CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DO RN

GERÊNCIA DE TECNOLOGIA DA INFORMAÇÃO E EDUCACIONAL DE TELEMÁTICA

Exercícios de Estrutura de Dados I

1. O algoritmo A usa $100n$ operações enquanto o algoritmo B usa $3n^2$ operações. Determine n_0 para o qual A é melhor do que B para $n \geq n_0$.

2. Considere dois algoritmos A e B com complexidades $8n^2$ e n^3. Qual o maior valor de n, para o qual o algoritmo B é mais eficiente que o algoritmo A?

3. Um algoritmo leva 2ms para terminar com uma entrada de tamanho 100. Qual o tamanho da entrada para que o algoritmo termine em 10, 30 e 60 segundos quando a ordem de complexidade for:

(a) O(n)
(b) O(n^2)
(c) O(n^3)
(d) O($\log n$)
(e) O($n \log n$)

4. Um algoritmo leva 0.5ms para terminar com uma entrada de tamanho 100. Em quanto tempo o algoritmo termina quando a entrada for de tamanho 10.000 e a ordem de complexidade for:

(a) O(n)
(b) O(n^2)
(c) O(n^3)
(d) O($\log n$)
(e) O($n \log n$)

5. Ordene as funções a seguir pela ordem de complexidade: n^2, n, $\log n$, 2^n, $n \log n$, n^3

6. Responda Verdadeiro ou Falso.

$n \in O(n^2)$

$n \in O(1)$

$\log n \in O(2^n)$

$\log n \in O(n)$

$n \log n \in O(n)$

$n \log n \in \Omega(n)$

$n \log n \in O(n^2)$

$n \log n \in \Omega(n^2)$

$2 \log n \in \Theta(\log n)$

$n^3 + n \log n + n\sqrt{n} \in O(n \log n)$

$n^3 + n \log n + n\sqrt{n} \in O(n^3)$

$n^3 + n \log n + n\sqrt{n} \in O(n^4)$

7. Faça

(a) Considere A um arranjo (array) ordenado de n número inteiros. Desenvolva um algoritmo que receba A e um número inteiro n e retorne a posição do arranjo em que n se encontra

(b) Qual o número de operações do seu algoritmo no pior caso? E no melhor?

(c) Qual a ordem de complexidade do algoritmo? Este algoritmo é bom?

8. Mostre que 2^{n+1} é O(2^n)

9. O que faz o seguinte algoritmo? Analise seu tempo de execução do pior caso e expresse seu valor usando a notação “big-Oh”.

```
Algoritmo A(a,n):
        Entrada dois inteiros, a e n
        Saída: ?
 k ← 0
 b ← 1
 enquanto (k < n) faça
    k ← k +1
    b ← b * a
 retorne b
```

10. O que faz o seguinte algoritmo? Analise seu tempo de execução do pior caso e expresse seu valor usando a notação “big-Oh”.

```
Algoritmo A(a,n):
        Entrada dois inteiros, a e n
        Saída: ?
 k ← n
 b ← 1
 c ← a
 enquanto (k > 0) faça
   se (k % 2 == 0) então
    k ← k / 2
    c ← c * c
   else
    k ← k -1
    b ← b * c
 retorne b
```

11. O que faz o seguinte algoritmo? Analise seu tempo de execução do pior caso e expresse seu valor usando a notação “big-Oh”.

```
Algoritmo C(A, t ,x):
        Entrada: Um array A (ordenado) e dois inteiros t e z
        Saída: ?
 l ← 0
 r ← t -1
 ac ← 0
 enquanto ((l ≤ r) ^ (ac == 0)) faça
   k ← (l + r) / 2
   se (x == A[k]) então
      ac ← 1
   senão se  (x < A[k]) então
      r ← k - 1
   senão
      l ← k + 1
 se (ac == 1)
     return k
 senão
     return -1
```

12. Seja A um array de inteiros (positivos e negativos), com tamanho n. Calcular a maior subseqüência de A. A maior subseqüência é a seqüência cuja soma possui o maior valor.

Por exemplo, considere A = {–2, 11,–4, 13,–5, 2}, a seqüência que possui o maior valor é a que corresponde a do segundo (A[1]) ao quarto (A[3]) elemento {11,–4, 13}, cuja soma é 20. Para a seqüência A = {1,–3, 4,–2,–1, 6} a maior é a que corresponde do terceiro (A[2]) ao último elemento (A[5]), cuja soma é 7. Se todos os elementos forem negativos, a maior subseqüência possui zero elementos e sua soma é 0.
Desenvolva um algoritmo que calcule a maior subseqüência