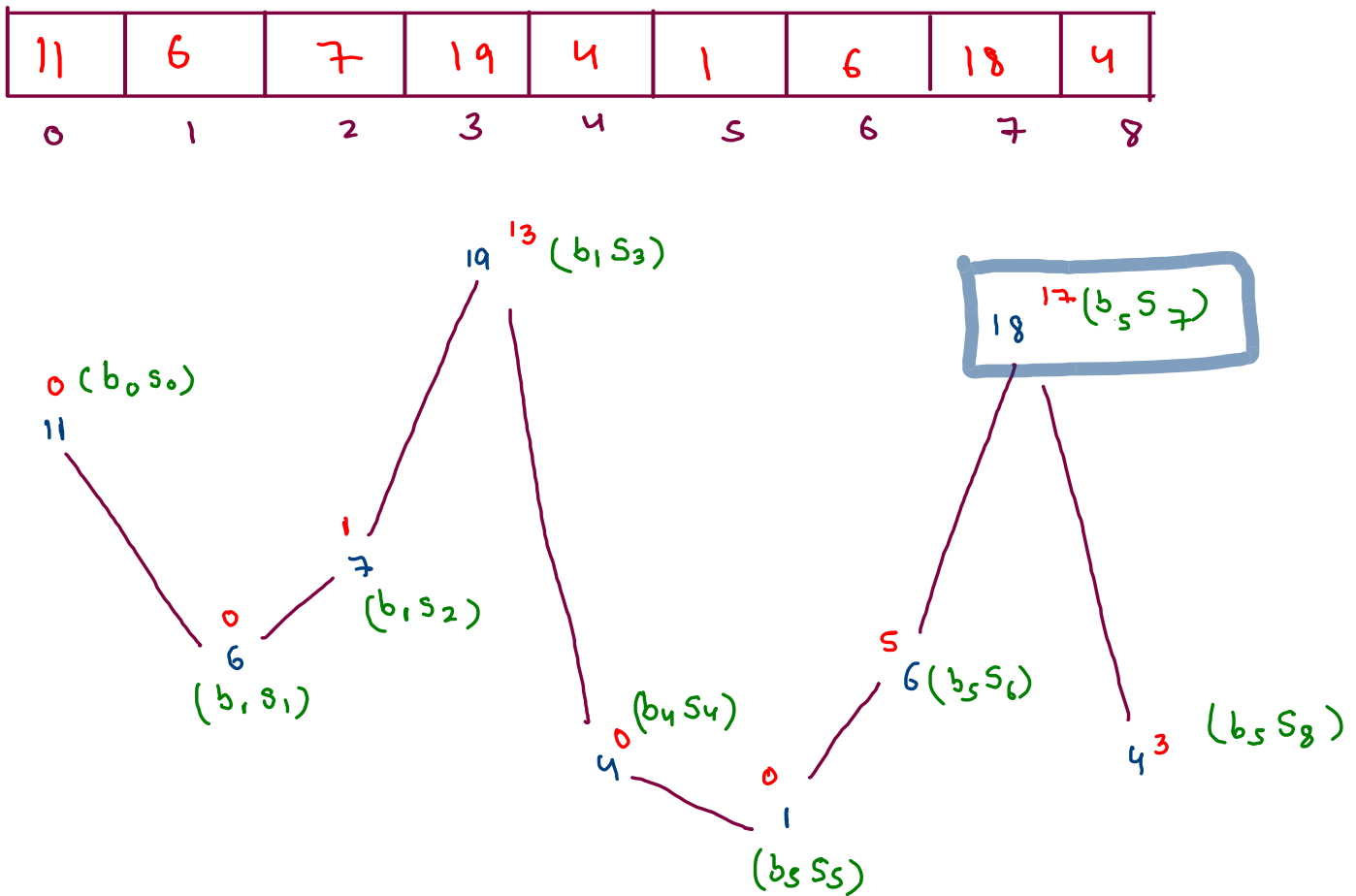
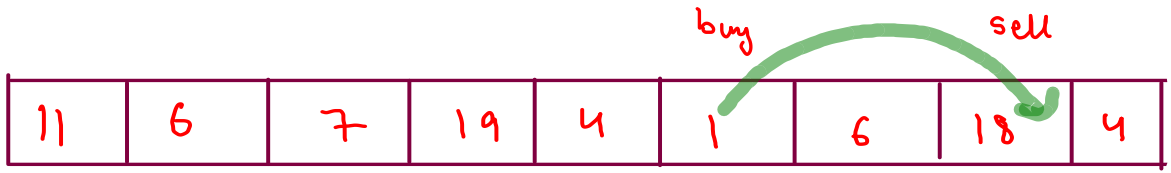
Buy And Sell Stocks - One Transaction Allowed





