Uso de Herança para reutilização de código

O exemplo de projeto que utilizaremos aqui refere-se a um projeto para o armazenamento de informações de itens multimídia. A classe Item serve como superclasse para a criação de novas classes de itens específicos. No nosso projeto criaremos duas classes para o armazenamento de informações de CD e DVD, estas herdarão de Item e cada uma delas compartilhará dos atributos e métodos de Item e poderão também ter elementos particulares a si.

```
/**
 * A classe Item representa um item de multimidia.
 * Informações sobre os itens podem ser armazenadas e recuperadas.
 * Esta classe serve como superclasse para mais itens específicos.
 *
 * Versão em português do livro Programação orientada a objetos
 * com Java de Michael Kolling e David J. Barnes
 *
 * @author Simone
 * @version 24/02/2021
 */
public class Item {
    private String titulo;
    private int duracao;
    private boolean tenho;
    private String comentario;

    /**
     * Inicializa os atributos do item.
     * @param titulo O título deste item.
     * @param duracao O tempo de duração deste item.
     */
    public Item(String titulo, int duracao) {
     this.titulo = titulo;
     this.duracao = duracao;
        tenho = false;
        comentario = "";
    }

    /**
     * Configura um comentário para este item.
     * @param comentario O comentario que será adicionado.
     */
    public void setComentario(String comentario) {
        this.comentario = comentario;
    }

    /**
     * @return O comentário para este item.
     */
```

```
    public String getComentario() {
        return comentario;
    }

    /**
     * Configura uma sinalização indicando se há a posse do item.
     * @param tenho true se há a posse do item, false caso
contrário.
     */
    public void setTenho(boolean tenho) {
        this.tenho = tenho;
    }

    /**
     * @return true se há a posse de uma cópia deste item.
     */
    public boolean getTenho() {
        return tenho;
    }

    /**
     * Imprime detalhes sobre este item.
     */
    public void print() {
        System.out.print("Título: " + titulo + " (" + duracao + "
mins)");
        if(tenho) {
            System.out.println("*");
        } else {
            System.out.println();
        }
        System.out.println("    " + comentario);
    }
}
```

A classe Item provê atributos e métodos gerais para um item de multimídia. A presença de uma superclasse neste projeto se dá pelo fato de que os itens de multimídia que serão criados têm várias características comuns entre si. A não presença de uma superclasse faz com que as classes para cada tipo de item multimídia tenha muito **código duplicado** para descrever as características essenciais do item. Sendo assim, as classes de itens específicos herdarão da classe Item. Veja as classes CD e DVD.

Classe CD

```
/**
 * A classe CD representa um objeto CD. Informações sobre o CD
 * são armazenadas e podem ser recuperadas.
 *
 * Versão em português do livro Programação orientada a objetos
 * com Java de Michael Kolling e David J. Barnes
 *
 * @author Simone
 * @version 24/02/2021
 */
public class CD extends Item
{
    private String artista;
    private int numeroDeFaixas;

    /**
     * Inicializa o objeto CD.
     * @param titulo O título do CD.
     * @param artista O artista do CD.
     * @param faixas O número de faixas no CD.
     * @param duracao O tempo de duração do CD.
     */
    public CD(String titulo,String artista,int faixas,int duracao){
        super(titulo, duracao); //chamada ao construtor da
superclasse
        this.artista = artista;
        numeroDeFaixas = faixas;
    }

    /**
     * @return O artista para este CD.
     */
    public String getArtista() {
        return artista;
    }

    /**
     * @return O número de faixas no CD.
     */
    public int getNumeroDeFaixas() {
        return numeroDeFaixas;
    }
}
```

Classe DVD

```
/**
 * A classe DVD representa um objeto DVD. Informações sobre um DVD
 * são armazenadas e podem ser recuperadas. Esta classe assume que
 * somente DVDs de filmes são usados.
 *
 * Versão em português do livro Programação orientada a objetos
 * com Java de Michael Kolling e David J. Barnes
 *
 * @author Simone
 * @version 24/02/2021
 */
public class DVD extends Item {
    private String diretor;

    /**
     * Construtor para objetos da classe DVD
     * @param titulo O título para este DVD.
     * @param diretor O diretor para este DVD.
     * @param duracao O tempo de duração do filme.
     */
    public DVD(String titulo, String diretor, int duracao) {
        super(titulo, duracao);//chamada ao construtor da
superclasse
        this.diretor = diretor;
    }

    /**
     * @return O diretor para este DVD.
     */
    public String getDiretor() {
        return diretor;
    }
}
```

As classes CD e DVD herdam de Item, usando o comando extends na declaração da classe. Isso quer dizer que todos os atributos e métodos da classe Item são passados para as classes herdadas. Dessa forma, cada classe classe-filha adiciona apenas aquilo que é específico para si.

Prática

Implemente essas classes. Todas elas devem estar no mesmo projeto. Teste as classes, crie objetos dos itens de multimídia CD e DVD, veja como é simples acessar os métodos públicos herdados sem necessidade de sintaxe especial.

Classe Database

A seguir temos a classe Database, ela deve estar dentro do mesmo projeto. Ela provê uma forma de armazenar listas de itens multimídia. Observe que a classe trabalha com objetos da classe Item, isso quer dizer que ela pode trabalhar com qualquer objeto da classe Item e todos os outros que herdam dela. Portanto, é possível usar com esta classe objetos da classe CD e DVD.

```
import java.util.ArrayList;
/**
 * A classe Database provê uma forma de armazenar itens de
 * multimídia. Uma lista de objetos Item podem ser impressos.
 * Esta versão não salva os dados no disco rígido, e não há métodos
 * de busca.
 * Versão em português do livro Programação orientada a objetos
 * com Java de Michael Kolling e David J. Barnes
 * @author Simone
 * @version 24/02/2021
 */
public class Database {
    private ArrayList<Item> itens;
    /**
     * Construtor cria uma lista vazia.
     */
    public Database() {
        itens = new ArrayList<Item>();
    }
    /**
     * Adiciona um item ao database.
     * @param item O item a ser adicionado.
     */
    public void addItem(Item item) {
        itens.add(item);
    }
    /**
     * Imprime uma lista de todos os itens atualmente armazenados.
     */
    public void lista() {
        for(Item item : itens) {
            item.print();
            System.out.println();//linha vazia entre os itens
        }
    }
}
```

Prática

Depois da criação de objetos CD e DVD, crie um objeto Database e armazene elementos nele, inclua objetos de ambas as classes. Faça uso dos métodos para ver se eles funcionam como deveriam.