IMD0029 - Estrutura de Dados Básicas 1 –2019.2 – Prova 01
Prof. Eiji Adachi M. Barbosa

Nome:_____

Matrícula: _____

ANTES DE COMEÇAR A PROVA, leia atentamente as seguintes instruções:

- Esta é uma prova escrita de caráter individual e sem consultas a pessoas ou material (impresso ou eletrônico).
- A prova vale 5,0 pontos na Unidade I e o valor de cada questão é informado no seu enunciado.
- Preze por respostas legíveis, bem organizadas e simples.
- As respostas devem ser fornecidas preferencialmente em caneta. Respostas fornecidas a lápis serão aceitas, mas eventuais questionamentos sobre a correção não serão aceitos.
- Celulares e outros dispositivos eletrônicos devem permanecer desligados durante toda a prova.
- **Desvios éticos ou de honestidade levarão a nota igual a zero na Unidade 1.**

Questão 1:(1,5ponto) Considere um array A contendo N inteiros, com possíveis repetições, já ordenado. Nesta questão, faça uma função que recebe como entrada um inteiro K e retorna um índice i tal que A[i] == K, sendo A[i] o elemento igual a K que está mais a esquerda em A. Por exemplo: considerando o array A = {0, 2, 2, 2, 3, 8, 8, 8, 10} e K = 8, sua função deve retornar i = 5, pois este é o índice do elemento igual a K que está mais a esquerda no array A. Sua função deverá obrigatoriamente ser **recursiva**, ter complexidade **Θ(lg(n))** e seguir a assinatura:

```
int find(int a[], int arraySize, int k)
```

Obs.: Nesta questão, não podem ser usadas instruções para realizar repetição, como for, while e do-while. Ou seja, você não poderá usar instruções de repetição; você deverá construir sua solução apenas com chamadas recursivas.

Questão2: (1,0 ponto) Nesta questão, implemente uma função que calcula de forma **recursiva** a divisão entre dois números inteiros e positivos. Sua função deverá seguir a assinatura:

```
int divide(int a, int b) // significa: a / b
```

Questão3: (1,0 ponto) Explique em quais ocasiões o algoritmo de ordenação Quick Sort cai em seu pior caso, deixando claro qual a sua complexidade assintótica neste caso. Em seguida, explique uma estratégia de implementação que garante que o Quick Sort nunca cai no seu pior caso.

Questão4: (1,5ponto) Para cada uma das afirmações a seguir, marque V (verdadeiro) ou F (falso), **justificando sucintamente** sua resposta. Marcações de V ou F **sem justificativas não serão aceitas.**

1 – ( ) Os algoritmos de busca sequencial e de busca binária podem ser empregados nas mesmas configurações de sequências de elementos passados como entrada.

2 – ( ) Os algoritmos de busca sequencial e de busca binária possuem mesma complexidade assintótica para o pior caso.

3 – ( ) Os algoritmos de ordenação Bubble Sort, Insertion Sort e Selection Sort possuem a mesma complexidade assintótica para o pior caso.

4 – ( ) Os algoritmos de ordenação Bubble Sort, Insertion Sort e Selection Sort possuem a mesma complexidade assintótica para o melhor caso.

5 – ( ) Os algoritmos de ordenação Quick Sort e Merge Sort possuem a mesma complexidade assintótica para o pior caso.

6 – ( ) Os algoritmos de ordenação Quick Sort e Merge Sort possuem a mesma complexidade assintótica para o melhor caso.

7 – ( ) O algoritmo de ordenação Merge Sort implementa uma estratégia de divisão e conquista recursiva em que a cada nível da recursão o problema de tamanho N é sempre dividido em dois sub-problemas de tamanhos N/2 (ou aproximadamente N/2).

8 – ( ) O algoritmo de ordenação Quick Sort implementa uma estratégia de divisão e conquista recursiva em que a cada nível da recursão o problema de tamanho N é sempre dividido em dois sub-problemas de tamanhos N/2 (ou aproximadamente N/2).