pr 0 0 1 13 0 0 5 17 3

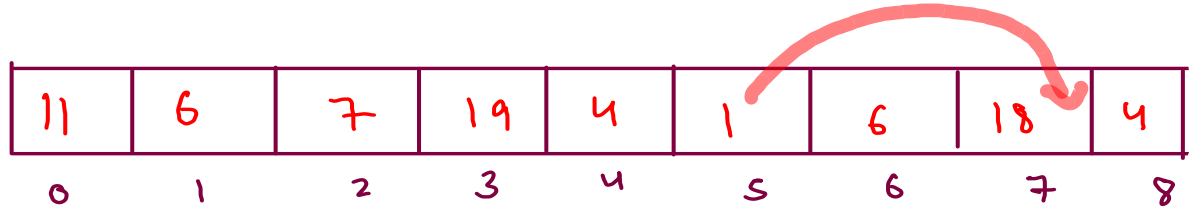
msf (min so far)

opr (overall best profit)

msf = ~~∞~~ ~~11~~ ~~6~~ ~~4~~ 1

opr = ~~0~~ ~~1~~ ~~13~~ 17

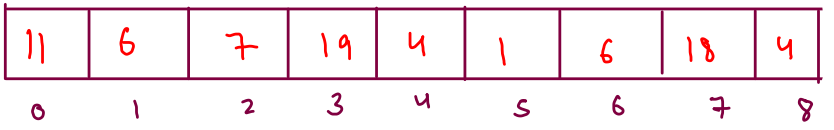
```java
public static int one_transaction(int[]prices) {

    int lmsf = Integer.MAX_VALUE; //left min so far
    int opr = 0; //overall best profit

    for(int i=0; i < prices.length;i++) {
        if(prices[i] < lmsf) {
            lmsf = prices[i];
        }

        int pr  = prices[i] - lmsf; //ith day is selling day

        if(pr > opr) {
            opr = pr;
        }
    }

    return opr;
}
```

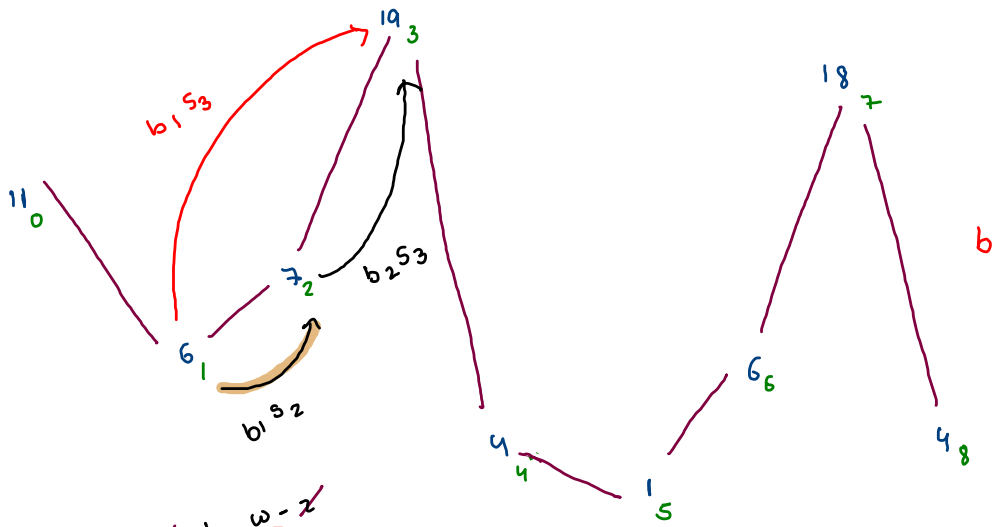


lmsf = 0 0 1 13 0 0 5 17 3

opr = 0 17

Buy And Sell Stocks - Infinite Transactions Allowed

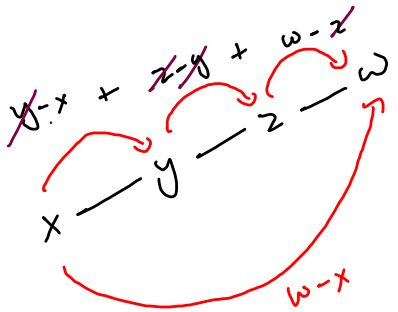




(i) overlapping transaction are not allowed.

