Universidade Federal do Pampa
Engenharia de Software
Prof. Dr. Fábio Paulo Basso
Estruturas de Dados – 2023/1

Questão 1) Escreva um programa iterativo (Java) que recebe como entrada dois arranjos (dois vetores de inteiros), cada um com **n** elementos, e devolve como saída a interseção entre os dois. O retorno é outro vetor contendo os elementos comuns a ambos os vetores. Faça a análise de complexidade de seu algoritmo no pior caso e relate no seu PIÁ.

```java
public class IntersecaoArrays {
    public static void main(String[] args) {
        int[] vet1 = { 1, 2, 3, 4, 5, 6, 7, 8, 9 };
        int[] vet2 = { 4, 5, 6, 7, 8, 9, 10, 11, 12 };

        int[] intersecao = encontrarIntersecao(vet1, vet2);

        System.out.print("Interseção: ");
        for (int elemento : intersecao) {
            System.out.print(elemento + " ");
        }
    }

    public static int[] encontrarIntersecao(int[] vet1, int[] vet2) {
        int tamanhoMaximo = Math.min(vet1.length, vet2.length);
        int[] intersecao = new int[tamanhoMaximo];
        int contador = 0;

        for (int i = 0; i < vet1.length; i++) {
            for (int j = 0; j < vet2.length; j++) {
                if (vet1[i] == vet2[j]) {
                    intersecao[contador] = vet1[i];
                    contador++;
                    break;
                }
            }
        }

        int[] resultado = new int[contador];
        System.arraycopy(intersecao, 0, resultado, 0, contador);
        return resultado;
    }
}
```

Questão 2) O método Selection Sort, abaixo, sofreu um pequeno ajuste: passou a trabalhar com a seleção do maior valor, colocando-o na última posição; do segundo maior valor colocando-o na penúltima posição e, assim, sucessivamente. Complete-o!

```
public void selectionSort () {
   for (int i = ___this.size()-1__; i >= __1___; __i--__) {
      int maior = this.posMaior(___i___);
         if (maior != i)
            this.troca (i, maior);
      }
}
/* Métodos auxiliares: */
private int posMaior (int fim) {
   int maior = fim;
   for (int i = fim-1; i >= 0; i--)
      if (valor[i] > valor[maior])
         maior = i;
   return maior;
}
private void troca (int a, int b) {
   float aux;
   aux = valor[a];
   valor[a] = valor[b];
   valor[b] = aux;
}
```

Questão 3) Implemente, em Linguagem Java, um método que calcula o enésimo termo da série de Fibonacci, tendo como base a descrição matemática recursiva a seguir.

$$Fibonacci(n) = \begin{cases} 0 & \text{se } n = 0, \\ 1 & \text{se } n = 1 \text{ e} \\ Fibonacci(n-1) + Fibonacci(n-2) & \text{se } n > 1. \end{cases}$$

Exemplo:

Primeiros números da série de Fibonacci 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, …

```java
public class Fibonacci {
    public static void main(String[] args) {
        int n = 20;
        int resultado = Fibonacci(n);
        System.out.println("O " + n + "º termo da série de Fibonacci é: "
+ resultado);
    }

    public static int Fibonacci(int n) {
        if (n <= 1) {
            return n;
        } else {
            return Fibonacci(n - 1) + Fibonacci(n - 2);
        }
    }
}
```

(Questão 4) Dados quatro algoritmos (A, B, C e D) e suas respectivas fórmulas gerais, analise a complexidade de cada um deles e classifique-os em ordem crescente de complexidade. Responda no seu PIÁ.

