**Classe LSENode:**

package br.unicap.pilhas;

class LSENode <T>{

private T info;

private LSENode<T> prox;

LSENode(T obj){

this.info = obj;

}

void setInfo(T obj){

this.info = obj;

}

T getInfo(){

return this.info;

}

void setProx(LSENode<T> prox){

this.prox = prox;

}

LSENode<T> getProx(){

return this.prox;

}

}

**Classe Stack:**

package br.unicap.pilhas;

public class Stack <T>{

private LSENode<T> topo;

public void push(T obj){

LSENode<T> novo = new LSENode<>(obj);

if (this.isEmpty()) {

this.topo = novo;

} else {

novo.setProx(this.topo);

this.topo = novo;

}

}

public T pop(){

LSENode<T> aux = this.topo;

this.topo = this.topo.getProx();

return aux.getInfo();

}

public boolean isEmpty(){

return this.topo==null;

}

public T top(){

return this.topo.getInfo();

}

public boolean isFull(){

return false;

}

public void exibir(){

LSENode aux = this.topo;

while(aux!=null){

System.out.print(aux.getInfo()+" ");

aux = aux.getProx();

}

}

}

**Classe ControllerPilha:**

package br.unicap.controles;

import br.unicap.pilhas.Stack;

public class ControllerPilha {

/\* 1º Questao \*/

public boolean stringBemFormada(String s) {

Stack<Character> pilhaString = this.parseStack(s);

boolean col = false, pare = false, chav = false;

boolean fechouCol = true, fechouPare = true, fechouChav = true;

LOOP:

while (!pilhaString.isEmpty()) {

Character c = pilhaString.pop();

switch (c) {

case ']':

if (!col) {

col = true;

fechouCol = false;

} else {

fechouCol = false;

break LOOP;

}

break;

case ')':

if (!pare) {

pare = true;

fechouPare = false;

} else {

fechouPare = false;

break LOOP;

}

break;

case '}':

if (!chav) {

chav = true;

fechouChav = false;

} else {

fechouChav = false;

break LOOP;

}

break;

}

if (col && c == '[') {

fechouCol = true;

col = false;

}

if (pare && c == '(') {

fechouPare = true;

pare = false;

}

if (chav && c == '{') {

fechouChav = true;

chav = false;

}

}

return fechouCol && fechouPare && fechouChav;

}

/\* 2º Questao \*/

public boolean isPalindromo(String frase){

Stack<Character> pilhaIndo = this.parseStack(frase.replaceAll(" ",""));

Stack<Character> copiaPilhaIndo = this.parseStack(frase.replaceAll(" ",""));

Stack<Character> pilhaVoltando = new Stack<>();

while(!copiaPilhaIndo.isEmpty()){

pilhaVoltando.push(copiaPilhaIndo.pop());

}

Character c1,c2;

boolean result = true;

while(!pilhaIndo.isEmpty()){

c1 = pilhaIndo.pop();

c2 = pilhaVoltando.pop();

if(Character.toUpperCase(c1)!=Character.toUpperCase(c2)){

result = false;

break;

}

}

return result;

}

/\* 3º Questao \*/

public Stack<Integer> converter(int decimal, int base){

Stack<Integer> resultado = new Stack<>();

int num = decimal;

int resto;

while(num>=base){

resto = num%base;

num = num/base;

resultado.push(resto);

}

resultado.push(num);

return resultado;

}

//METODOS AUXILIARES

private Stack<Character> parseStack(String s) {

Stack<Character> stackAux = new Stack();

int tam = s.length(), i;

for (i = 0; i < tam; i++) {

stackAux.push(s.charAt(i));

}

return stackAux;

}

}

**Classe PilhaMain:**

package br.unicap.TestaPilha;

import br.unicap.controles.ControllerPilha;

import br.unicap.pilhas.Stack;

public class PilhaMain {

public static void main(String[] args) {

ControllerPilha t = new ControllerPilha();

System.out.println("------Teste Questao 1º ------");

boolean result = t.stringBemFormada("sas{asd[sdds(sds]sd]sds}das");

String resultado = result ? "String bem formada" : "String mal formada";

System.out.println(resultado);

