Raul Alexandre Gonzalez Augusto RA: 211023698

# 1

Em casos de n <= 5

# 2

a(n) = n² - n + 549

b(n) = 49n + 49

# 3

n, sqrt(n), log(n), log(log(n)), log(n)², n/log(n), sqrt(n)log(n)², (1/3)n, (3/2)n, 17.

Ordenado: 17, log(log(n)), log(n), (1/3)n, n, (3/2)n, n/log(n), log(n)², sqrt(n)log(n)², sqrt(n)

# 4

1. Complexidade: O(N).
2. Não, para ser O(n²) teria de ter um for dentro de outro for.

# 5

Temos de pensar como o algoritmo se comportaria no pior caso para tentar otimiza-lo.

# 6

direita = true

x = 1

porta = false

enquanto nao porta:

    se direita:

        dar x passo a direita

        verificar a posicao do muro

    se nao direita:

        dar x passo a esquerda

        verificar a posicao do muro

    x++

    direita = nao direita

entrar na porta

# 7

O(N³)

# 8

1. 150n\*log(n)
2. n²
3. 150n\*log(n)
4. Não.

# 9

Hanoi: 2^n – 1

Fibonacci: O(n)

Ackerman: O(i) complexidade espacial e O(iA(i,n)) complexidade de tempo

# 10

n/2

# 11

1. f(n) = n2 + 2

O(N)

b) g(n) = 503

O(1)

1. g(n) = 2 log n + n

O(log(N))

1. g(n) = 10.2n

O(N)

e) f(n) = n log n + log n2.

O(Nlog(N))

# 12

4n^2, n!, log3n, 3^n, 20n, 2, log2n.

Ordenado: 2, log2n, log3n, 20n, 4n^2, 3^n, n!