Coin-row Problem

In this lab you're going to be solving the coin-row problem with recursion which is stated as follows:

There is a row of n coins whose values are some positive integers c1, c2, . . . , cn, not necessarily distinct stored inside a list. The goal is to pick up the maximum amount of money subject to the constraint that no two coins adjacent in the initial row can be picked up.

For example if you have coins 5,1,2,10,6,2 you can get maximum value of 17 picking up coins 5, 10 and 2.

1) Write a function called **coin_row(L)** that takes in a list of coin values and returns maximum money that can be picked up from these coins.

2) Write a function called **coin_row_with_values(L)** that takes in a list of coin values and returns a list of maximum money that can be picked up from the list and what coins to actually select.

Here are some test cases and their expected outputs you can try:

```
print(coin_row([]))
print(coin_row_with_values([]))
print(coin_row([5, 1, 2, 10, 6, 2]))
print(coin_row_with_values([5, 1, 2, 10, 6, 2]))
print(coin_row([10, 5, 5, 5, 10, 50, 1, 10, 1, 1, 25]))
print(coin_row_with_values([10, 5, 5, 5, 10, 50, 1, 10, 1, 1, 25]))
```

```
0
[0, []]
17
[17, [5, 10, 2]]
100
[100, [10, 5, 50, 10, 25]]
```