Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação Orientada a Objetos
AP1 2° semestre de 2019.

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) (4.0 pontos)

Escreva um programa em Java que receba, como argumento, várias Strings e retorne se elas são permutações entre si. Por exemplo, se o seu programa recebesse, como argumento, as strings **`amor roma ramo`**, seu programa deve informar que todas essas strings são permutações entre elas.

RESPOSTA:

```
public class Q1{
  public static void main(String[] args){
    int i, j, n = args.length;
    for(i = 0; i < n - 1; i++){
      for(j = i + 1; j < n; j++){
        if(!ePermutacao(args[i], args[j])){
            System.out.println(args[i] + " " + args[j] + " nao sao
permutacoes...\n");
          return;
        }
      }
    }
    System.out.println("Todas sao permutacoes...\n");
  }

  static boolean ePermutacao(String s1, String s2){
    if(s1.length() != s2.length()) return false;
    char aux1[] = s1.toCharArray(), aux2[] = s2.toCharArray();
    int i, j;
```

```
    for(i = 0; i < s1.length(); i++){
      for(j = 0; j < s1.length(); j++){
        if((aux1[i] == aux2[j]) && (aux1[i] != '0')){
          aux1[i] = aux2[j] = '0';
          break;
        }
      }
      if(j == s1.length()) return false;
    }
    return true;
  }
}
```

Questão 2) (3.0 pontos)
Dada a classe abaixo, a qual representa um ponto em 2 dimensões:

```
class Ponto {
        private double x, y;

        public Ponto(double x, double y) {
                this.x = x;
                this.y = y;
        }
}
```

(a) (1.0 pto) Defina uma classe Ponto3D que permite a criação de um ponto em 3 dimensões.
(b) (1.0 pto) Dado um outro ponto como argumento (um outro objeto Ponto3D ou as coordenadas x, y e z deste outro ponto), retorne o objeto Ponto3D referente à diferença entre as coordenadas.
(c) (1.0 pto) Calcule a distância entre 2 pontos definindo um método de instância (método não estático). Supondo ponto P com dimensões px, py, pz, Q com dimensões qx, qy e qz, a distância é calculada com a seguinte fórmula:

$$distancia = raiz\_quadrada\ (\ (px - qx)^2 + (py - qy)^2 + (pz - qz)^2\ )$$

Obs.: 1) Utilize os conceitos de OO vistos sempre que possível; 2) A raiz quadrada pode ser calculada com o método Math.sqrt

GABARITO:

```
class Ponto {
        private double x, y;
        public Ponto(double x, double y) {
                this.x = x;
                this.y = y;
        }

        // Necessário para acessar os campos na classe Ponto3D
        public double getX() { return x; };
        public double getY() { return y; };
}
```

```
class Ponto3D extends Ponto {
	private double z;

	public Ponto3D(double x, double y, double z) {
		super(x, y);
		this.z = z;
	}

	public double getZ() { return z; };

	// item b)
	public Ponto3D diferenca (Ponto3D p) {
		return new Ponto3D (p.getX() - this.getX(), p.getY() - this.getY(), p.getZ() -
this.getZ());
	}
	// OU ...
	public Ponto3D diferenca (double x, double y, double z) {
		return new Ponto3D (x - this.getX(), y - this.getY(), z - this.getZ());
	}
	// item c)
	public double distancia (Ponto3D p) {
		return Math.sqrt(Math.pow(p.getX() - this.getX(), 2) +
					Math.pow(p.getY() - this.getY(), 2) +
					Math.pow(p.getZ() - this.getZ(), 2));
	}
}
```

Questão 3) (3.0 pontos)
Suponha que a classe abaixo é utilizada para representar um sistema operacional mobile, como Android e iOS:

```
class Sistema extends App {
	App instalados[];
	int qtdInstalados;

	public Sistema(String nome, int memoria) {
		super(nome, memoria);
		this.instalados = new App[1000];
		qtdInstalados = 0;
	}
}
```

Do código podemos observar uma relação entre a classe Sistema e a classe App. Um App é um aplicativo e um sistema, além de conter um conjunto de aplicativos (vetor dentro da classe), é por si só um aplicativo também. Neste programa, para cada aplicativo armazenamos seu nome, a memória que ocupa (valor numérico) e seu status (se está rodando ou não). Dada estas características, faça:

a) (0,5 pto) Defina a classe App (atributos e construtor). Observe que um aplicativo sempre é criado com status *false*, ou seja, não está rodando.
b) (0,5 pto) Nesta classe App, defina métodos para consultar status, iniciar o aplicativo (tornar o status *true*) e parar o aplicativo (tornar o status *false*).
c) (1,0 pto) Na classe Sistema, insira um método chamado instalar, o qual inserirá um aplicativo neste sistema (vetor declarado na classe). Para um aplicativo ser

instalado, o Sistema precisa estar rodando. Caso não esteja, imprima a mensagem "Sistema desligado". Para simplificar, não haverá a remoção de aplicativos.

d) (1,0 pto) Também na classe Sistema, insira um método que retorne a quantidade de memória usa (soma da memória de todos os aplicativos mais a memória do sistema).

GABARITO:

```
class App {
	String nome;
	boolean rodando;
	int memoria;

	public App(String nome, int memoria) {
		this.nome = nome;
		this.memoria = memoria;
		this.rodando = false;
	}

	public void iniciar() {
		this.rodando = true;
	}

	public void parar() {
		this.rodando = false;
	}

	public boolean estaRodandao() {
		return this.rodando;
	}
}

class Sistema extends App {
	App instalados[];
	int qtdInstalados;

	public Sistema(String nome, int memoria) {
		super(nome, memoria);
		this.instalados = new App[1000];
		qtdInstalados = 0;
	}

	public void instalar (App aplic) {
		if (this.estaRodandao()) {
			this.instalados[qtdInstalados] = aplic;
			qtdInstalados++;
		}
		else
			System.out.println("Sistema desligado");
	}

	public int memoriaOcupada () {
		int tamAplics = 0;
		for (int i=0; i<qtdInstalados; i++) {
			tamAplics += this.instalados[i].memoria;
		}
		return tamAplics + this.memoria;
	}
}
```

```
// NÃO SOLICITADO NA QUESTÃO !!!
// APENAS PARA TESTE !!!
public class AP1_2019_2_Q3 {
	public static void main(String[] args) {
		App uber = new App("Uber", 150);
		App spotify = new App("Spotify", 110);
		Sistema android = new Sistema("Android", 550);
		android.instalar(uber);
		android.iniciar();
		android.instalar(uber);
		android.instalar(spotify);
		System.out.println("Memória ocupada: " + android.memoriaOcupada());
	}
}
```