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

package main

import "math"

// # O(nk) time | O(n) space
func MaxProfitWithKTransactions(prices []int, k int) int {
  if len(prices) == 0 {
    return 0
  }
  evenProfits := make([]int, len(prices))
  oddProfits := make([]int, len(prices))
  var currentProfits, previousProfits []int
  for t := 1; t < k+1; t++ {
    maxThusFar := math.MinInt32
    if t%2 == 1 {
      currentProfits, previousProfits = oddProfits, evenProfits
    } else {
      currentProfits, previousProfits = evenProfits, oddProfits
    }
    for d := 1; d < len(prices); d++ {
      maxThusFar = max(maxThusFar, previousProfits[d-1]-prices[d-1])
      currentProfits[d] = max(currentProfits[d-1], maxThusFar+prices[d])
    }
  }
  if k%2 == 0 {
    return evenProfits[len(prices)-1]
  }
  return oddProfits[len(prices)-1]
}

func max(arg int, rest ...int) int {
  curr := arg
  for _, num := range rest {
    if curr < num {
      curr = num
    }
  }
  return curr
}

```