| **Disciplina:** | Programação Orientada a Objetos - POCO4A | | |
| --- | --- | --- | --- |
| **Aluno:** | Arnald Souza | **R.A.:** | 2271923 |
| **Email:** | arnald@alunos.utfpr.edu.br | | |

**Segunda Avaliação Individual: a prova deve ser entregue em formato .PDF no link de submissão da plataforma de ensino. Boa prova!**

Questão 1) (5,0 pontos) Implemente com Programação Orientada a Objetos:

1. (1,0 ponto) Implemente 3 (três) classes. Crie 2 (dois) construtores com sobrecarga para cada classe.
2. (1,0 ponto) Forneça para cada classe 2 (dois) atributos e 2 (dois) comportamentos únicos que não estão presentes nas outras classes. Inicialize os atributos nos construtores de cada uma das Classes.
3. (1,0 ponto) Implemente um tipo Interface com um método abstrato. Cada classe deve implementar essa Interface.
4. (1,0 ponto) Implemente uma classe Abstrata com um método abstrato. Esta Classe Abstrata tem um relacionamento de Generalização com as Classes do item a).
5. (1,0 ponto) Implemente uma classe com um método "main" com uma lista de alocação dinâmica do tipo do item c), e que armazena objetos das Classes do item a). Itere polimorficamente a lista com a impressão do estado de execução de cada objeto.

| import java.util.ArrayList;  import java.util.List;  public class Principal {    public void iniciar(){  List<IUniversidade> lista = new ArrayList<IUniversidade>();    IUniversidade p = new Professor(12500.0f,"15487");  lista.add(p);    IUniversidade a = new Aluno("2271923",8.5f);  lista.add(a);    IUniversidade v = new Visitante(15,250);  lista.add(v);    for(IUniversidade b : lista)  System.out.println(b);    }      public static void main(String[] args) {  Principal principal = new Principal();  principal.iniciar();  }    }  //—----------------------------------------------  public class Professor extends Universidade implements IUniversidade{  private String nome = "Arnald";  private float salario;  private String RP;  private String endereco = "Rua Arthur, 318";        public Professor(float salario) {  this.salario = salario;  }    public Professor(float notaAvaliacao, String RP) {  this.salario = notaAvaliacao;  this.RP = RP;  }  public void setSalario(String salario) {  this.RP = RP;  }  public void notaGeral(float notaAvaliacao, float notaGeral){  notaGeral = 0.75f \* notaAvaliacao;  }    public String nomePessoa(){  return nome ;  }    public String Endereco(){  return endereco;  }  public String toString() {  return "Professor{" + "nome=" + nome + ", salario=" + salario + ", RP=" + RP + ", endereco=" + endereco + '}';  }      }  //—----------------------------------------------  public class Aluno extends Universidade implements IUniversidade {  private String nome = "Hirata";  private String RA;  private float notaProva;  private String endereco = "Rua Marques, 54";  public Aluno(String RA) {  this.RA = RA;  }  public Aluno(String RA, float notaProva) {  this.RA = RA;  this.notaProva = notaProva;  }    public void setRA(String RA){  this.RA = RA;  }    public void notaMedia(float notaAnterior, float notaProva, float media){  media = notaAnterior + notaProva / 2;  }    public String nomePessoa(){  return nome;  }    public String Endereco(){  return endereco;  }  public String toString() {  return "Aluno{" + "nome=" + nome + ", RA=" + RA + ", notaProva=" + notaProva + ", endereco=" + endereco + '}';  }    //—----------------------------------------------  public class Visitante extends Universidade implements IUniversidade{  private String nome = "Isaac";  private int idade;  private int numero;  private String endereco = "Rua dos Desenhos, 184";  public Visitante(int idade) {  this.idade = idade;  }  public Visitante(int idade, int numero) {  this.idade = idade;  this.numero = numero;  }    public void setNumero(int numero){  this.numero = numero;  }    public void verificaIdade(int idade){  if(this.idade < 15){  System.out.println("Menores de 15 anos não podem entrar");  }else{  System.out.println("Entrada Permitida");  }  }    public String nomePessoa(){  return nome;  }    public String Endereco(){  return endereco;  }  public String toString() {  return "Visitante{" + "nome=" + nome + ", idade=" + idade + ", numero=" + numero + ", endereco=" + endereco + '}';  }      }  //—----------------------------------------------  public interface IUniversidade {  public abstract String nomePessoa();  }  //—----------------------------------------------  public abstract class Universidade {  public abstract String Endereco();  } |
| --- |

