Our Solution(s)   Run Code

Solution 1   **Solution 2**

```cpp
1  // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3  #include <vector>
4  #include <climits>
5  using namespace std;
6
7  // O(nk) time | O(n) space
8  int maxProfitWithKTransactions(vector<int> prices, int k) {
9    if (prices.size() == 0) {
10     return 0;
11   }
12   int *evenProfits = new int[prices.size()];
13   int *oddProfits = new int[prices.size()];
14   for (int i = 0; i < prices.size(); i++) {
15     evenProfits[i] = 0;
16     oddProfits[i] = 0;
17   }
18   for (int t = 1; t < k + 1; t++) {
19     int maxThusFar = INT_MIN;
20     int *currentProfits = new int[prices.size()];
21     int *previousProfits = new int[prices.size()];
22     if (t % 2 == 1) {
23       currentProfits = oddProfits;
24       previousProfits = evenProfits;
25     } else {
26       currentProfits = evenProfits;
27       previousProfits = oddProfits;
28     }
29     for (int d = 1; d < prices.size(); d++) {
30       maxThusFar = max(maxThusFar, previousProfits[d - 1] - prices[d -
31       currentProfits[d] = max(currentProfits[d - 1], maxThusFar + pric
32     }
33   }
```

Your Solutions   Run Code

Solution 1   **Solution 2**   **Solution 3**

```cpp
1  #include <vector>
2  using namespace std;
3
4  int maxProfitWithKTransactions(vector<int> prices, int k) {
5    // Write your code here.
6    return -1;
7  }
8
```

Our Tests

Custom Output   Submit Code