Trabalho Teórico 3

**Exercícios iniciais**

1a) 20= 1 d) 23= 8 g) 26= 64 j) 29= 512

b) 21= 2 e) 24= 16 h) 27= 128 k) 210 = 1024

c) 22=4 f) 25= 32 i) 28= 256 l) 211= 2048

2a) lg(2048) =11 d) lg(256) = 8 g) lg(32) =5 j) lg(4) =2

b) lg(1024) =10 e) lg(128) = 7 h) lg(16) = 4 k) lg(2) =1

c) lg(512) = 9 f) lg(64) =6 i) lg(8) =3 l) lg(1) =0

3a) 4,01˥ = 5 d) 4,99 ˩ = 4 g) lg(17) =4,08 j) lg(15) = 3,98

b) 4,01˩ = 4 e) lg(16) ˥ = 4 h) lg(17)˥=5 k) lg(15) ˥ =4

c) 4,99 ˥ = 5 f) lg(16) ˩ = 4 i) lg(17) ˩ =4 l) lg(15) ˩ =3

d) f(n) = sqrt(n) i) f(n) = 5n4

e) f(n) = lg (n) = log2 f) f(n) = 3n2+ 5n - 3 g) f(n) = - 3n2+ 5n - 3

h) f(n) = | - n2 e) f(n) = lg (n) = log2 i) f(n) = 5n4 j) f(n) = n \* lg (n)

|  |  |
| --- | --- |
| 4a) f(n) = n | b) f(n) = n2 |
| c) f(n) = n3 | d) f(n) = sqrt(n) |
| e) f(n) = lg (n) = log2 | f) f(n) = 3n2+ 5n – 3 |
| g) f(n) = - 3n2+ 5n – 3 | h) f(n) = | - n2| |
| i) f(n) = 5n4 | j) f(n) = n \* lg (n) |

**Contagem de operações**

Exercícios Resolvidos.

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

a--;

a -= 3;

a = a - 2;

Resp: 3.

2) Calcule o número de adições que o código abaixo realiza:

...

if (a + 5 < b + 3)

{

i++;

++b;

a += 3;

}

Else

{

j++;

}

...

Resp: Melhor Caso 3

Pior Caso 5

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

...

if (a + 5 < b + 3 || c + 1 < d + 3)

{

i++;

++b;

a += 3;

}

else

{

j++;

}

...

Resp: Melhor Caso 5

Pior Caso 7

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

...

for (int i = 0; i < 4; i++)

{

a--;

}

...

Resp: 4

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

...

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

{

a--;

b--;

}

...

Obs: Resp dada em função de “n”.

Resp: 2n

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

...

int i = 0,

b = 10;

while (i < 3)

{

i++;

b--;

}

Resp: 3

7) Calcule o número de subtrações que o código abaixo realiza

...

for (int i = 3; i < n; i++)

{

a--;

}

...

Resp: -3+n.

Exercícios

5) Calcule o número de subtrações que o código abaixo realiza

...

int i = 10;

while (i >= 7)

{

i--;

}

...

Resp 4.

6) Calcule o número de subtrações que o código abaixo realiza

...

for (int i = 5; i >= 2; i--)

{

a--;

}

...

Resp 4x2=8

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

... for (int i = 0; i < 5; i++)

{

if (i % 2 == 0)

{

a--;

b--;

}

Else

{

c--;

}

}...

Resp (3x2)+(3x1)=9

Exercícios Resolvidos.

8) Calcule o número de subtrações que o código abaixo realiza

...

int a = 10;

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 2; j++)

{

a--;

}

}

Resp 3x2x1=6

Exercícios

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

...

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

{

for (int j = 0; j < n; j++)

{

a--;

}

}

...

Resp: n2 x 1

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

Int i = 1,

b = 10;

while (i > 0)

{

b--;

i = i >> 1;

}

i = 0;

while (i < 15)

{

b--;

i += 2;

}

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

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

for (int j = 0; j < n - 3; j++)

a \*= 2;

Resp: n \*(n-3).

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

for (int i = n - 7; i >= 1; i--)

for (int j = 0; j < n; j++)

a \*= 2;

Resp: (n-7) \* (n) se ‘n > 7’.

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

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

a \*= 2;

Resp: lg(n)+1

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

for (int i = n+4; i > 0; i >>= 1)

a \*= 2;

Resp: n/2 1

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

for (int i = n - 7; i >= 1; i--)

for (int j = n - 7; j >= 1; j--)

a \*= 2

Resp: (n-7)2

Exercícios Resolvidos.

