Pontifícia Universidade Católica de Minas Gerais
Curso de Ciência da Computação - Coração Eucarístico
Profs. Max do Val Machado e Felipe Domingos da Cunha
Disciplina: Algoritmos e Estruturas de Dados II

Prova I :

Aluno: _____

1. Dado o código abaixo, apresente a função de complexidade para o número de multiplicações para o pior e melhor caso usando a notação Θ. Além disso, descreva quando acontece cada um desses dois casos. Em seguida, responda (e justifique) se esse custo para cada caso é $O(n^2 \times lg\ n)$ e $\Omega(lg\ n)$.

```
Random gerador = new Random();
gerador.setSeed(4);
for (int i = 4; i < n; i++){
    if(Math.abs(gerador.nextInt()) % 9 < 4){
        a *= 2; b *= 3; l *= 2;
    } else if (Math.abs(gerador.nextInt()) % 9 == 5) {
        a *= 2; l *= 3;
    } else if (Math.abs(gerador.nextInt()) % 9 > 5) {
        a *= 2;
    }
}
for (int i = n; i >= 1; i = i >> 1, a *= 2);
for (int i = n - 2; i > 5; i--, b *= 3);
for (int i = n - 1; i >= 1; i /= 2, a *= 2);
if ((n < a - 3) == false || n > b + 4){
    l*= 2; k*=3; m*=7;
} else {
    l*= 5;
}
for (int i = 0; i < n ; i++){
    for (int j = 0; j < n - 1; j++){
        l = a * 2 + b * 5;
    }
}
```

2. Um deque é um conjunto de itens a partir do qual podem ser eliminados e inseridos itens em ambas as extremidades. Chame as duas extremidades de um deque esq e dir. Sabendo que um deque pode ser representado como um *array* em Java, escreva quatro métodos removerDir, removerEsq, inserirDir, inserirEsq, para remover e inserir elementos nas extremidades esquerda e direita de um deque. Certifique-se de que as funções funcionem corretamente para o deque vazio e detectem o estouro e o *underflow* (tentativa de remoção quando a estrutura está vazia).

3. Suponha que os elementos de um *array* estão organizados como um *heap*, implemente um método que ache o maior elemento da subárvore da esquerda desse *heap*.

4. No trabalho prático, você aplicou os algoritmos de ordenação para organizar os Times considerando diversos campos-chave. Entretanto, em caso de empate o critério para desempatar foi ordenar Times de chaves iguais pelo nome do Time. Considerando esse critério, implemente uma função/método que compare os dois Times pelos nomes. Seu algoritmo deve ser recursivo, seguindo o cabeçalho INT COMPARENOME(TIME TIME1, TIME TIME2). Lembre-se, você deve comparar letra por letra e retornar valores que definam qual Time deverá preceder o outro na ordenação. Além disso, você pode optar entre *C* ou *Java*, lembrando de fazer as devidas consistências de cada uma.