$b_i s_k + b_k s_j \rightarrow b_i s_j$

$= -p[i] + p[k] - p[k] + p[j]$

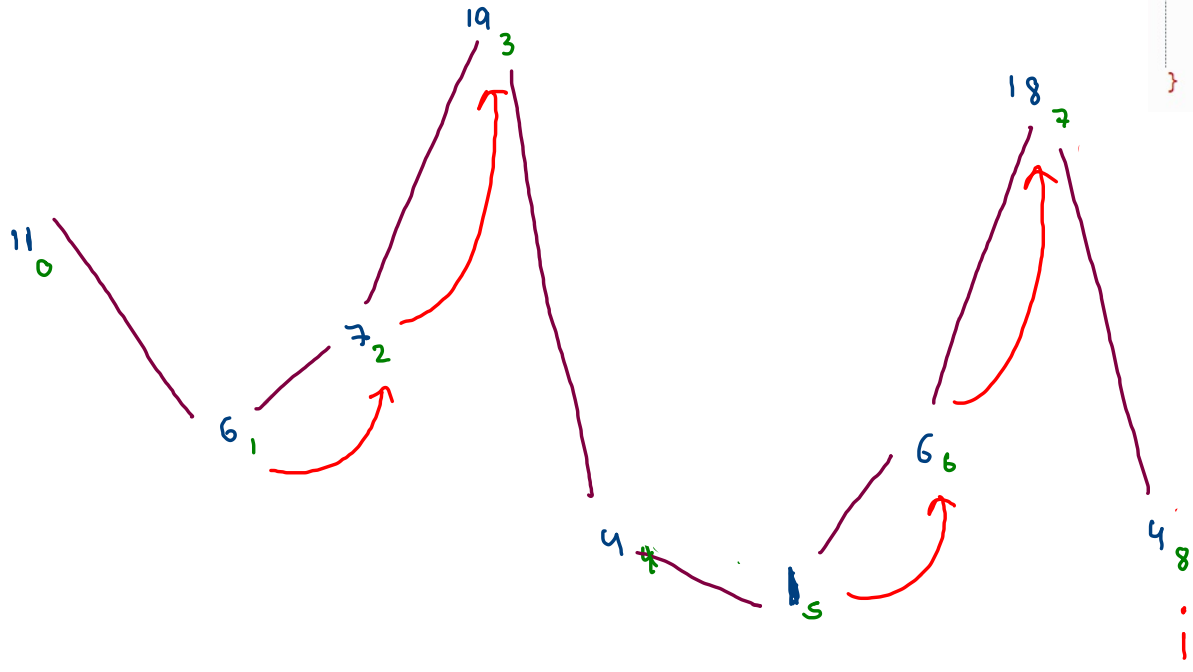


$b_1 s_3 \rightarrow -6 + 19$

$b_1 s_2 + b_2 s_3 \rightarrow -6 + 7 - 7 + 19$

$b_1 s_3 = b_1 s_2 + b_2 s_3$

```java
public static int infinite_transaction(int[]prices) {
    int acc = 0; //overall profit accumulator

    for(int i=0; i < prices.length - 1;i++) {
        if(prices[i] < prices[i+1]) {
            acc += (prices[i+1] - prices[i]);
        }
    }

    return acc;
}
```

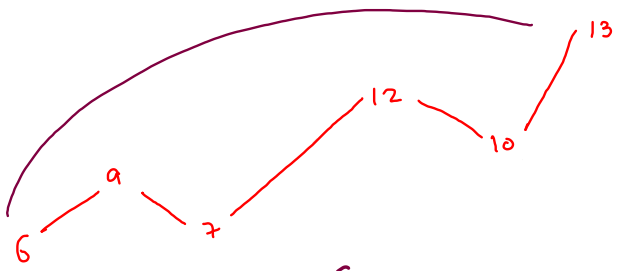


acc = 1 + 12 + 5

12

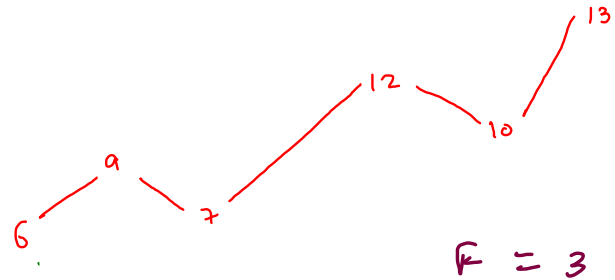
acc = 30

Buy And Sell Stocks With Transaction Fee - Infinite Transactions Allowed



F = 2

	bs	ss
6_0	-6 b_0	0 .
9_1	-6 b_0	1 $b_0 s_1$
7_2	-6 b_0	1 $b_0 s_1$
12_3	-6 b_0	4 $b_0 s_3$
10_4	-6 b_0	4 $b_0 s_3$
13_5	-6 b_0	5 $b_0 s_5$



