Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

**Curso de Tecnologia em Sistemas de Computação**
**Disciplina: Programação Orientada a Objetos**
**AP2 1° semestre de 2018.**

**Nome –**

**Assinatura –**

---

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

**Questão 1) (3.0 pontos)**
Dada uma expressão qualquer com parênteses, indique se a quantidade de parênteses está correta ou não, sem levar em conta o restante da expressão. Por exemplo:

**a+(b*c)-2-a: correta**
**(a+b*(2-c)-2+a)*2: correta**

enquanto
**(a*b-(2+c): incorreta**
**2*(3-a)): incorreta**
**)3+b*(2-c)(: incorreta**

Ou seja: (1) todo parênteses que fecha deve ter um outro parênteses que abre correspondente; (2) não pode haver parênteses que fecha sem um parênteses prévio que abre; e (3) a quantidade total de parênteses que abre e fecha deve ser igual.

Seu programa deve receber um arquivo, **<nome-arq.txt>**, com várias expressões e, deve imprimir no arquivo específico, cujo nome é **"saida"** acrescido do nome do arquivo original, a expressão e se ela está **"correta"** ou **"incorreta"**.

**RESPOSTA:**

```
import java.io.*;
import java.util.*;


public class Q1_AP2_POO_2018_1{
```

```
  public static void main(String[] args) throws IOException{
    BufferedReader in;
    in = new BufferedReader(new FileReader(args[0]));
    BufferedWriter out;
    out = new BufferedWriter(new FileWriter("saida-" + args[0]));
    try {
      String vet = in.readLine(), resp = "";
      while(vet != null){
        int abriu = 0, fechou = 0, i;
        for(i = 0; i < vet.length(); i++){
          if(vet.charAt(i) == ')'){
            if((abriu - fechou) == 0){
              resp += "incorreta\n";
              break;
            }
            fechou++;
          }
          else if(vet.charAt(i) == '(') abriu++;
        }
        if(i == vet.length()){
          if((abriu - fechou) != 0) resp += "incorreta\n";
          else resp += "correta\n";
        }
        vet = in.readLine();
      }
      in.close();
      out.write(resp);
      out.close();
    }
    catch (Exception e){
      System.out.println("Excecao\n");
    }
  }
}
```

**Questão 2) (3.0 pontos)**
Escreva um programa em Java que lê, de um arquivo passado como parâmetro de entrada, vários números e, em seguida, informe quantas vezes cada número aparece no arquivo de dados, em ordem crescente de valor:

| Arquivo de entrada | Resposta na tela |
|---|---|
| **8**<br>**10**<br>**8**<br>**260**<br>**4**<br>**10**<br>**10** | **4 aparece 1 vez(es)**<br>**8 aparece 2 vez(es)**<br>**10 aparece 3 vez(es)**<br>**260 aparece 1 vez(es)** |

**RESPOSTA:**

```
import java.io.*;
import java.util.*;

class no{
  int elem, total;
  no prox;
```

```
  no(int elem){
    this.elem = elem;
    total = 1;
    prox = null;
  }

  public String toString(){
    return elem + " aparece " + total + " vez(es)";
  }
}


class lista{
  no prim;

  lista(){ prim = null; }

  void insere(int elem){
    no novo, p = prim, ant = null;
    while((p != null) && (p.elem <= elem)){
      if(p.elem == elem){
        p.total++;
        return;
      }
      ant = p;
      p = p.prox;
    }
    novo = new no(elem);
    novo.prox = p;
    if(ant == null) prim = novo;
    else ant.prox = novo;
  }

  public String toString(){
    String resp = "";
    no p = prim;
    while (p != null){
      resp += p.toString() + "\n";
      p = p.prox;
    }
    return resp;
  }
}
public class Q2_AP2_POO_2018_1{
  public static void main(String[] args) throws IOException{
    BufferedReader in;
    in = new BufferedReader(new FileReader(args[0]));
    try {
      lista l = new lista();
      String s = in.readLine();
      while(s != null){
        l.insere(Integer.parseInt(s));
        s = in.readLine();
      }
      in.close();
      System.out.print(l);
    }
    catch (Exception e){
      System.out.println("Excecao\n");
    }
  }
```

```
}
```

**Questão 3) (4.0 pontos)**
Testes codificados são muito importantes no desenvolvimento de sistemas. A sua implementação garante consistência na execução, principalmente após modificações. Considere o código abaixo:

```
class TesteConta {
	public void test() {
		Conta c = new Conta(1, 100, "Fulano");
		assert(c.saldo() == 100);
		c.credito(150);
		assert(c.saldo() == 250);
		c.debito(300);
		assert(c.saldo() == 150);

		Conta ce = new Especial(2, 100, "Ciclano", 50);
		assert(ce.saldo() == 100);
		ce.credito(150);
		assert(ce.saldo() == 250);
		ce.debito(300);
		assert(ce.saldo() == 150);
	}
}

public class AP2_2018_1_Q3 {
	public static void main(String[] args) {
		TesteConta t = new TesteConta();
		t.test();
	}
}
```

A class TesteConta foi uma classe criada para testar as classes Conta e Especial, que representam contas corrente (número, saldo inicial e nome do correntista) e contas corrente especiais (com saldo extra) numa aplicação bancária. A título de curiosidade, a classe TesteConta utiliza o método **assert**, o qual recebe uma expressão booleana e levanta uma exceção (java.lang.AssertionError) quando a condição é falsa. O método **assert** só causa este comportamento quando a flag -enableassertions é passada via linha de comando. Caso não seja passada, nada acontece. É como se tivéssemos um if, o qual levanta uma exceção caso a condição dê falso. A vantagem de não usar o if é permitir que apenas eventualmente o código execute (num ambiente de testes, por exemplo).

Implemente as classes Conta e Especial baseado no código acima. Para este código, a exceção é levantada na chamada `assert(c.saldo() == 150);`, mas não é levantada na chamada `assert(ce.saldo() == 150);` em função do valor extra da conta especial.

RESPOSTA:

```
class Conta {
	private int numero;
	private double saldo;
	String nome;
```

```
	public Conta(int numero, double saldo, String nome) {
		this.numero = numero;
		this.saldo = saldo;
		this.nome = nome;
	}

	public void debito(double v) {
		if (this.saldo >= v) {
			this.saldo -= v;
		}
	}

	public void credito(double v) {
		this.saldo += v;
	}

	public double saldo() {
		return this.saldo;
	}
}

class Especial extends Conta {
	private double valor;
	public Especial(int numero, double saldo, String nome, double
valor) {
		super(numero, saldo + valor, nome);
		this.valor = valor;
	}
}
```