Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação Orientada a Objetos
AP2 2° semestre de 2017.

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)
Suponha que foi contratado por uma construtora para implementar um sistema que estima o custo de projetos na construção civil. Considerando que as áreas dos cômodos das casas e apartamentos são sempre retangulares, têm-se a classe abaixo:

```
abstract class Comodo {
        double lado1, lado2, pe_direito;

        public Comodo (double l1, double l2, double a) {
                this.lado1 = l1; this.lado2 = l2; this.pe_direito = a;
        }

        public double perimetro() {
                return lado1 + lado2;
        }

        public double area() {
                return lado1 * lado2;
        }
}
```

A classe foi criada como abstrata pois os cômodos tem manipulação distinta e é necessário saber seus tipos. Por exemplo, por padrão da construtora, as paredes das

cozinhas sempre possuem azulejos até o teto. Os quartos, por sua vez, possuem apenas piso no chão. Sabendo disso, faça:

a) (2.0 pontos) Crie as classes Quarto e Cozinha. Nestas, crie um método *area_ceramica()*, o qual deve retornar a quantidade de cerâmica (piso + azulejo) necessária em cada tipo de cômodo.

b) (2.0 pontos) Crie, numa classe chamada Principal, os seguintes objetos e imprima a quantidade de cerâmica total requerida. (Utilize polimorfismo para resolver este item).

	Lado 1	Lado 2	Altura (Pé direito)
Quarto	3	4	3
Cozinha	2	3	3

RESPOSTA:

```
class Quarto extends Comodo {
	public Quarto(double l1, double l2, double a) {
		super(l1, l2, a);
	}
}

class Cozinha extends Comodo {
	public Cozinha(double l1, double l2, double a) {
		super(l1, l2, a);
	}

	public double area_ceramica() {
		return 2 * this.pe_direito * (this.lado1 +
this.lado2) + super.area_ceramica();
	}
}

public class AP2_2017_2_Q1 {
	public static void main(String[] args) {
		List<Comodo> apto = new ArrayList();
		apto.add(new Quarto(3, 4, 3));
		apto.add(new Cozinha(2, 3, 3));
		double quantidade_piso = 0;
		for (Comodo c : apto) {
			quantidade_piso += c.area_ceramica();
		}
		System.out.println("Área de piso necessária: " +
quantidade_piso);
	}
}
```

Questão 2) (3.0 pontos)
Implemente um programa que solicite ao usuário, via parâmtero de entrada, o nome de um arquivo texto, o índice de uma linha no arquivo (valor inteiro não negativo) e uma String qualquer.

Seu programa deverá inserir no arquivo informado uma linha, na posição indicada pelo índice informado, contendo a String informada. Caso o índice da nova linha seja maior do que a quantidade de linhas do arquivo original, linhas em branco deverão ser inseridas no arquivo antes da inserção da nova linha de modo a respeitar o posicionamento da nova linha.

Restrição: Não é permitido manter todo o conteúdo do arquivo de entrada na memória principal, pois o arquivo é tão grande que certamente levará à falta de memória e término prematuro do programa.

Sugestão: Utilize um arquivo texto auxiliar para realizar a operação de inserção e não se esqueça de copiar de volta o conteúdo do arquivo auxiliar para o arquivo cujo nome foi informado pelo usuário.

Exemplo:

Arquivo antes da inserção		Arquivo após a inserção
`Exemplo de arquivo texto contendo algumas linhas.`	Índice da nova linha: `4` Texto a ser inserido: `palavras distribuídas em`	`Exemplo de arquivo texto contendo algumas palavras distribuídas em linhas.`

RESPOSTA:

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

public class AP2_Q2_POO_2017_2{
  public static void main(String[] args) throws IOException {
    int indiceNovaLinha = Integer.parseInt(args[1]);

           try   (BufferedReader   in   =   new   BufferedReader(new
FileReader(args[0]));
                  BufferedWriter   tmp   =   new   BufferedWriter(new
FileWriter(args[0] + ".temp"))) {
      String linhaAtual;
      int indiceLinhaAtual = 1;

      while ((linhaAtual = in.readLine()) != null) {
        if (indiceLinhaAtual == indiceNovaLinha) {
          tmp.write(args[2]);
          tmp.newLine();
        }
        tmp.write(linhaAtual);
        tmp.newLine();
        indiceLinhaAtual++;
      }

      if (indiceLinhaAtual <= indiceNovaLinha) {
       for(;indiceLinhaAtual < indiceNovaLinha;indiceLinhaAtual++)
          tmp.newLine();
```

```
        tmp.write(args[2]);
        tmp.newLine();
      }
    }

    try (InputStream tmp = new FileInputStream(args[0] + ".temp");
        OutputStream out = new FileOutputStream(args[0])) {
      int b;
      while ((b = tmp.read()) != -1) out.write(b);
    }
  }
}
```

Questão 3) (3.0 pontos)
Escreva uma classe que implemente a seguinte interface utilizando encadeamento simples:

```
public interface Fila {
    public Object inicio();
    public Object fim();
    public void enfileirar(Object o);
    public void desenfileirar();
    public boolean estaVazia();
}
```

Faça uso do mecanismo de exceções.

RESPOSTA:

```
class no{
  Object info;
  no prox;

  no(Object o){
   info = o;
   prox = null;
  }

  public String toString(){ return info.toString() + "\n"; }
}


class FilaObjeto implements Fila{
  no ini, fim;

  FilaObjeto(){ ini = fim = null; }

  public boolean estaVazia(){ return (ini == null); }

  public Object inicio(){
    if (estaVazia()) return null;
    return ini.info;
  }

  public Object fim(){
    if (estaVazia()) return null;
    return fim.info;
  }

  public void enfileirar(Object o){
    no novo = new no(o);
```

```
    if(estaVazia()) ini = fim = novo;
    else{
      fim.prox = novo;
      fim = novo;
    }
  }

   public void desenfileirar(){
    if(!estaVazia()){
      no novo = ini;
      ini = ini.prox;
      if(ini == null) fim = null;
    }
  }

  public String toString(){
    if(estaVazia()) return null;
    String s = "";
    no p = ini;
    while(p != null){
      s += p.toString();
      p = p.prox;
    }
    return s;
  }
}


//SOMENTE PARA TESTAR A QUESTAO...
public class AP2_Q3_POO_2017_2{
  public static void main(String[] args){
    FilaObjeto f = new FilaObjeto();
    int i;
    for(i = 0; i < args.length; i++){
      f.enfileirar(args[i]);
      System.out.println(f);
    }
    for(i = 0; i < args.length; i++){
      f.desenfileirar();
      System.out.println(f);
    }
  }
}
```