F = 3

	bs	ss
6_0	-6 b_0	0 .
9_1	-6 b_0	0 .
7_2	-6 b_0	0 .
12_3	-6 b_0	3 $b_0 s_3$
10_4	-6 b_0	3 $b_0 s_3$
13_5	-6 b_0	4 $b_0 s_5$

12 10 15 17 20
16 18 22 20 22
20 23 25 3

	buy	state	sell	state
10_0	-10	b_0	0	
15_1	-10	b_0	2	$b_0 s_1$
17_2	-10	b_0	4	$b_0 s_2$
20_3	-10	b_0	7	$b_0 s_3$
16_4	-9	$b_0 s_3 b_4$	7	$b_0 s_3$
18_5	-9	$b_0 s_3 b_4$	7	$b_0 s_3$
22_6	-9	$b_0 s_3 b_4$	10	$b_0 s_3 b_4 s_6$
20_7	-9	$b_0 s_3 b_4$	10	$b_0 s_3 b_4 s_6$
22_8	-9	$b_0 s_3 b_4$	10	$b_0 s_3 b_4 s_6$
20_9	-9	$b_0 s_3 b_4$	10	$b_0 s_3 b_4 s_6$
23_{10}	-9	$b_0 s_3 b_4$	11	$b_0 s_3 b_4 s_{10}$
25_{11}	-9	$b_0 s_3 b_4$	13	$b_0 s_3 b_4 s_{11}$

fees = 3

buy state :

-> one extra 'b'
-> end at b

sell state :

-> no. of b = no. of s.

obsp, ossp

```
i  nbsp = max(obsp, ossp - p[i]);
   nssp = max(ossp, obsp + p[i] - fee);
```

```java
public static int infinite_transactions_with_fee(int[]prices,int fee) {
    int obsp = -prices[0]; //old buy state profit
    int ossp = 0; //old sell state profit

    for(int i=1; i < prices.length;i++) {
        int nbsp = Math.max(obsp,ossp - prices[i]); //new buy state profit
        int nssp = Math.max(ossp, obsp + prices[i] - fee); //new sell state profit

        obsp = nbsp;
        ossp = nssp;
    }

    return ossp;
}
```

fees = 3

days	buy state	sell state
15_0	-15	0
10_1	-10	0
17_2	-10	4
20_3	-10	7
16_4	-9	7
18_5	-9	7
22_6	-9	10
20_7	-9	10
23_8	-9	11
25_9	-9	13

Buy And Sell Stocks With Cooldown - Infinite Transaction Allowed

12 10 15 17 20
16 18 22 20 22
20 23 25

	buy state	sell state	cooldown
10_0	$-10\ b_0$	0	0
15_1	$-10\ b_0$	$5\ b_0s_1$	0
17_2	$-10\ b_0$	$7\ b_0s_2$	$5\ b_0s_1c_2$
20_3	$-10\ b_0$	$10\ b_0s_3$	$7\ b_0s_2c_3$
16_4	$-9\ b_0s_2c_3b_4$	$10\ b_0s_3$	$10\ b_0s_3c_4$
18_5	$-8\ b_0s_3c_4b_5$	$10\ b_0s_3$	$10\ b_0s_3c_4$
22_6	$-8\ b_0s_3c_4b_5$	$14\ b_0s_3c_4b_5s_6$	$10\ b_0s_3c_4$
20_7	$-8\ b_0s_3c_4b_5$	$14\ b_0s_3c_4b_5s_6$	$14\ b_0s_3c_4b_5s_6c_7$
22_8	$-8\ b_0s_3c_4b_5$	$14\ b_0s_3c_4b_5s_6$	$14\ b_0s_3c_4b_5s_6c_7$
20_9	$-6\ b_0s_3c_4b_5s_6c_7b_9$	$14\ b_0s_3c_4b_5s_6$	$14\ b_0s_3c_4b_5s_6c_7$
23_{10}	$-6\ b_0s_3c_4b_5s_6c_7b_9$	$17\ b_0s_3c_4b_5s_6c_7b_9s_{10}$	$14\ b_0s_3c_4b_5s_6c_7$

→ buy state
extra b

→ sell state
no. of s = no. of b

→ cool state
extra c

ob, os, oc

```
nb = max(ob, oc - p[i]);
ns = max(os, ob + p[i]);
nc = max(oc, os to);
```