1. **specifications**

Given the price of a stock over an n-day period, determine the best time to have bought and sold 1000 shares of that stock. (Buy and sell once, on different days.)

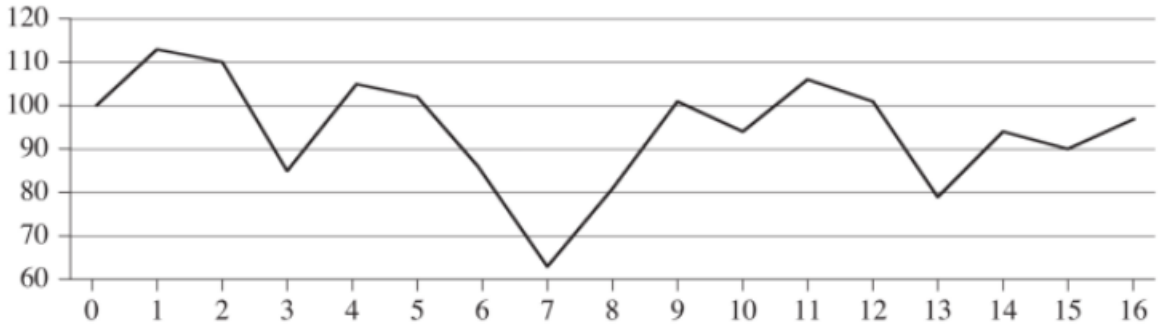
input: n prices on various days

output: the buy/sell dates (one pair) that maximizes profit

legal output: buy date before sell date

optimization goal: buy/sell to maximize profit

2. **examples**



Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Price	100	113	110	85	105	102	86	63	81	101	94	106	101	79	94	90	97

buy on day 7, sell on day 11 → profit 106-73 = 33 per share

3. **size**

number of days

smaller problem – fewer days

smallest problem – 2 days

4. **targets**

brute force: for every legal pair of buy/sell days (buy < sell), find the profit and pick the max - Th(n^2)

5. **tactics**

6. **approaches**

process input: split the n days into first half, second half; friends will find buy/sell dates in each half that max profit; we have to find the overall buy/sell for the whole thing

produce output: n/a

narrow the search space: n/a

7. **generalize / define subproblems**

original problem: find buy/sell dates to max profit in the whole n days: maxprofit(A) is buy/sell dates to max profit in A

generalized subproblem: find buy/sell dates to max profit in a range of days low..high: maxprofit(A,low,high) is the buy/sell dates to max profit in A[low..high]

8. **base case(s)**

size 2: buy on day low, sell on day high

size 3: days low, mid, high: (low,mid), (low,high), (mid,high) – pick the best

9. **main case**

10. **top level**
    a) **initial subproblem**
    b) **setup**
    c) **wrapup**

11. **special cases**

12. **algorithm**

13. **termination**
    a) **making progress**
    b) reaching the end

14. **correctness**
    a) **establish the base case(s)**
    b) **show the main case**
    c) **final answer**

15.implementation

16.time and space