PARADIGMAS DE LINGUAGENS DE PROGRAMAÇÃO – LISTA DE EXERCÍCIOS 4b
Programação Orientada a Objetos – JAVA – 1º. Sem/2018 – Profa. Heloisa Camargo
==========

1) Definir uma classe que tem três Strings S1, S2 e S3 (objetos da classe String) como campos de dados e um método para, dada uma String como parâmetro, fazer, usando métodos da classe String:
   - Se a string começa com o caracter "a", concatenar essa string em S1; se a string começa com "b", concatenar em S2 e se começa com "c", concatenar em S3. Se a string começa com qualquer outro caracter, ignorar.
   - Para cada uma das strings S1, S2 e S3, contar quantas strings foram concatenadas para montar a string correspondente.
   - Escrever um método main que instancia um objeto da classe definida, lê algumas strings do teclado e, para cada uma delas, chama o método para concatenar a string dada.

2) Construir um método para, dados um caracter e uma string (objeto da classe String), retornar quantas vezes o caracter aparece na string, usando métodos da classe String.

3) Construir um método para, dados um caracter e uma string(objeto da classe String), eliminar da string dada todas as ocorrências do caracter, e retornar a nova string, usando métodos da classe String.

```
public String eliminaElem(char a, String cadeia) {
   int i = cadeia.indexOf(a);
   while (i != -1) {
      String s1 = cadeia.substring(0,i);
      String s2 = cadeia.substring(i+1);
      cadeia = s1 + s2;
      i = cadeia.indexOf(a);
  }
return cadeia;
}
```

4) Escreva uma classe em Java que represente o nome completo de uma pessoa, composto de três strings (objetos da classe String) que representam o nome próprio, nome do meio e nome de família da pessoa. Escreva nessa classe o método ***rubrica*** que retorna somente as iniciais do nome completo em caracteres minúsculos, e o método ***assinatura*** que retorna as iniciais dos nomes próprio e do meio (com pontos) e o nome de família completo. Por exemplo, se o nome da pessoa representado por essa classe for "Richard Langton Gregory", o método rubrica deve retornar "rlg" e o método assinatura deve retornar "R.L.Gregory". Usar métodos da classe String.

5) Construa uma classe, subclasse da classe LinkedList, que manipule estruturas de listas duplamente encadeadas em que seja possível acessar, além do elemento do início e do fim da lista, o antecessor e o sucessor de um dado elemento. (ATENÇÃO para o que é pedido neste exercício: uma classe que estende a class LinkedList deve ser criada, com as repectivas operações. Não é necessário definir método main nem instanciar objetos da classe definida.)

```
import java.util.*;
public class ListaDuplaEnc<E> extends LinkedList<E> {
   public E primeiro() {
     return getFirst();
   }
   public E ultimo() {
     return getLast();
```

```
    }
    public E antecessor(E elemento) {
      int i = indexOf(elemento);
      if (i == -1) {System.out.println("O elemento não se encontra na
lista");
                    return null;}
      if (i == 0) {System.out.println("O element é o primeiro da
lista");
                   return elemento;}
      return get(i-1);
   }
    public E sucessor(E elemento) {
      int i = indexOf(elemento);
      if (i == -1) {System.out.println("O elemento não se encontra na
lista");
                    return null;}
      if (i == size() - 1) {System.out.println("O element é o ultimo da
lista");
                    return elemento;}
      return get(i+1);
    }
}
```

6) Construir um programa para manipular duas pilhas, pilha1 e pilha2, usando a classe LinkedList com genéricos. Instanciar os objetos pilha1 e pilha2 como coleções de inteiros ou float. Fornecer uma série de dados e fazer:
   - se o número dado for negativo, empilhar na pilha1
   - se o número dado for positivo, empilhar na pilha2
   - se o número dado for zero, parar.
   - ao final, imprimir o conteúdo das duas pilhas.

7) O vetor listaNomes, definido como ArrayList, tem nomes em ordem alfabética. Escreva um método que recebe um nome na forma de String como parâmetro e insere o nome em listaNomes, na posição correta para manter a ordem alfabética. O método deve retornar sempre o valor booleano true.

```
ArrayList<String> listaNomes = new ArrayList<String>();
```

8) O vetor listaNum, definido como ArrayList, tem números em ordem crescente. Escreva um método que recebe um número como parâmetro e insere o número em listaNum, na posição correta para manter a ordem crescente. O método deve retornar sempre o valor booleano true. Escreva um método que recebe dois números como parâmetro, procura o primeiro número na coleção e, se encontrar, substitui pelo segundo (só a primeira ocorrência do número). Se a substituição ocorrer, retornar true, senão, retornar false.

```
ArrayList<Double> listaNum = new ArrayList<Double>();
```

9) Faça o seguinte, em Java:
   a) Defina uma classe que representa um Funcionário e que tenha três campos de dados: nome (String), CPF (String) e salário (double).

   b) Crie, dentro de outra classe, uma coleção do tipo ArrayList que vai conter objetos da classe Funcionário. Assumindo que a estrutura já está inicializada, defina um método para, dado um

nome de funcionário, retorne o seu CPF. Caso o nome não se encontre na coleção, retorne uma mensagem apropriada.

10) Uma conta bancária é representada por um número, nome do correntista, e saldo. Em uma operação de saque, caso não haja saldo suficiente, a operação não é realizada. Crie uma classe Conta em JAVA para representar a conta corrente e defina um método que faz a operação de saque de um valor passado como parâmetro e retorne o saldo restante. Caso o saque não seja feito, retornar -1. Crie uma classe Banco que contém uma coleção do tipo ArrayList, para manter contas de correntistas como objetos da classe Conta. Defina um método que recebe um número de conta como parâmetro, procura a conta na coleção e, se encontrar, faz a operação de saque usando o método da classe Conta.