Trabalho Teórico 8

Marco Aurélio Silva de Souza Júnior - 696809

Exercício Resolvido 1:

a) $2^{10} = 1024$
b) $lg(1024) = 10$
c) $lg(17) = 4.08746\ldots$
d) $\lceil lg(17) \rceil = 5$
e) $\lfloor lg(17) \rfloor = 4$

Exercício Resolvido 2:

Exercício Resolvido 3:

```
for (int i = 0; i < n; i++){
  if (i % 2 == 0){
    a--;
    b--;
  } else {  // Melhor caso: f(n) = n, logo, O(n), Ω(n) e θ(n)
    c--;
```

```
  }  // Pior caso: f(n) = 2n, logo, O(n), Ω(n) e θ(n)
}
```

Exercício Resolvido 4:

```
for (int i = 3; i < n; i++){
  a--;
} //n - 3 subtrações, logo O(n), Ω(n) e θ(n)
```

Exercício Resolvido 5:

```
for (int i = n; i > 0; i /= 2)
  a *= 2;
// O(lg n), Ω(lg n) e θ(lg n)
```

Exercício Resolvido 6:

```
class Log {
  public static void main(String[] args) {
    int[] n = { 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 31, 32, 33, 63, 64, 65 };
    int cont;

    // laco para atribuir a i, no laco interno, os valores do array n
    for (int k = 0; k < n.length; k++) {
      System.out.print("\n[n = " + n[k] + "] => ");  // [n = 4]
      cont = 0;

      // exibe os valores que i recebe, quando i/=2, e i comeca com cada um dos valores do array n.
      for (int i = n[k]; i > 0; i /= 2) {
        //  \theta lg(n)
        System.out.print(" " + i);  // 4 2 1 // 5 2 1 // ... // 65 32 16 8 4 2 1
        cont++;
      }
      System.out.print(" (" + cont + " vezes)");  //
    }
    System.out.print("\n");
  }
}
```

Exercício 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

```
void show_minimo_maximo(int arr[], int n) {  // n: tamanho do array
  int min, max, cont = 0;
  min = max = arr[cont];

  for(cont = 1; cont < n; cont++) {
    if(arr[cont] < min) // n-1 comparacoes
      min = arr[cont];  // pior caso: array em ordem decrescente. n-1 movimentacoes.
```

```
    if(arr[cont] > max) // n-1 comparacoes
      max = arr[cont];  // pior caso: array em ordem crescente. n-1 movimentacoes.
  }
  printf("Maximo = %i\n\n", max);
  printf("Minimo = %i\n\n", min);
}
```

Função de complexidade de tempo: $\theta(n)$.

Exercício Resolvido 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?

Resp.: A primeira opção é mais eficiente por ter custo $\theta(n)$, enquanto a segunda tem custo $\theta(n.lg(n)) + \theta(lg(n))$.

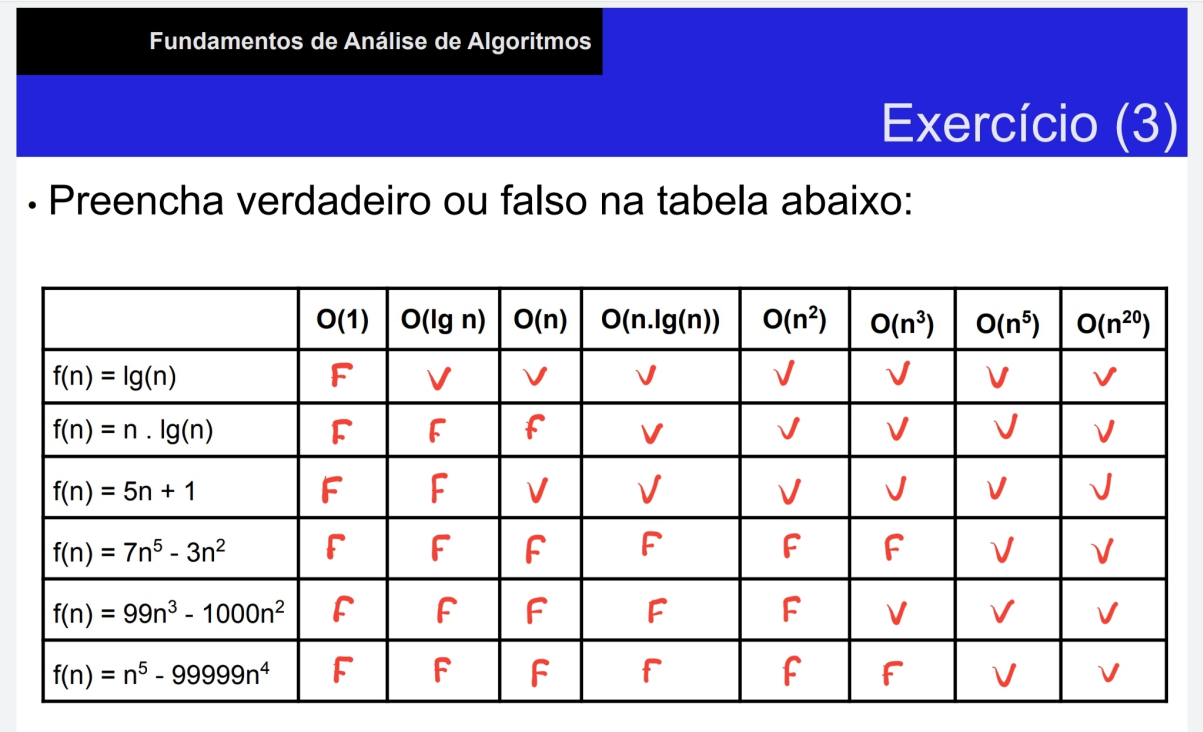
Exercício Resolvido 11:

a), b), c) $3n^2 + 5n + 1 = \mathrm{O}(n^2)$ ou $\mathrm{O}(n^3)$

d), e), f) $3n^2 + 5n + 1 = \Omega(n)$ ou $\Omega(n^2)$

g), h), i) $3n^2 + 5n + 1 = \Theta(n^2)$

Exercício 3 (slide 69):



Exercício 4:

Fundamentos de Análise de Algoritmos

Exercício (4)

- Preencha verdadeiro ou falso na tabela abaixo:

	Ω(1)	Ω(lg n)	Ω(n)	Ω(n.lg(n))	Ω(n^2)	Ω(n^3)	Ω(n^5)	Ω(n^{20})
f(n) = lg(n)	V	V	F	F	F	F	F	F
f(n) = n . lg(n)	V	V	V	V	F	F	F	F
f(n) = 5n + 1	V	V	V	F	F	F	F	F
f(n) = $7n^5 - 3n^2$	V	V	V	V	V	V	V	F
f(n) = $99n^3 - 1000n^2$	V	V	V	V	V	V	F	F
f(n) = $n^5 - 99999n^4$	V	V	V	V	V	V	V	F

Exercício 5:

Fundamentos de Análise de Algoritmos

Exercício (5)

- Preencha verdadeiro ou falso na tabela abaixo:

	Θ(1)	Θ(lg n)	Θ(n)	Θ(n.lg(n))	Θ(n^2)	Θ(n^3)	Θ(n^5)	Θ(n^{20})
f(n) = lg(n)	F	F	V	F	F	F	F	F
f(n) = n . lg(n)	F	F	F	V	F	F	F	F
f(n) = 5n + 1	F	F	V	F	F	F	F	F
f(n) = $7n^5 - 3n^2$	F	F	F	F	F	F	V	F
f(n) = $99n^3 - 1000n^2$	F	F	F	F	F	V	F	F
f(n) = $n^5 - 99999n^4$	F	F	F	F	F	F	V	F

Exercício 6:

$f(n) = 3n^2 - 5n - 9, g(n) = n.lg(n), l(n) = n.lg^2(n), h(n) = 99n^8$

a) f(n) + g(n) - h(n) = $\Theta(n^8)$

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

c) f(n) x g(n) = $\Theta(n^3.lg(n))$

d) g(n) x l(n) + h(n) = $\Theta(n^8)$

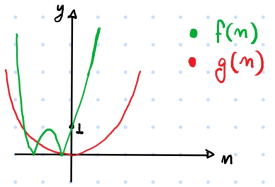
e) f(n) x g(n) x l(n) = $\Theta(n^4.lg^3(n))$

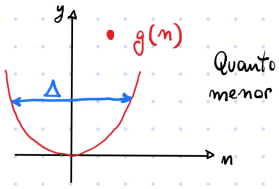
f) O(O(O(O(f(n))))) = $O(n^2)$

Exercício 7:

Sendo $f(n) = |3n^2 + 5n + 1|$ e
$g(n) = c \cdot |n^2|$

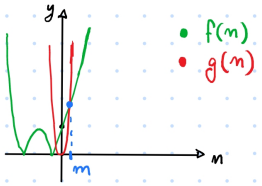
Para $c = 1$, o gráfico das funções é dado por:





Quanto maior for c, menor será a distância Δ

então devemos achar um c, tal que, para $n \geq 0$, exista um m em que $\forall n \geq m$, $g(n) \geq f(n)$.



Para $c \to +\infty$, a distância Δ será 0 e $m = 0$

De modo análogo, quanto maior for a potência de n em g(n), menor será a distância Δ e mais próximo de 0 será o m.

Assim, percebemos que $f(n) = O(n^x)$, onde $x \geq 2$.