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

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

a \*= 2;

Resp: lg(n)+1 (arredondado pra baixo)

Exercícios

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

for (int i = n + 1; i > 0; i /= 2)

a \*= 2;

Resp: lg(n+1)+1

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

for (int i = n; i > 1; i /= 2)

a \*= 2;

Resp: lg(n)

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

for (int i = 1; i < n; i \*= 2)

a \*= 2;

Resp: (lg(n)+1)\* 2

lg(n) pois é a condição de parada é \*2 pois temos “i\*=2” e “a\*=2;”

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

for (int i = 1; i <= n; i \*= 2)

a \*= 2;

Resp: (lg(n)+2)\* 2

Exercícios Resolvidos.

a) 3n + 2n2

i = 0;

while (i < n)

{

i++;

a --; b--; c--;

}

for (i = 0; i < n; i++)

for (j = 0; j < n; j++

a--; b--;

b) 5n + 4n3

i = 0;

while (i < n)

{

for (j = 0; j < 5; j++)

{

a--;

}

I++;

}

for (i = 0; i < n; i++)

for (j = 0; j < n; j++)

for (k = 0; k < n; k++)

a--; b--;c--; d--;

c) lg(n) + n

for (int i = n; i > 0; i /= 2) //primeiro for faz lg(n)

a --;

for (j = 0; j < n; j++) //segundo for faz a - - n vezes.

a--;

d) 2n3 + 5

for (i = 0; i < n; i++) //3 for alinhados para fazer n3 com duas variáveis pra multiplicar por 2

for (j = 0; j < n; j++)

for (k = 0; k < n; k++)

a--; b--;

for (j = 0; j < 5; j++) //segundo for faz a - - 5 vezes.

a--;

e) 9n4 + 5n2 + n/2

for (i = 0; i < n; i++) //4 for alinhados para fazer n4 + um for de 0 a 9 pra fazer 9\* uma variavel

for (j = 0; j < n; j++)

for (k = 0; k < n; k++)

for(l=0; l < n; l++)

for(h=0; l < 9; h++)

a--;

for (i = 0; i < n; i++)//2 for alinhados para fazer n2 + um for de 0 a 5 pra fazer 5\* uma variavel

for (j = 0; j < n; j++

for(h=0; l < 5; h++)

a--;

for (int i = n; i >=1 i /= 2) //for pra fazer n/2\*variável.

a --;

f) lg(n) + 5 lg(n)]

for (int i = n; i > 0; i /= 2) //primeiro for faz lg(n)

a --;

for (j = 0; j < 5; j++) // for pra fazer 5 \* x

for (int i = n; i > 0; i /= 2) for pra fazer ln(n)

a --;

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?

Resp:“min > array[i]”

2º) Quantas vezes ela será executada?

Resp: n-1

Exercícios.

19)Qual é a diferença entre as notações O, Ω e Θ?

**Notação Θ**

Uma função f(n) é Θ(g(n)) se existem constantes positivas c1 e c2 tal que, multiplicadas por g(n), c1g(n) e c2g(n) limitem f(n) acima e abaixo, para um valor a partir de um determinado no.

Segundo o prof. Meidanis, a expressão f(n) = Θ(g(n)) significa que as funções f(n) e g(n) possuem a mesma ordem de crescimento, isto é, crescem de forma equiparável. As funções n2 e 5n2, por exemplo, crescem em ritmo quadrático.

**Notação O**

Quando uma função f(n) é O(g(n)), a função g(n) impõe a f(n) um limite assintótico superior. Ou seja, multiplicada por uma constante c, a função g(n) limita superiormente a função f(n), a partir de um determinado  no.

A expressão f(n) = O(g(n)) significa que f(n) não cresce mais que g(n), podendo crescer de forma igual ou inferior.

Quando questionado sobre a relação da notação O com o pior caso de um algoritmo, o prof. Meidanis respondeu que essa é a notação mais utilizada na análise de complexidade de algoritmos em geral, normalmente para pior caso (entretanto, há casos como o QuickSort, por exemplo, que a notação O também é utilizada para o caso médio). As outras são raramente utilizadas: a notação Ω é usada algumas vezes para definição de limites inferiores. Porém, é importante lembrar que as definições de notações são independentes da análise de algoritmos, podendo ser utilizados para outros fins.