Atividade Avaliativa 1

Cauã Borges Faria(834437)

(1)

```
funcao (n) {
    a = 999
    j = n
    K = 0
    while j > 0 {
        while K < j {
            a = a - 3
            K = K + 1
        }
        j = j - 1
        K = 0
    }
    return a
}
```

j começa em n e vai até 1 (decrementando n interações)

→ executa j vezes

∴

nº de atualizações de $a = a - 3$ é

$$T(n) = \sum_{j=1}^{n} j = \frac{n(n+1)}{2}$$

$$T(n) = O(n^2)$$

(2)

```
FUNC(n) {
    for i=1 to n {                — n vezes
        func1(i)                  — O(1) * n = O(n)
        for j=1 to n {            — n vezes
            func2(j)              — O(n) * n = O(n²)
            for k=1 to n          — n vezes
                func3(k)          — O(n²) * n = O(n³)
        }
    }
}
```

$func1() = O(1)$

$func2() = O(n)$

$func3() = O(n^2)$

Sequencialmente o código é executado, o termo de maior ordem é $O(n^3)$ que é executado "n" vezes no for(j) e mais "n" vezes no for(i), dessa forma complexidade do algoritmo é $O(n^5)$.

③

a) https://onlinegdb.com/oL3CkP36M

b)

```
def shellsort_shell(arr):
    n = len(arr)
    gap = n//2
    while gap > 0:
        for i in range(gap, n):
            temp = arr[i]
            j = i
            while j >= gap and arr[j-gap] > temp:
                arr[j] = arr[j-gap]
                j -= gap
            arr[j] = temp
        gap //= 2
```

$$T(n) = \left(n - \frac{n}{2}\right)1 + \left(n - \frac{n}{4}\right)2 + \left(n - \frac{n}{8}\right)4 + \dots + \left(n - \frac{n}{2^k}\right)2^{k-1}$$

$$T(n) = \frac{n}{2}\left(1 + 3 + 7 + 15 + \dots + (2^k - 1)\right) \quad 2(n-1) - \log_2 n$$

$$T(n) = \frac{n}{2}\left[2(n-1) - \log_2 n\right] = n^2 - n - \frac{1}{2}\log_2 n$$

$$T(n) = O(n^2)$$

4

a) https://onlinegdb.com/kaUCknieZ

b)

```
def porticion(L, low, high):
    x = L[High]
    i = low-1
    for j in range(low, high):
        if L[j] <= x:
            i += 1
            L[i], L[j] = L[j], L[i]
    L[i+1], L[high] = L[high], L[i+1]
    return i+1

def quicksort(L, low, high):
    if low < high:
        q = porticion(L, low, high)
        quicksort(L, low, q-1)
        quicksort(L, q+1, high)
```

complexidade:

$(1, n-1)$; $(2, n-2)$

$(3, n-3) \dots (n-1, 1)$

$$T(n) = \frac{1}{n}\sum_{i=1}^{n} T(i-1) + \frac{1}{n}\sum_{i=1}^{n} T(n-i) + cn$$

$$T(n) = \frac{2}{n}\sum_{i=1}^{n} T(i-1) + cn = \frac{2}{n}\sum_{i=0}^{n-1} T(i) + cn$$

$$nT(n) = 2\sum_{i=0}^{n-1} T(i) + cn^2 \quad (i) \quad n = n-1$$

$$(n-1)\,T(n-1) = 2\sum_{i=0}^{n-2} T(i) + c(n-1)^2 \quad (ii)$$

$i - ii$

$$nT(n) - (n-1)\,T(n-1) = 2T(n-1) + 2cn$$

$$nT(n) = 2T(n-1) + nT(n-1) - T(n-1) + 2cn$$

$$nT(n) = (n+1)\,T(n-1) + 2cn$$

$$\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{2c}{n+1}$$

... expansão $T(n-1) \dots T(n-2)$

$$\frac{T(n)}{n+1} = \frac{T(1)}{2} + \frac{2c}{3} + \frac{2c}{4} + \frac{2c}{5} + \dots + \frac{2c}{n+1}$$

$T(1) = O(1)$

$$\frac{T(n)}{n+1} = O(1) + 2c\sum_{i=3}^{n+1} \frac{1}{i}$$

$$\log_e a = \ln a = \int_1^a \frac{1}{x}\,dx \quad \therefore$$

$$T(n) = (n+1)(O(1) + 2c\,O(\log_e n)) =$$

$$(n+1) + 2cn\,O(\log_e n) + 2c\,O(\log_e n) \quad \therefore$$

$$O(n \log_e n)$$

5

a) https://onlinegdb.com/YK6j82l68

b)

$$T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$

expandindo

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

$$T\left(\frac{n}{2}\right) = 2T\left(\frac{n}{4}\right) + n$$

...

$$T(n) = 2\left[2\left[2\left[2T\left(\frac{n}{16}\right) + \frac{n}{8}\right] + \frac{n}{4}\right] \frac{n}{2}\right] + n$$

$$= 2^4 T\left(\frac{n}{2^4}\right) + 4n$$

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + kn$$

p/ ter $T(1)$ $n = 2^k$, ou seja, $k = \log_2 n$

$$T(n) = 2^{\log_2 n} T(1) + n \log_2 n = nT(1) + n \log_2 n$$

$\therefore$

$$O(n \log_2 n)$$