DEPARTAMENTO DE
CIÊNCIAS E TECNOLOGIAS

Paradigmas de Programação

Profa. Valéria Pequeno

Objetivos:

- Consolidar os conceitos de herança e polimorfismo (sobrescrita e sobrecarga)
- Consolidar o conceito de classe abstrata e interface
- Exercícios foram retirados/baseados nos exercícios do livro:Curso Prático de Java, Carla Jesus, FCA, 2013

Plano de aula:

Tomar como base as notas das aulas teóricas e práticas, e os conceitos aprendidos na UC de programação orientada a objetos, para resolver exercícios de revisão que exemplificam o paradigma orientado a objetos

Ferramentas usadas: NetBeans e Java 8

Não esqueça de enviar os ficheiros com a resolução dos exercícios até no máximo o dia anterior à próxima aula prática! (Em pares ou individualmente)

Exercício 1 (exercício 8 do livro):

1. Crie um projeto denominado "ExercicioP2E1" e copie para dentro dele as classes "Carro" e "Condutor" que desenvolveu no Exercício 2 (Aula prática 1):

```
//Carro.java:
public class Carro {
   private String matricula;
   private int velocidadeAtual;
   private final int velocidadeMaxima = 200;
   private Condutor condutor;
   private boolean ligado;

   public Carro(){
      this.ligado = false;
    }

    public void setMatricula(String nome){
      this.matricula = nome;
    }

   public void setCondutor(Condutor condutor){
      this.condutor = condutor;
    }

   public String getMatricula(){
      return this.matricula;  }

    public int getVelocidadeAtual(){
      return this.velocidadeAtual; }

    public int getVelocidadeMaxima(){
      return this.velocidadeMaxima;
}

    public Condutor getCondutor(){
      return this.condutor;
}
```

```
    void acelerar() {
      if (this.ligado) {
         this.velocidadeAtual += 10 + this.condutor.getDestreza()*0.1;
         if (this.velocidadeAtual > this.velocidadeMaxima)
            this.velocidadeAtual = this.velocidadeMaxima;
      } else
        System.out.println("O Carro não pode acelerar pois está desligado!");
    }

   void travar(int intensidadeTravagem) {

      if (this.ligado) {

         if (intensidadeTravagem == this.velocidadeMaxima)

            intensidadeTravagem = this.velocidadeMaxima;

        else if (intensidadeTravagem < 0)

            intensidadeTravagem = 0;

        this.velocidadeAtual -= intensidadeTravagem;

         if (this.velocidadeAtual > this.velocidadeMaxima)

           this.velocidadeAtual = this.velocidadeMaxima;

         else if (this.velocidadeAtual < 0)

           this.velocidadeAtual = 0;

      } else

         System.out.println("O Carro não pode travar pois está desligado!");  }
}
```

```
//Condutor.java:
public class Condutor {

   //Atributos

   private String nome;

   private int idade;

   private int destreza;
```

```
    public Condutor(){}
    public void setNome(String nome){
        this.nome = nome;}

    public void setIdade(int idade){
        this.idade = idade;}

    public void setDestreza(int destreza){
        this.destreza = destreza;    }

    public String getNome(){
        return this.nome;}

    public int getIdade(){
        return this.idade;}

    public int getDestreza(){
        return this.destreza;}
}
```

2. Defina os métodos da classe “Carro” como public e acrescente o seguinte método:

```
public void buzinar() {
  System.out.println(“Buzina do carro”);
}
```

3. Defina as classes “Citadino”, “Familiar” e “Jipe” dentro do projeto onde estão as classes “Carro” e “Condutor”. Estas classes são subclasses da classe

"Carro" e não possuem propriedades específicas, apenas os seguintes métodos:

Na classe "Citadino":

```
public void ligarACManual(){
  System.out.println("AC ligado!");
}
```

Na classe "Familiar":

```
public void desligarAirbagPassageiro(){
  System.out.println("Airbag desligado!");
}
```

Na classe "Jipe":

```
public void ligarTracao4x4(){
  System.out.println("Tração ligada!");
}
```

4. Desenvolva a classe "TestaBuzinar1" que deve criar uma instância de cada uma das subclasses de "Carro" e invocar o método buzinar() sobre cada uma delas. Execute a classe e analise o resultado.

5. Sobrescrever o método buzinar em cada uma das subclasses de "Carro".

6. Execute novamente a classe "TestaBuzinar1" e verifique a diferença nos resultados obtidos.

7. Acrescente a linha de código seguinte no método buzinar de cada uma das subclasses de "Carro":

```
super.buzinar();
```

8. Execute novamente a classe "TestaBuzinar1" e verifique a diferença nos resultados obtidos.

9. Altere a classe “TestaBuzina1”, de forma a que antes da invocação do método buzinar de cada subclasse de “Carro”, apareça a representação textual de cada um dos três objetos criados. Execute a classe e analise o resultado.
10. Subescreva o método toString() em cada uma das subclasses de “Carro”.
11. Execute novamente a classe “TestaBuzinar1” e verifique a diferença nos resultados obtidos.
12. Desenvolva a classe “TestaBuzinar2” que deve criar uma instância da subclasse “Carro” de acordo com a preferência manifestada pelo utilizador (“citadino”, “familiar” ou “jipe”), e invocar o método buzinar sobre essa mesma instância.
13. Execute novamente a classe “TestaBuzinar2” e verifique a diferença nos resultados obtidos.
14. Altere a classe “TestaBuzinar2” invocando o método ligarACManual, desligarAirbagPassageiro” ou ligarTracao4x4, consoante o tipo de carro criado.

Exercício 2 (Exercício 12 do livro):

1. Crie um projeto denonimado ExercicioP2E2 e copie para dentro dele as classes “Carro”, “Condutor”, “Citadino”, “Familiar”, “Jipe”, “TestaBuzinar1” e “TestaBuzinar2” do projeto “ExercicioP2E1”.
2. Defina a classe “Carro” como abstrata.
3. Execute a classe “TestaBuzinar1” e analise o resultado.
4. Compile a classe “TestaBuzinar2” e analise o resultado. Ajeite o código de modo ao programa ficar correto.
5. Defina o método “buzinar” da classe “Carro” como abstract.
6. Compile as subclasses “Citadino”, “Familiar”, e “Jipe” e analise o resultado.

Exercício 3 (Exercício 16 do livro):

1. Crie um projeto denonimado “ExercicioP2E3” e copie para dentro dele as classes do projeto “ExercicioP2E2”.
2. Defina uma interface “Buzina” com o método buzinar().
3. Na classe “Carro”, implemente a interface “Buzina”.
4. Coloque o método “buzinar() da classe “Carro” como comentário. Compile a classe e analise o resultado.
5. Coloque o método buzinar() da classe “Citadino” como comentário. Compile a classe e analise o resultado.