class Solution(object):

def maxProfit(self, k, prices):

"""

:type k: int

:type prices: List[int]

:rtype: int

"""

K>>N/2, then there must be a way to merge two buy/sell into one.

So K>>N/2 can be converted to K==N/2

if k>len(prices)//2:

prof = 0

for i in range(1,len(prices)):

if prices[i]>prices[i-1]:

prof+=prices[i]-prices[i-1]

return prof

f = [[0 for i in range(2\*k+1)] for i in range(2)]

f[0][0] = 0

for i in range(1,2\*k+1):

f[0][i] = float("-inf")

for i in range(1,len(prices)+1):

for j in range(1,2\*k+1):

cost1=cost2=0

if j==0:

pass

elif not j%2:

cost1 = f[0][j]

cost2 = f[0][j-1]+prices[i-1]

if i-2>=0:

cost2-=prices[i-2]

else:

cost1 = f[0][j-1]

cost2 = f[0][j]+prices[i-1]

if i-2>=0:

cost2-=prices[i-2]

f[1][j] = max(cost1,cost2)

for s in range(2\*k+1):

f[0][s] = f[1][s]

return max(f[-1])