



Lista 25 – Teóricos

1. Analise o seguinte código Java e escreva nas linhas a saída gerada pela execução do programa.

```
// Início do arquivo Ex1.java
package ex1;

public class Ex1 {

    public static void main(String[] args) {

        int x = 10;
        int repeticoes = 0;

        while(x > 0) {
            int y = calcula(x);
            x = y + 1;
            System.out.println(x);
            repeticoes++;
        }

        System.out.println(repeticoes + " repeticoes.");
    }

    private static int calcula(int x) {
        if(x > 0) {
            x = x - 3;
            return x;
        }
        return 0;
    }
}
// Final do arquivo Ex1.java
```

Saída:

8
6
4
2
0
5 repeticoes.

2. Analise o seguinte código Java e escreva nas linhas a saída gerada pela execução do programa.

```
// Início do arquivo Ex2.java
```

```
package ex2;
```

```
public class Ex2 {
```

```
    public static void main(String[] args) {  
        Cidade c = new Cidade(10000);  
        c.incrementaPopulacao(c.getPopulacao() / 2);  
        System.out.println(c.getPopulacao());  
        c.incrementaPopulacao(c.getPopulacao() / 5);  
        System.out.println(c.getPopulacao());  
        c.incrementaPopulacao(c.getPopulacao() / 9);  
        System.out.println(c.getPopulacao());  
    }
```

```
}
```

```
// Final do arquivo Ex2.java
```

```
// Início do arquivo Cidade.java
```

```
package ex2;
```

```
public class Cidade {
```

```
    private int populacao;
```

```
    public Cidade(int populacao) {  
        this.populacao = populacao;  
    }
```

```
    public int getPopulacao() {  
        return populacao;  
    }
```

```
    public void incrementaPopulacao(int valor) {  
        this.populacao += valor;  
    }
```

```
}
```

```
// Final do arquivo Cidade.java
```

Saída:

15000

18000

20000

3. Analise o seguinte código Java e escreva nas linhas a saída gerada pela execução do programa.

```
// Início do arquivo Ex3.java
```

```
package ex3;
```

```
public class Ex3 {  
    public static void main(String[] args) {  
        Piloto p1 = new Piloto("Senna", "F1");  
        p1.imprimir();  
        Piloto p2 = new Piloto("Valentino Rossi", "MotoGP");  
        p2.imprimir();  
    }  
}
```

```
// Final do arquivo Ex3.java
```

```
// Início do arquivo Pessoa.java
```

```
package ex3;
```

```
public class Pessoa {  
    private String nome;  
  
    public Pessoa(String nome) {  
        this.nome = nome;  
    }  
    public String getNome() {  
        return nome;  
    }  
    public void setNome(String nome) {  
        this.nome = nome;  
    }  
    public void imprimir() {  
        System.out.print(nome);  
    }  
}
```

```
// Final do arquivo Pessoa.java
```

```
// Início do arquivo Piloto.java
package ex3;

public class Piloto extends Pessoa {
    private String modalidade;

    public Piloto(String nome, String modalidade) {
        super(nome);
        this.modalidade = modalidade;
    }
    public void imprimir() {
        super.imprimir();
        System.out.println(" é da modalidade " + modalidade + ".");
    }
}
// Final do arquivo Piloto.java
```

Saída:

Senna é da modalidade F1.

Valentino Rossi é da modalidade MotoGP.

