Universidade Federal do Ceará – UFC
Centro de Ciências – CC
Mestrado e Doutorado em Ciências da Computação - MDCC
Estruturas de Dados

Exercício: Complexidade de Algoritmos
Objetivos: Exercitar os conceitos relacionados à Complexidade de Algoritmos.
Data da Entrega: 08/09/2022

OBS 1: Exercício Individual.
OBS 2: A entrega da lista deverá ser realizada via SIGAA.
OBS 3: Esta lista poderá ser solucionada utilizando-se as linguagens Go, Rust, Escala ou C++.

NOME: **Patrícia de Sousa Paula**_____ MATRÍCULA: **521773**

Questão 1

As funções f(n) mostradas abaixo fornecem o tempo de processamento T(n) de um algoritmo resolvendo um problema de tamanho n. Complete a tabela abaixo colocando, para cada algoritmo, sua complexidade (O maiúsculo) e a ordem do mais eficiente para o menos eficiente. Em caso de empate repita a ordem (por exemplo: 1º ,2º, 2º, ...).

f(n)	$O(\ldots)$	*ordem*
$5 + 0.001n^3 + 0.025n$	$O(n^3)$	9o
$500n + 100n^{1.5} + 50n \log_{10} n$	$O(n^{1.5})$	5o
$0.3n + 5n^{1.5} + 2.5 \cdot n^{1.75}$	$O(n^{1.75})$	6o
$n^2 \log_2 n + n(\log_2 n)^2$	$O(n^2 \log n)$	8o
$n \log_3 n + n \log_2 n$	$O(n \log n)$	2o
$3 \log_8 n + \log_2 \log_2 \log_2 n$	$O(\log n)$	1o
$100n + 0.01n^2$	$O(n^2)$	7o
$0.01n + 100n^2$	$O(n^2)$	7o
$2n + n^{0.5} + 0.5n^{1.25}$	$O(n^{1.25})$	4o
$0.01n \log_2 n + n(\log_2 n)^2$	$O(n (\log n)^2)$	3o
$100n \log_3 n + n^3 + 100n$	$O(n^3)$	9o
$0.003 \log_4 n + \log_2 \log_2 n$	$O(\log n)$	1o

Questão 2

Os algoritmos abaixo são usados para resolver problemas de tamanho n. Descreva e informe para cada algoritmo sua complexidade no pior caso (O maiúsculo/Ômicron). Tente entender o problema antes de apresentar uma resposta.

a) Resposta: **O pior caso vai ser próximo a O(n $\log^2$)**

```
for ( i=1; i < n; i *= 2 ) {
    for ( j = n; j > 0; j /= 2 ) {
        for ( k = j; k < n; k += 2 ) {
            sum += (-j * k) << i/2;
        }
    }
}
```

b) Resposta: **O pior caso vai ser próximo a O(n^2 $\log^2$)**

Leia(n);

x ←0

Para i ←1 até n faça

Para j ←i+1 até n faça

Para k ←1 até j-i faça

x←x + 1

Questão 3

Suponha um algoritmo A e um algoritmo B com funções de complexidade de tempo a(n) = n^2– n + 549 e b(n) = 49n + 49, respectivamente. Determine quais são os valores de n pertencentes ao conjunto dos números naturais para os quais A leva menos tempo para executar do que B.

Resposta:

A leva sempre mais tempo que B.

Questão 4

O Casamento de Padrões é um problema clássico em ciência da computação e é aplicado em áreas diversas como pesquisa genética, editoração de textos, buscas na internet, etc. Basicamente, ele consiste em encontrar as ocorrências de um padrão P de tamanho m em um texto T de tamanho n. Por exemplo, no texto T = "PROVA DE AEDSII" o padrão P = "OVA" é encontrado na posição 3 enquanto o padrão P ="OVO" não é encontrado. O algoritmo mais simples para o casamento de padrões é o algoritmo da "Força Bruta", mostrado abaixo. Analise esse algoritmo e responda: Qual é a função de complexidade do número de comparações de caracteres efetuadas no melhor caso e no pior caso. Dê exemplos de entradas que levam a esses dois casos. Explique sua resposta!

```
#define MaxTexto 100
#define MaxPadrao 10

/* Pesquisa o padrao P[1..m] no texto  T[1..n] */
void ForcaBruta(   char T[MaxTexto], int n,
                   char T[MaxPadrao], int m)
{
    int i,j,k;
    for( i = 0 ; i < n - m + 1 ; i++ )
    {
         k = i;
         j = 0;
         while ( ( j <= m ) && ( T[k] == P[j] ) )
         {
              j = j + 1;
              k = k + 1;
         }
         if (j > m)
         {
              printf("Casamento na posicao %d",i);
              break;
         }
    }
}
```

Questão 5

Considere que você tenha um problema para resolver e duas opções de algoritmos. O primeiro algoritmo é quadrático tanto no pior caso quanto no melhor caso. Já o segundo algoritmo, é linear no melhor caso e cúbico no pior caso. Considerando que o melhor caso ocorre 90% das vezes que você executa o programa enquanto o pior caso ocorre apenas 10% das vezes, qual algoritmo você escolheria? Justifique a sua resposta em função do tamanho da entrada.

Questão 6

Perdido em uma terra muito distante, você se encontra em frente a um muro de comprimento infinito para os dois lados (esquerda e direita). Em meio a uma escuridão total, você carrega um lampião que lhe possibilita ver apenas a porção do muro que se encontra exatamente à sua frente (o campo de visão que o lampião lhe proporciona equivale exatamente ao tamanho de um passo seu). Existe uma porta no muro que você deseja atravessar. Supondo que a mesma esteja a n passos de sua posição

inicial (não se sabe se à direita ou à esquerda), elabore um algoritmo para caminhar ao longo do muro que encontre a porta em O(n) passos. Considere que n é um valor desconhecido (informação pertencente à instância). Considere que a ação composta por dar um passo e verificar a posição do muro correspondente custa O(1).