- **A.** $O(n) = 4n^2 + 3n + 2$
- **B.** $O(n) = 2^n + 34$
- **C.** $O(n) = 2\sqrt{n} + 12$
- **D.** $O(n) = 3 + 5n \log n$

(Questão 5) Implemente um método que retorna o número de elementos pares da lista duplamente encadeada. Este método deve possuir o seguinte protótipo:

```
public int nroPares();
```

```java
public int nroPares(){
    int retorno = 0;
    for(Noh i = inicio; i != null; i = i.getProx()){
        if(i.getInfo()%2 == 0){
            System.out.println("Num: " +i.getInfo());
            retorno++;
        }
    }
    return retorno;
}
```

Questão 6) Implemente um método que insere números inteiros em uma lista duplamente encadeada de forma ordenada. Por exemplo, suponha que [2, 5, 7, 10] é uma LDE. Se o número 6 for inserido isso ocorrerá entre os elementos 5 e 7. O protótipo do método deve ser o seguinte. Explique no seu PIÁ a complexidade para o pior caso.

```
public int add_ordenado();
```

```java
public void add_ordenado(int a){
    Noh novo = new Noh(a);
    if(inicio == null){
        inicio = novo;
        ultimo = novo;
    }else {
        for(novo = inicio; novo != null; novo = novo.getProx()){
            if(novo.getInfo() > a){
                novo.setProx(inicio);
                inicio.setAnt(novo);
                inicio = novo;
            }
        }
    }

}
```

Questão 7) Implemente um método para imprimir números inteiros de forma recursiva de uma lista simplesmente encadeada. O protótipo é o seguinte:

```
public void imprime_rec();
```

```java
public void imprime_rec(Recursivo recursivo) {
    if (recursivo == null) {
        return;
    }

    System.out.println(recursivo.data);
    imprime_rec(recursivo.next);
}
```

Questão 8) Implemente um método que concatene duas listas duplamente encadeadas de números inteiros. Por exemplo, dada a LDE [1,2,3,4] e outra LDE [5,6,7, 8] o resultado será uma nova LDE resultante da concatenação das duas anteriores [1,2,3,4,5,6,7,8]. O protótipo é o seguinte:

```
public void concat(LDE l);
```

```java
public void concat(LDE l) {
    LDE resultList = new LDE();

    // Verificar se a primeira lista está vazia
    if (this.head == null) {
        resultList.head = l.head;
    }
    // Verificar se a segunda lista está vazia
    else if (l.head == null) {
        resultList.head = this.head;
    }
```

```
    // Caso as duas listas não estejam vazias
    else {
        Node current = this.head;

        // Percorrer a primeira lista até o último nó
        while (current.next != null) {
            current = current.next;
        }

        // Atualizar as referências para concatenar a segunda lista
        current.next = l.head;
        l.head.prev = current;

        resultList.head = this.head;
    }
}
```

Questão 9) Implemente um método que ordene uma lista simplesmente encadeada de inteiros utilizando o algoritmo bubbleSort.

Questão 10) Ordene crescentes os elementos do vetor [3, 7, 1, 4, 9, 2], utilizando os seguintes algoritmos de ordenação: Bubble, selection e insertion sort. Mostrar o estado do vetor toda a vez que ocorrer uma troca de elementos. Atenção, isso deve ser feito no próprio PIÁ ou, então, em papel e anexado ao PIÁ como foto.

Questão 11) Dado o vetor a seguir [7, 1, 1, 3, 2], explique com suas palavras o conceito de estabilidade em um algoritmo de ordenação. Além disso, demonstre, desenhando o estado do vetor a cada troca de elementos. Atenção, isso deve ser feito no próprio PIÁ ou, então, em papel e anexado ao PIÁ como foto.

Questão 12) Implemente, em Java, um método recursivo que representa a definição matemática recursiva a seguir. Além disso, responda: a implementação recursiva seria sua primeira opção para resolver o problema do somatório de números inteiros? Justifique sua resposta, mostrando matematicamente o porquê!

$$\sum_{k=m}^{n} = \begin{cases} m & se\ n = m\ e \\ m + \sum_{k=m+1}^{n} & se\ n > m \end{cases}$$