Questão 2) (5,0 pontos) Observe o diagrama UML a seguir e responda as questões:

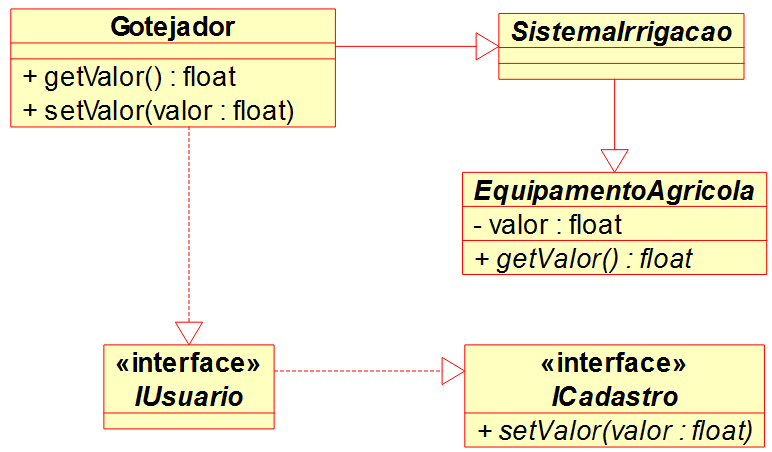


Figura 1 - Diagrama UML.

| **Nome** | **Tipo** |
| --- | --- |
| Gotejador | Classe Concreta |
| SistemaIrrigacao | Classe Abstrata |
| EquipamentoAgricola | Classe Abstrata |
| IUsuario | Interface |
| ICadastro | Interface |

Tabela 1 - Especificação do Diagrama UML da Figura 1.

1. (2,0 pontos) Implemente o diagrama UML da Figura 1 em linguagem de programação orientada a objetos.
2. (3,0 pontos) Implemente a Classe Principal que instancia um objeto da Classe Gotejador. A partir deste objeto, ilustre a invocação dos métodos acessores e mutadores deste objeto para atribuir e recuperar o conteúdo da variável de instância "valor" da classe EquipamentoAgricola, com captura de exceções verificadas personalizadas.

| import java.util.InputMismatchException;  import java.util.Scanner;//—------------------  public class Principal {        public void iniciar(){  float e = 0;  Gotejador g = new Gotejador();    try{    System.out.println("Digite o valor total dos equipamentos: ");  Scanner entrada = new Scanner(System.in);  e = entrada.nextInt();  g.setValor(e);  g.getValor();    }catch(InputMismatchException a){  System.out.println("Entrada errada de dados");  }    System.out.println("Valor do equipamento: " + g.getValor() + " reais");  }    public static void main(String[] args) {  Principal principal = new Principal();  principal.iniciar();  }    }  //—------------------  public class Gotejador extends SistemaIrrigacao implements IUsuario{    public float getValor() {  return valor;  }  public void setValor(float valor) {  this.valor = valor;  }        }  //—------------------  public abstract class SistemaIrrigacao extends EquipamentoAgricola{      }  //—------------------  public abstract class EquipamentoAgricola {    private float valor;    public float getValor() {  return valor;  }      }  //—------------------  public interface IUsuario extends ICadastro{    }  //—------------------  public interface ICadastro {  public void setValor(float valor);  } |
| --- |