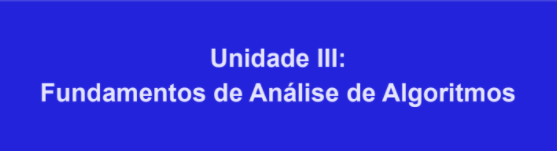
Trabalho Teórico 8



Slide A.

**Exercícios resolvidos:**

1)Resolva as equações abaixo:

a)

b)=10

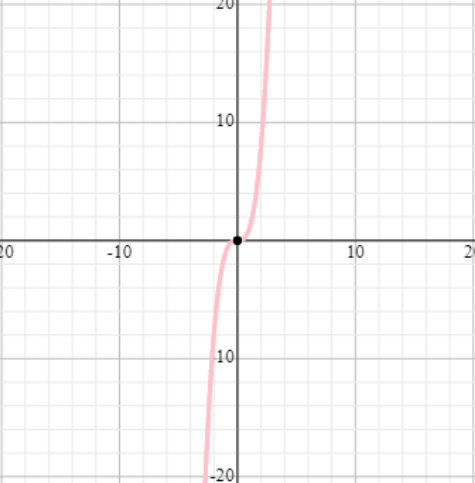
c)

d)

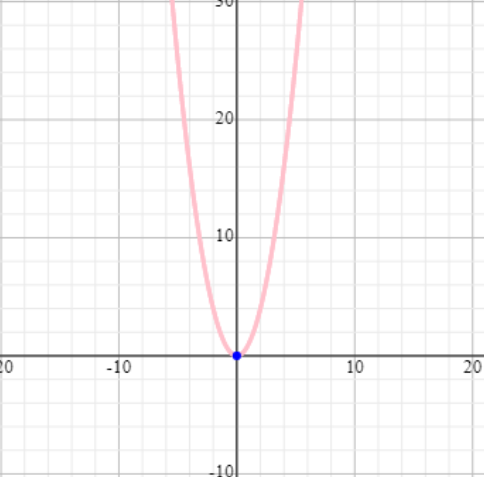
e) 4

2) Plote um gráfico com todas as funções abaixo:

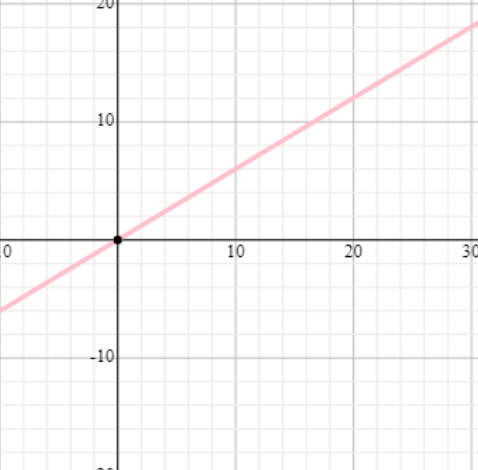
a) f(x) = n3



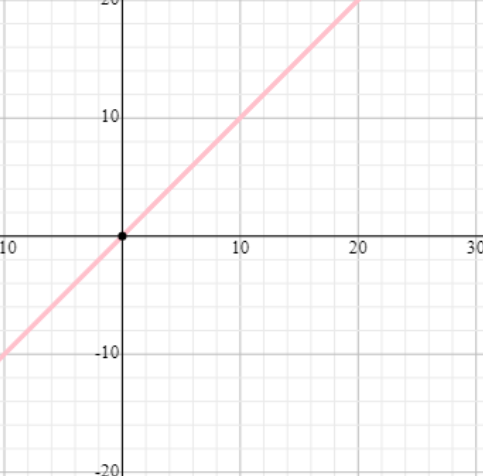
b)f(x) = n²



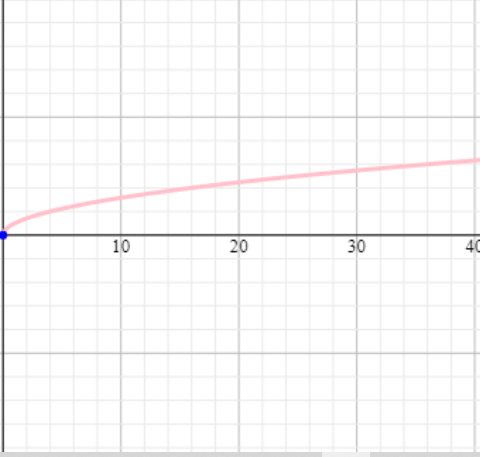
c)f(x)=n x lg(n)



