IMD0029 - Estrutura de Dados Básicas 1 –2024.1 – 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 em caneta. Respostas em 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) Sequências bitônicas são aquelas que possuem duas sequências, sendo uma sequência inicial crescente, seguida de uma sequência decrescente. Ou seja, os elementos de uma sequência bitônica inversa obedecem a seguinte relação:

$$A_0 < A_1 < ... < A_{i-1} < A_i > A_{i+1} > ... > A_n$$

Considere que um array bitônico é um array de inteiros sem repetições cujos elementos representam uma sequência bitônica. Neste contexto, implemente uma função que recebe como entrada um array bitônico e retorna o índice do elemento do "pico". O elemento do "pico" é o último elemento da sequência inicial crescente e o primeiro elemento da sequência final decrescente, ou seja, é o elemento Ai da relação acima. Sua função deverá obrigatoriamente ser **recursiva**, ter complexidade **Θ(lg(n))** e seguir a assinatura:

```
int acharPico(int array[], int esquerda, int direita)
```

Obs. 1: Considere que a chamada inicial à função acharPico recebe inicialmente para os parâmetros esquerda e direita os valores 0 e tamanho-1, em que tamanho representa a quantidade de elementos do array.
Obs. 2: 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: (1,5 ponto) Explique em quais situações o Quick Sort cai em seu pior caso, explicitando qual a complexidade do seu pior caso, e discuta ao menos uma estratégia para mitigar esse problema.

Questão 3: (2,0 pontos) 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 – (F) O algoritmo de busca binária possui complexidade assintótica de ordem logarítmica (O(lgn)) no melhor caso e complexidade assintótica de ordem linear (O(n)) no pior caso.

2 – ( ) O algoritmo de ordenação Selection Sort possui a mesma complexidade tanto para o melhor caso quanto para o pior caso, tornando-o mais previsível em termos de tempo de execução.

3 – ( ) O algoritmo de ordenação Insertion Sort possui a mesma complexidade tanto para o melhor caso quanto para o pior caso, tornando-o mais previsível em termos de tempo de execução.

4 – ( ) Considerando o melhor caso, o Insertion Sort tem menor complexidade assintótica do que o Selection Sort.

5 – ( ) O algoritmo de ordenação Merge Sort possui a mesma complexidade assintótica no melhor e no pior caso.