Por outro lado, se $g(n) = c \cdot |n|$, nunca haverá um c que fará a função g(n) dominar assintoticamente f(n), logo f(n) não é O(n).

Exercício 8:

Sendo $g(n) = 3n^2 + 5n + 1$ e $f(n) = n^2$,
se c=0 e m=0, $\forall n \geq m$, $|g(n)| \geq c.|f(n)|$

Sendo $g(n) = 3n^2 + 5n + 1$ e $f(n) = n$,
se c=0 e m=0, $\forall n \geq m, |g(n)| \geq c.|f(n)|$

Sendo $g(n) = 3n^2 + 5n + 1$ e $f(n) = n^3$,
se c=0 e m=0, $\forall n \geq m, |g(n)| \geq c.|f(n)|$

Se c > 0, analogamente à explicação do exercício 7, a distância Δ será menor à medida que c aumenta, nunca havendo uma situação em que f(n) se torna limite assintótico inferior para g(n).

Exercício 9:

Sendo $g(n) = 3n^2 + 5n + 1$ e $f(n) = n^2$,
se $c_1 = 0, c_2 = 1, m = 0, \forall n \geq m, c_1.f(n) < g(n) < c_2.f(n)$

g(n) não é $\Theta(n)$ pois, f(n)=n nunca será limite assintótico de g(n). Analogamente, g(n) não é $\Theta(n^3)$.

Exercício 10:

Faça um resumo sobre Teoria da Complexidade, Classes de Problemas P,
NP e NP-Completo. Use LaTeX e siga o modelo de artigos da SBC (sem
abstract, resumo e seções) com no máximo duas página

Exercício Resolvido 12:

Função de complexidade:
Pior caso: $f(n) = 2 + (n - 2)$ movimentações, $f(n) = 1 + 2(n - 2)$ comparações.
Melhor caso: $f(n) = 2$ movimentações, $f(n) = n - 1$ comparações.

Complexidade:
Pior caso: $\Theta(n)$ movimentações e $\Theta(n)$ comparações.
Melhor caso: $\Theta(1)$ movimentações e $\Theta(n)$ comparações.

Exercício Resolvido 13:

Pior caso: $f(n) = n + 2$, $\Theta(n)$.
Melhor caso: $f(n) = n + 1$, $\Theta(n)$.

Exercício Resolvido 14:

Pior caso = melhor caso = $f(n) = n(2n + 1)$, $\Theta(n^2)$.

Exercício Resolvido 15:

Pior caso = melhor caso = $f(n) = n.lg(n) + n$, $\Theta(n.lg(n))$

Exercício 11:

```
void sistemaMonitoramento() {
  if (telefone() == true && luz() == true){
    alarme(0);
  } else {
    alarme(1);
```

```
  }
  for (int i = 2; i < n; i++){
    if (sensor(i- 2) == true){
      alarme (i - 2);
    } else if (camera(i- 2) == true){
      alarme (i - 2 + n);
} } }
```

Alarme: $f(n) = n - 1, \Theta(n)$.

Telefone e luz: $f(n) = 1, \Theta(1)$.

Sensor e câmera: $f(n) = n - 2, \Theta(n)$.

Exercício 12:

```
for(int i = 0; i < n; i++) {
  array[i] = array[i+1];
  tmp = array[i];
}
```

Operação relevante: movimentação de array.

complexidade: $f(n) = 2.n, \Theta(n)$.

Exercício Resolvido 16:

	Constante	Linear	Polinomial	Exponencial
3n		✗		
1	✗			
(3/2)n		✗		
$2n^3$			✗	
2^n				✗
$3n^2$			✗	
1000	✗			
$(3/2)^n$				✗

Exercício Resolvido 17:

$f_6(n) = 1$

$f_2(n) = n$

$f_1(n) = n^2$

$f_5(n) = n^3$

$f_4(n) = (3/2)^n$

$f_3(n) = 2^n$

Exercício Resolvido 18:

$f_6(n) = 64$

$f_3(n) = log_8(n)$

$f_2(n) = lg(n)$

$f_9(n) = 4n$

$f_1(n) = n.log_6(n)$

$f_5(n) = n.lg(n)$

$f_4(n) = 8n^2$

$f_7(n) = 6n^3$

$f_8(n) = 8^{2n}$

Exercício Resolvido 19:

$n + 30 \longleftrightarrow 3n - 1$

$n^2 + 2n - 10 \longleftrightarrow n^2 + 3n$

$n^3.3n \longleftrightarrow n^4$

$lg(n) \longleftrightarrow lg(2n)$

Exercício 13:

Executando-se *n* pesquisas, o custo disto seria $\Theta(n^2)$, assim, seria mais vantajoso ordenar e executar a pesquisa binária para um custo de $\theta(n.lg(n)) + \theta(lg(n))$.