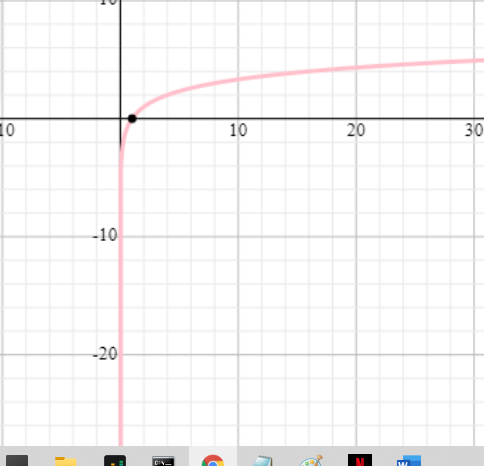
d) f(n) = n



e) f(n) = sqrt(n)



f) f(n) = lg(n)



3) Calcule o número de subtrações que o código abaixo realiza:

for (*int* i = 0; i < n; i++){

if (i % 2 == 0){

a--;

b--;

} else {

c--;

}

}

Melhor caso: f(n) = n logo O(n), Ω(n) e Θ(n)

Pior caso: f(n) = 2n logo O(n), Ω(n) e Θ(n)

4) Calcule o número de subtrações que o código abaixo realiza:

for (*int* *i* = *3*; *i* < n; i++){

a--;

}

f(n) = n-3 logo O(n), Ω(n) e Θ(n)

5) Calcule o número de multiplicações que o código abaixo realiza:

for (*int* *i* = *n*; *i* > *0*; *i* /= *2*)

a \*= 2;

f(n) = lg(n)+1 logo O(lg n), Ω(lg n) e Θ(lg n)

7 e 8) Encontre o menor valor em um array de inteiros:

*int* min = array[0];

for (*int* *i* = *1*; *i* < n; i++){

            if (min > array[i]){

            min = array[i];

            }

        }

1º) Qual é a operação relevante? 🡪 min>array[i]

2º) Quantas vezes ela será executada? 🡪 n-1

3º) O nosso T(n) = n – 1 é para qual dos três casos? 🡪 Para os três casos

4º) O nosso algoritmo é ótimo? Por que? 🡪Sim pois ele testa todos os casos antes da resposta.

9) Analise o código abaixo:

*boolean* resp = false;

        for (*int* *i* = *0*; *i* < n; i++){

            if (array[i] == x){

            resp = true;

            i = n;

            }

        }

1º) Qual é a operação relevante? 🡪 Comparação entre elementos do array

2o) Quantas vezes ela será executada? 🡪 Melhor caso: f(n) = 1

Pior caso: f(n) = n

Caso médio: f(n) = (n + 1) / 2

3o) O nosso algoritmo é ótimo? Por que? 🡪 Sim, porque temos que testar

todos os elementos para garantir nossa resposta.

10) Um aluno deve procurar um valor em um array de números reais. Ele tem duas alternativas. Primeiro, executar uma pesquisa sequencial. Segundo, ordenar o array e, em seguida, aplicar uma pesquisa binária. O que fazer?

O aluno deve escolher a primeira opção, pois a pesquisa sequencial tem custo Θ(n). A segunda opção tem custo Θ (n \* lg n) para ordenar mais Θ(lg n) para a pesquisa binária.