4. Analise o seguinte código e escreva nas linhas a saída gerada pela execução do programa.

```
// Início do arquivo Exercicio1.java
package exercicio1;

public class Exercicio1 {
    public static void main(String[] args) {
        Heroi c = new Heroi("Batman");
        c.atacar(1);
        c.atacar(2);
        c.atacar(1);
    }
}
// Final do arquivo Exercicio1.java

// Início do arquivo Personagem.java
package exercicio1;

public class Personagem {
    private String nome;

    public Personagem(String nome) {
        this.nome = nome;
    }
    public String getNome() {
        return this.nome;
    }
}
// Final do arquivo Personagem.java
```

```

// Início do arquivo Heroi.java
package exercicio1;

public class Heroi extends Personagem {
    private int danoAdversario = 0;
    private boolean adversarioAbatido = false;

    public Heroi(String nome) {
        super(nome);
    }

    public void atacar(int golpe) {
        switch(golpe) {
            case 1 : {
                System.out.println(getNome() + " aplica golpe " + golpe + "!");
                danoAdversario = danoAdversario + (golpe * 2);
                break;
            }
            case 2 : {
                System.out.println(getNome() + " aplica golpe " + golpe + "!");
                danoAdversario = danoAdversario + (golpe * 4);
                break;
            }
        }

        if(danoAdversario > 5) {
            if(adversarioAbatido) {
                System.out.println(getNome() + " sem nocaio!");
            } else {
                System.out.println("Oponente fora de combate!");
                adversarioAbatido = true;
            }
        }
    }
} // Final do arquivo Heroi.java

```

Saída:

Batman aplica golpe 1!
 Batman aplica golpe 2!
 Oponente fora de combate!
 Batman aplica golpe 1!
 Batman sem nocaio!

5. Qual é a diferença entre uma classe e um objeto?

Uma classe é uma estrutura que abstrai um conjunto de objetos, definindo o comportamento de seus objetos através de métodos e estados possíveis através de atributos. Um objeto é a instância de uma classe.

6. Qual é a diferença entre o modificador de acesso public e o private?

Um atributo private significa que ele só poderá ser acessado dentro do objeto no qual ele está declarado. Um atributo public pode ser acessado externamente, a partir de outros objetos.

7. Explique o mecanismo de herança.

A herança é uma forma de reutilização de software em que novas classes são criadas a partir de classes existentes, absorvendo os seus atributos e comportamentos, e adicionando novos recursos que as novas classes exigem.

8. Explique a sobrescrita de método.

Consiste em criar um novo método em uma subclasse, sendo que este método possui a mesma assinatura e o mesmo tipo de retorno do método da superclasse a ser sobrescrito. Com a sobrescrita, podemos especializar métodos herdados de superclasses, modificando o seu comportamento nas subclasses.

9. Dê um exemplo de sobrescrita de método.

10. Explique a sobrecarga de método.

A linguagem Java permite que vários métodos com o mesmo nome sejam definidos, contanto que esses métodos tenham conjuntos diferentes de parâmetros (com base na quantidade, nos tipos e na ordem dos parâmetros). Esse recurso é chamado de sobrecarga de método. Quando se chama um método sobrecarregado, será selecionado o método adequado examinando a quantidade, tipos e ordem dos argumentos na chamada.

11. Dê um exemplo de sobrecarga de método.

12. Marque V para verdadeiro e F para falsa. Justifique as afirmativas que forem falsas.

- (V) Todas variáveis devem receber um tipo quando forem declaradas.
- (F) O Java considera que as variáveis **number** e **Number** são idênticas.
- (F) O método **Integer.parseInt** converte um inteiro em uma String.
- (F) A expressão $((x > y) \ \&\& \ (a < b))$ é verdadeira se $(x > y)$ for verdadeira ou se $(a < b)$ for verdadeira.
- (V) O uso do **break** não é obrigatório na estrutura de seleção **switch**.
- (F) Supondo que **r** é um objeto de **Random**, **r.nextInt(10)** gera um inteiro aleatório entre 0 e 10.
- (F) Para construir uma interface gráfica, podemos inserir um objeto JFrame em um objeto JPanel.
- (V) Um método get pode ser usado para recuperar valores de dados **private** de um objeto.
- (F) Os membros de classe **protected** são acessíveis somente por métodos da própria classe.
- (V) Se a classe Alpha herda da classe Beta, a classe Alpha é a subclasse e Beta é a superclasse.
- (V) A herança permite a reutilização de código.
- (V) A superclasse, em geral, representa um número maior de objetos que sua classe.
- (F) A subclasse, em geral, encapsula menos funcionalidades que a sua superclasse.
- (F) A maneira correta de comparar duas Strings é por meio do operador **==**.
- (F) É possível ter uma sobrescrita de método sem uma herança.
- (V) É possível ter uma sobrecarga de método sem uma herança.