System.out.println("------Teste Questao 2º ------");

result = t.isPalindromo("Roma me tem amor");

resultado = result ? "Palindromo" : "nao palindromo";

System.out.println(resultado);

System.out.println("------Teste Questao 3º ------");

Stack<Integer> i = t.converter(123, 6);

i.exibir();

}

}

**Classe Carro:**

package br.unicap.dados;

public class Carro {

private int codigo;

private String carga;

public Carro(int codigo, String carga) {

this.codigo = codigo;

this.carga = carga;

}

public Carro(int codigo) {

this.codigo = codigo;

}

public int getCodigo() {

return codigo;

}

public void setCodigo(int codigo) {

this.codigo = codigo;

}

public String getCarga() {

return carga;

}

public void setCarga(String carga) {

this.carga = carga;

}

@Override

public String toString() {

return "Carro{" + "codigo=" + codigo + ", carga=" + carga + '}';

}

}

**Classe ControllerCarros:**

package br.unicap.controles;

import br.unicap.controles.excecoes.DadosRepetidosException;

import br.unicap.dados.Carro;

import br.unicap.pilhas.Stack;

import java.util.Scanner;

public class ControllerCarros {

Stack<Carro> carros = new Stack<>();

public void inserirCarro() throws DadosRepetidosException {

Scanner in = new Scanner(System.in);

System.out.println("Codigo Do Carro");

int cod = in.nextInt();

in.nextLine();

System.out.println("Descriçao da carga Do Carro");

String carga = in.nextLine();

Carro c = new Carro(cod, carga);

Stack<Carro> auxINV = new Stack();

Carro prox;

while (!carros.isEmpty()) {

prox = carros.pop();

if (prox.getCodigo() == cod) {

throw new DadosRepetidosException("Codigo do Carro");

} else if (prox.getCodigo() > cod) {

carros.push(prox);

break;

}

auxINV.push(prox);

}

carros.push(c);

while (!auxINV.isEmpty()) {

carros.push(auxINV.pop());

}

System.out.println("Carro adicionado!");

}

public void removerCarro() {

Scanner in = new Scanner(System.in);

boolean removido = false;

System.out.println("Codigo Do Carro");

int cod = in.nextInt();

in.nextLine();

Stack<Carro> auxINV = new Stack();

Carro prox;

while (!carros.isEmpty()) {

prox = carros.pop();

if (prox.getCodigo() == cod) {

removido = true;

break;

}

auxINV.push(prox);

}

while (!auxINV.isEmpty()) {

carros.push(auxINV.pop());

}

if (removido) {

System.out.println("Carro removido!");

} else {

System.out.println("Carro nao encontrado!");

}

}

public void exibirCarros() {

if (this.carros.isEmpty()) {

System.out.println("Sem Carros!");

} else {

this.carros.exibir();

System.out.println("");

}

}

}

**Classe TestaPilhaCarros:**

package br.unicap.TestaPilha;

import br.unicap.controles.ControllerCarros;

import br.unicap.controles.excecoes.DadosRepetidosException;

import java.util.Scanner;

public class TestaPilhaCarros {

public static void main(String[] args) throws CloneNotSupportedException{

Scanner in = new Scanner(System.in);

ControllerCarros cc = new ControllerCarros();

char op;

System.out.println("### Composiçao de Carros ###");

do {

menuOpcoes();

System.out.print("Informe a opção: ");

op = in.next().charAt(0);

in.nextLine();

switch (op) {

case '1':

try {

cc.inserirCarro();

} catch (DadosRepetidosException ex) {

System.out.println(ex.getMessage());

}

break;

case '2':

cc.removerCarro();

break;

case '3':

cc.exibirCarros();

break;

case '0':

System.out.println("Fim do programa!");

break;

default:

System.out.println("Opção inválida!");

}

} while (op != '0');

}

public static void menuOpcoes() {

System.out.println("[1] - Adicionar Carro\n"

+ "[2] - Remover Carro\n"

+ "[3] - Exibir Composiçao\n"

+ "[0] - Sair do programa");

}

}

**Exceção DadosRepetidosException:**

package br.unicap.controles.excecoes;

public class DadosRepetidosException extends Exception{

public DadosRepetidosException(String obj) {

super(obj+" Já existe!");

}

}