses (1-based and ordered from left to right): as a return param (if there're multiple combinations that have the same max total gain, return ANY of them)

Template

- C# template

Test Cases

#	Input	Output
1	N = 5, values = [5 2 1 3 1]	8 [1,4]
2	N = 8, values = [8 3 5 1 7 6 5 3]	25 [1,3,5,7]
4	N = 6, values = [3 2 5 6 6 9]	18 [1,4,6]

C# Help

If you need any help regarding the syntax of C#, **ask any TA**.

Creating 1D array

```
int [] array = new int [size]
```

Creating 2D array

```
int [,] array = new int [size1, size2]
```

Sorting single array

Sort the given array "items" in ascending order

```
Array.Sort(items);
```

Sorting parallel arrays

Sort the first array "master" and re-order the 2nd array "slave" according to this sorting

```
Array.Sort(master, slave);
```