Exercício – Programação Orientada a Objetos

1) Implete as classes Veiculo e Aplicacao01, conforme os trechos de código a seguir.

```
1   public class Veiculo {
2       private String marca;
3       private String modelo;
4       private String placa;
5       private String chassi;
6       private float quilometragemAtual;
7       private float quantidadeCombustivel;
8       public final int CAPACIDADE_TANQUE = 50;
9       public final float MEDIA_CONSUMO_80 = 15;
10      public final float MEDIA_CONSUMO_100 = 13;
11      public final float MEDIA_CONSUMO_120 = 11;
12      public final float MEDIA_CONSUMO_140 = 10;
13
14      public Veiculo(String marca, String modelo ) {
15          this.marca = marca;
16          this.modelo = modelo;
17      }
18
19      public Veiculo(String marca, String modelo, String placa, String chassi ) {
20          this.marca = marca;
21          this.modelo = modelo;
22          this.placa = placa;
23          this.chassi = chassi;
24      }
25
26      public void setMarca(String marca) {
27          this.marca = marca;
28      }
29
30      public String getMarca( ) {
31          return marca;
32      }
```

```
33
34      public void setModelo(String modelo) {
35          this.modelo = modelo;
36      }
37
38      public String getModelo( ) {
39          return modelo;
40      }
41
42      public void setPlaca(String placa) {
43          this.placa = placa;
44      }
45
46      public String getPlaca( ) {
47          return placa;
48      }
49
50      public void setChassi(String chassi) {
51          this.chassi = chassi;
52      }
53
54      public String getChassi( ) {
55          return chassi;
56      }
57
58      public void zeraQuilometragem( ) {
59          quilometragemAtual = 0;
60      }
61
62      public void esvaziarTanque( ) {
63          quantidadeCombustivel = 0;
64      }
```

```
65
66      public void abastecer(float combustivel) {
67          if (combustivel > 0 && quantidadeCombustivel + combustivel < CAPACIDADE_TANQUE)
68              quantidadeCombustivel += combustivel;
69          else
70              quantidadeCombustivel = CAPACIDADE_TANQUE;
71      }
72
73      public void andou(float quilometros, float velocidade) {
74          float consumo = 0;
75
76          if(velocidade <= 80)
77              consumo = quilometros / MEDIA_CONSUMO_80;
78          else if(velocidade > 80 && velocidade <= 100)
79              consumo = quilometros / MEDIA_CONSUMO_100;
80          else if(velocidade > 100 && velocidade <= 120)
81              consumo = quilometros / MEDIA_CONSUMO_120;
82          else if(velocidade > 120)
83              consumo = quilometros / MEDIA_CONSUMO_140;
84
85          quantidadeCombustivel -= consumo;
86          quilometragemAtual += quilometros;
87      }
88
89      public float getQuilometragemAtual( ) {
90          return quilometragemAtual;
91      }
92
93      public float getQuantidadeCombustivel( ) {
94          return quantidadeCombustivel;
95      }
96
97  }
```

```
1  import javax.swing.JOptionPane;
2  public class Aplicacao01 {
3
4      public static void main (String args[ ]){
5          Veiculo vcl;
6
7          vcl = new Veiculo("Fiat","Palio");
8
9          vcl.setPlaca("XZY 7070");
10         vcl.setChassi("ZZZ234567MT26785");
11
12         vcl.zeraQuilometragem( );
13         vcl.esvaziarTanque( );
14
15         vcl.abastecer(40);
16         vcl.andou(150,140);
17
18         String resultado =  "O veiculo: "+vcl.getMarca( )+" / "+vcl.getModelo( )+
19                             "\nPlaca: "+vcl.getPlaca( )+" e chassi "+vcl.getChassi( )+
20                             " \n esta com "+vcl.getQuilometragemAtual( )+" Km "+
21                             "\n e "+ vcl.getQuantidadeCombustivel( )+ " litros de combustivel";
22
23         JOptionPane.showMessageDialog(null,resultado,"Resultado",
24                                                 JOptionPane.INFORMATION_MESSAGE);
25         System.exit(0);
26     }
27 }
```

Responda as questões

1) Sobre a definição dos atributos da classe *Veiculo*, responda as questões.

a) Qual a justificativa para alguns atributos serem declarados com modificador de escopo *private* e outros com o modificador *public*?
b) O que indica a palavra reservada *final* na definição de alguns atributos?