11) Responda se as afirmações são verdadeiras ou falsas:

a) 3n2+ 5n + 1 é O(n): Falso

b) 3n2+ 5n + 1 é O(n2): Verdadeiro

c) 3n2+ 5n + 1 é O(n3): Verdadeiro

d) 3n2+ 5n + 1 é Ω(n): Verdadeiro

e) 3n2+ 5n + 1 é Ω(n2): Verdadeiro

f) 3n2+ 5n + 1 é Ω(n3): Falso

g) 3n2+ 5n + 1 é Θ(n): Falso

h) 3n2+ 5n + 1 é Θ(n2): Verdadeiro

i) 3n2+ 5n + 1 é Θ(n3): Falso

12) Apresente a função e a complexidade para os números de comparações e movimentações de registros para o pior e melhor caso:

*void* imprimirMaxMin( *int* [] *array*, *int* *n*){

*int* maximo, minimo;

        if (array[0] > array[1]){

            maximo = array[0]; minimo = array[1];

        } else {

            maximo = array[1]; minimo = array[0];

        }

        for (*int* i = 2; i < n; i++){

            if (array[i] > maximo){

                    maximo = array[i];

                } else if (array[i] < minimo){

                    minimo = array[i];

                }

            }

        }

}

Movimentações:

PIOR f(n) = 2 + (n – 2) || O(n), Ω(n) e Ɵ(n)

MELHOR f(n) = 2 || O(1), Ω(1) e Ɵ(1)

Comparações:

PIOR: f(n) = 1 + 2(n – 2) || O(n), Ω(n) e Ɵ(n)

MELHOR: f(n) = 1 + (n – 2) || O(n), Ω(n) e Ɵ(n)

13) Apresente a função e a complexidade para o número de subtrações para opior e melhor caso.

while (*i* < n) {

    i++;

    a--;

}

if (b > c) {

    i--;

} else {

    i--;

    a--;

}

PIOR f(n) = n + 2 || O(n), Ω(n) e Ɵ(n)

MELHOR f(n) = n +1 || O(n), Ω(n) e Ɵ(n)

14) Apresente a função e a complexidade para o número de subtrações para o

pior e melhor caso

for (*i* = *0*; *i* < n; i++) {

for (j = 0; j < n; j++) {

          a--;

         b--;

      }

         c--;

}

TODOS f(n) = (2n + 1)n || O(n2), Ω(n2) e Ɵ(n2)

15) Apresente a função e a complexidade para o número de subtrações para o

pior e melhor caso.

for (*i* = *0*; *i* < n; i++) {

        for (j = 1; j <= n; j \*= 2) {

        b--;

        }

}

TODOS f(n) = (lg(n) + 1) \* n = n \* lg(n) + n O(n x lg(n)), Ω(n x lg(n)) e Ɵ(n x lg(n))

16)Apresente o tipo de crescimento que melhor caracteriza as funções abaixo

1. 3n = Linear
2. 1 = Constante
3. (3/2)n =Linear
4. 2n³ = Polinomial
5. = Exponencial
6. 3n² = Polinomial
7. 1000 = Constante
8. = Exponencial

17) Classifique as funções f1(n) = n2, f2(n) = n, f3(n) = 2n, f4(n) = (3/2)n, f5(n) = n3 e f6(n) = 1 de acordo com o crescimento, do mais lento para o mais rápido(Khan academy, adaptado)

* f6(n) = 1
* f2(n) = n
* f1(n) = n2
* f5(n) = n3
* f4(n) = (3/2)n
* f3(n) = 2n

18) Classifique as funções f1 (n) = n.log6 (n), f2 (n) = lg(n), f3 (n) = log8 (n), f4(n) =8n2, f5(n) = n.lg(n), f6(n) = 64, f7(n) = 6n3, f8 (n) = 82n e f9(n) = 4n de acordo com o crescimento, do mais lento para o mais rápido (Khan Academy, adaptado)

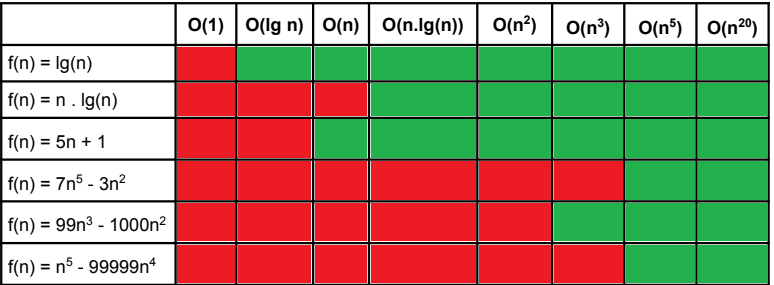
* f6(n) = 64
* f3(n) = log8(n)
* f2(n) = lg(n)
* f9(n) = 4n
* f1(n) = n.log6(n)
* f5(n) = n.lg(n)
* f4(n) = 8n2
* f7(n) = 6n3
* f8(n) = 82n

