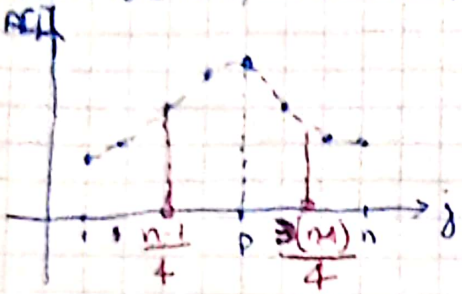
PAP - PARADIGMA DIV E CONQ. — Ex. Res. 1 e 2 (Kleinberg - Ch 5) + 1, 2, 6 (Grim Ch)

SOLVED EXERCISE 1)

A = [A[1], A[2], ..., A[n]] → unimodal



Se eu souber o tamanho da entrada, e ela for discreta, eu posso selecionar um valor [meio] e avaliar $A[\frac{n-1}{2}]$. Depois, eu avalio $A[3\frac{(n-1)}{4}]$ e $A[\frac{n-1}{4}]$, verificando qual é menor. Se percebo que, a partir de $\frac{n-1}{2}$ estou crescendo, p está na segunda metade da função. Se for o contrário, p está na primeira.

Posso realizar esta operação iterativamente, sempre na direção de crescimento da função. Este método é similar ao método de Newton, e garante que fazemos $\log(n)$ iterações até encontrar

SOLVED EXERCISE 2) I have a question! It's different from

Days → $i = 1, 2, 3 \ldots n$
Price $p(i)$ per share on a specific day
Q: when to buy / sell → maximizing $

Ex: n: 3, p(1): 9, p(2): 1, p(3): 5
output: buy on 2, sell on 3

Solução: Analisar o 1º dia e armazenar o valor em uma variável, como best day to buy. Se eu sei o número de dias, eu analiso n/2 (o dia específico

MENTIRA! Eu posso usar uma abordagem similar ao quicksort, em $O(n)$:

Início no dia 1, armazeno seu valor na var. best day to buy. no dia 2, se o valor for menor que no dia 1, eu atualizo best day to buy; senão, eu coloco o valor na best day to sell. Avanço para o próximo dia. Se o valor for menor que best day to buy, atualizo ele e zero best day to sell. se o valor for maior que best day to sell, mantenho best day to buy e atualizo best day to sell.

Contudo, se eu pegar a diferença entre best day to buy e best day to sell, e ela for consideravelmente maior que best day to buy (valor), mesmo que eu encontre um dia muito melhor para comprar eu não compro pois não há garantia que haverá um dia no futuro com uma diferença maior