2) Sobre a definição de construtores da classe *Veiculo*, responda as questões.

a) Foram definidos dois construtores explícitos para a classe Veículo, quais são eles?
b) O que justifica a definição de mais de um construtor para a classe Veiculo?

3) Sobre a implementação da classe *Aplicacao1*, responda as questões.

a) Qual é a justificativa da diretiva *import*?
b) Quais são os atributos da classe Aplicacao01?
c) Em qual a linha é declarado o objeto da classe Veiculo e em qual linha ele é instanciado?
d) No contexto dessa implementação podemos dizer que a classe *Aplicacao1* é cliente da classe *Veiculo? Justifique.*

4) Analise os métodos abaixo e preencha com L – leitura, E – escrita ou P – processamento.

a (
) public void setMarca(String marca);
b ( ) public String getMarca( );
c ( ) public void setModelo(String modelo);
d ( ) public String getModelo( );
e ( ) public void zeraQuilometragem( )
f ( ) public void esvaziarTanque( )
g ( ) public void abastecer(float combustivel)
h ( ) public void andou(float quilometros, float velocidade)
i ( ) public float getQuilometragemAtual( )
j ( ) public float getQuantidadeCombustivel( )

5) Analise os atributos abaixo e preencha com L – permitem somente leitura, E – somente escrita ou LE – leitura e escrita.

a ( ) private String marca;
b ( ) private String modelo;
c ( ) private String placa;
d ( ) private String chassi;
e ( ) private float quilometragemAtual;
f ( ) private float quantidadeCombustivel;
g ( ) public final int CAPACIDADE_TANQUE = 50;

Programação Orientada a Objetos

Exercício

1) Implementar uma classe com as seguintes características:

Nome da classe: **Produto**
Atributos:

nome - String
preco – float (deve ser > 0)
margem – float (margem de lucro em % deve ser >= 0 e <=100)

Métodos:

Get's e Set's para os atributos: nome, preco e margem
calcularValorVenda() => devolve o valor de venda calculado a partir da expressão:
preco + (preco * margem / 100)

Desenvolver uma classe Aplicação com método main para testar a classe Produto:

```
Produto pro1, pro2;

pro1 = new Produto();
pro1.setNome("yyyyyyyyyyyyyyyyy");
pro1.setPreco(100);
pro1.setMargem(30);
System.out.println("nome = "+pro1.getNome());
System.out.println("preco de venda = "+pro1.calcularValorVenda());

pro2 = new Produto();
pro2.setNome("xxxxxxxxxxxxxxxx");
pro2.setPreco(200);
pro2.setMargem(20);
System.out.println("nome = "+pro2.getNome());
System.out.println("preco de venda = "+pro2.calcularValorVenda());
```

Programação Orientada a Objetos

Exercício

1) Implementar uma classe com as seguintes características:

Nome da classe: **ContaCorrente**

Atributos:

banco – int (identificador numérico para do banco)

agencia – int (identificador numérico para a agência)

conta – int (identificador numérico para a conta)

saldo – float (valor atual do saldo da conta, sem considerar o limite)

taxa – float (valor cobrado por cada operação de saque)

limite – float (limite da conta corrente, que amplia o saldo)

Métodos:

Métodos get e set para os atributos: banco, agencia, conta, taxa e limite;

Método depositar(float valor), acrescenta o valor ao saldo;

Método float sacar(float saque), caso o valor disponível seja maior ou igual ao valor do saque, cobra a taxa e atualiza o saldo. Caso contrário emite uma mensagem de falta de saldo disponível.

Método float getSaldo() devolve o saldo atual (valor sem considerar o limite);

Método float getDisponivel() devolve o valor do saldo atual acrescido do limite;

Desenvolver uma classe com método main que instancie um objeto contaCorrente e faz operações com esse objeto.