19 Faça a correspondência entre cada função f(n) com sua g(n) equivalente, em termos de Θ. Essa correspondência acontece quando f(n) = Θ(g(n)) (Khan Academy, adaptado)

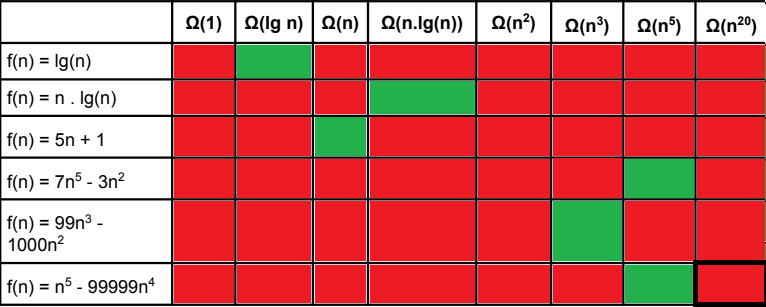
|  |  |  |
| --- | --- | --- |
| n + 30 | Θ | 3n-1 |
| n²+2n-10 | Θ | n²+3n |
| n³ .3n | Θ. |  |
| lg(n) | Θ. | lg(2n) |

1) Encontre o maior e menor valores em um array de inteiros e, em seguida, encontre a função de complexidade de tempo para sua solução.

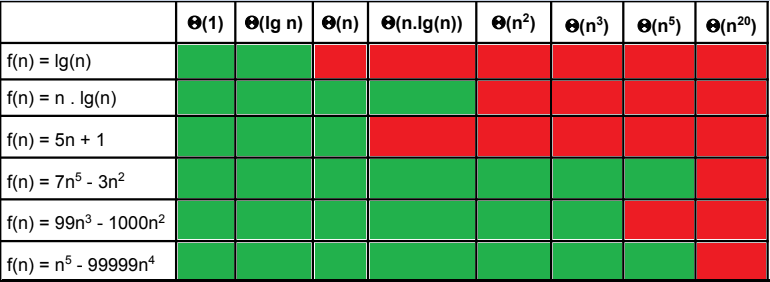
3) Preencha verdadeiro ou falso na tabela abaixo:



4) Preencha verdadeiro ou falso na tabela abaixo:



5) Preencha verdadeiro ou falso na tabela abaixo:



6) Dado f(n)=3n²-5n-9, g(n) = n\*lg(n), l(n) = n.lg²(n) e h(n) = 99n^8, qual é a ordem de complexidade das operações:

a) f(n) + g(n) - h(n) = O(n^8)

b) O(f(n) + O(g(n)) - O(h(n)) = O(n²)

c) f(n) x g(n) = O(n³\*lg(n))

d) g(n) x l(n) + h(n) = O(n^8)

e) f(n) x g(n) x l(n) = O(n^4\*lg³(n))

f) O(O(O(O(f(n))))) = O(n²)

7) Dada a definição da notação O:

* Mostre um valor c e outro m tal que, para n ≥ m, |3n2+ 5n +1| ≤ c x |n2| provando que 3n2+ 5n +1 é O(n2)

C=5

|  |  |  |
| --- | --- | --- |
| N | 3n²+ 5n +1 | C \* n² |
| 2 | 23 | 20 |
| 3 | 33 | 45 |
| 4 | 69 | 80 |
| 5 | 101 | 125 |

* Mostre um valor c e outro m tal que, para n ≥ m, |3n2+ 5n +1| ≤ c x |n3|, provando que 3n2+ 5n +1 é O(n3)

|  |  |  |
| --- | --- | --- |
| N | 3n²+ 5n +1 | C \* n³ |
| 1 | 9 | 5 |
| 2 | 23 | 40 |
| 3 | 33 | 135 |
| 4 | 69 | 320 |
| 5 | 101 | 625 |

* Prove que 3n2+ 5n +1 não é O(n)

|  |  |  |
| --- | --- | --- |
| N | 3n²+ 5n +1 | C \* n |
| 2 | 23 | 10 |
| 3 | 33 | 15 |
| 4 | 69 | 20 |
| 5 | 101 | 25 |

8)