1 Considere o programa abaixo escrito na linguagem Java

public class veiculo{ }

public class carro extends veiculo{ }

public class aviao extends veiculo{ }

Qual a afirmativa CORRETA ?

- **a** A classe veiculo é superclasse das classes carro e aviao.
- **b** A classe aviao é subclasse da classe carro.
- **c** As classes veículo e carro são subclasses da classe maquinas.
- **d** As classes carro e aviao são superclasses da classe veiculo.
- **e** A classe veiculo é subclasse da classe aviao.

Pontuação: 1

2 Na linguagem Java, o polimorfismo refere-se à ligação tardia de uma chamada a uma ou várias implementações diferentes de um método em uma hierarquia de herança. Neste contexto, considere as seguintes classes descritas na Linguagem Java.

```
public class POO1 {
int Calcula()
{ return 1; };
}
public class POO2 extends POO1 {
int Calcula()
{ return super.Calcula()+1; }
}
public class POO3 extends POO2 {
int Calcula()
{ return super.Calcula()+1; }
}
```

Se estas classes forem utilizadas a partir do programa a seguir

```
public class testaPOO{
    public static void  main( String args[] ){
        int Result=0;
        POO1 Objs[] = new POO1[3];
        Objs[0] = new POO1();
        Objs[1] = new POO2();
        Objs[2] = new POO3();
        for (int i=0; i<3; i++)
           Result += Objs[i].Calcula();
        System.out.println( Result );
    }
}
```

a saída desse programa será:

- **a** 6

b 5

c 9

d 3

e 0

Pontuação: 1

3 Assinale a alternativa INCORRETA acerca dos conceitos de classes em linguagem Java:

a A forma básica de herança em Java é a extensão simples entre uma superclasse e sua classe derivada. Para tanto, utiliza-se na definição da classe derivada a palavra-chave extends seguida pelo nome da superclasse.

b Uma interface é uma classe abstrata para a qual todos os métodos são públicos e todos os atributos são privados.

c Métodos abstratos de classes abstratas precisam do modificador abstract.

d Uma classe abstrata pode ser instanciada, ou seja, existem objetos que podem ser construídos diretamente de sua definição.

e Uma classe pode implementar diversas interfaces, mas pode estender apenas uma classe abstrata.

Pontuação: 1

4 Considere o código em Java abaixo:

```
import java.util.ArrayList;
public class testaProva{
    public static void main(String args[]){
        int qtd = 0;
        ArrayList <Prova> lista = new ArrayList<>();
        while(qtd <=3){
           lista.add(new Prova());
           qtd++;
        }
        System.out.println(lista);
    }
}
public class Prova{
public String toString()
{ return "prova ";};
}
```

Assinale a alternativa CORRETA acerca dos conceitos envolvidos na utilização de ArrayList na linguagem Java:

a Será impresso uma mensagem com as 4 referências dos objetos da classe Prova.

b Teremos erro de execução, pois os objetos da classe Prova não foram instanciados, em razão da classe Prova não possuir construtor.

c Será impresso a mensagem "prova" 4 vezes, separado por espaço em branco em uma única linha.

d Será impresso o endereço armazenado na variável de referência lista, algo parecido com: ArrayList@4e25154f

e Teremos uma mensagem de erro de compilação pois o ArrayList foi declarado de forma incorreta.

Pontuação: 1

5 Qual é o modificador de acesso, utilizado na linguagem Java, para definir que manteremos somente uma cópia de

determinados atributos na memória, independentemente da quantidade de objetos que forem instanciados? Ou seja, o programa deve conter apenas uma cópia de cada variável definida com esse modificador em memória, mesmo se tivermos dez objetos instanciados.

Analise as alternativas e marque a que apresente esse modificador de acesso:

a default

b protected

c public

d private

e static

Pontuação: 1

6 Considere o código em Java abaixo:

```
public class Calculadora{
  public int quadrado(int x){
    return x*x;
  }
  public double quadrado(int y){
    return y*y;
  }
}
```

Assinale a alternativa INCORRETA acerca dos conceitos envolvidos na sobrecarga de métodos na linguagem Java:

a A sobrecarga de métodos acontece na herança, quando a subclasse sobrepõe o método original.

b Os métodos com o mesmo nome podem ser declarados na mesma classe, desde que tenham conjuntos de parâmetros diferentes (determinado pelo número, tipos e ordem dos parâmetros).

c As chamadas de método não podem ser distinguidas pelo tipo de retorno do método.

d O código dos métodos **quadrado()** pode ser considerado um exemplo de polimorfismo estático ou clonagem.

e O código da classe **Calculadora** está incorreto quanto à sobrecarga do método **quadrado()**, pois resulta em erro de compilação.

Pontuação: 1

7 Considere a classe Ponto para representar uma coordenada (x,y) , a classe possui o método igual(Ponto p) que verifica se dois pontos são iguais, o método recebe um parâmetro de forma **explícita** (Ponto p) e outro de **forma implícita.**

```
public class Ponto {
   private int x,y;
   public Ponto(int x, int y) {
        this.x = x;
        this.y = y;
   }
   public boolean igual(Ponto p) {
        return this.x == p.x && this.y == p.y;
   }
}
```

A seguir temos a classe Circulo para representar a figura geométrica círculo, note que a classe Circulo foi modelada usando herança, o centro do círculo será uma coordenada (x,y) definida na classe pai e raio é um atributo float.

```
public class Circulo extends Ponto {
   private float raio;
   public Circulo(int x, int y, float raio) {
     super(x,y);
     this.raio = raio;
   }
}
```

A sua tarefa é finalizar a implementação da classe Circulo sobrescrevendo o método igual() da classe Ponto para que agora o método verifique se dois círculos são iguais, ou seja, se estão na mesma coordenada (x,y) e se tem o mesmo raio. Nesse exercício não é permitido modificar a classe Ponto para atender as necessidades da classe Circulo.

```
public class Circulo extends Ponto{
 private float raio;
 public Circulo( int x, int y, float raio){
 super(x,y);
 this.raio = raio;
 }
```

```
 @Override
 public boolean igual(Ponto p){
 Circulo c = (Circulo) p;
 return super.igual(p) && c.raio == this.raio;
}
}
```

Conceito: Certo - Pontuação: 4

Explicação:

```
public class Circulo extends Ponto {
   private float raio;
   public Circulo(int x, int y, float raio) {
     super(x,y);
     this.raio = raio;
   }
   // sobreescrita
   public boolean igual(Ponto p){
      Circulo c = (Circulo)p;
      return super.igual(p) && this.raio == c.raio;
   }
}
```

Legenda:

Alternativa correta

Resposta do aluno