Interfaces e Classes Abstratas
LAB - 8

Neste laboratório você irá explorar o conceito de **interfaces** e classes **abstratas**. Ressaltamos que os exemplos deste laboratório dependem de outras listagens de código desenvolvidas nos laboratórios anteriores.

Duração prevista: 60 minutos

Exercícios

Exercício 1: Classes Abstratas

Exercício 2: Interface

Classes Abstratas

As classes do projeto Banco precisam de um padrão de comportamento que identifique que instancias dessas classes são referentes ao projeto, será criada uma entidade que irá ser a classe pai de todas as classes do projeto.

1 – Crie a classe abstrata **EntidadeBanco** conforme exemplo abaixo.

```
public abstract class EntidadeBanco {
        public abstract Long getIdentificador();
}
```

2 - Agora modifique as classes **Pessoa**, **Conta** e **Transacao** de forma que essas classes se tornem subclasse da classe **EntidadeBanco**.

OBS: Como o método **getIdentificador()** é abstrato**,** devemos sobrescrever este método em todas as subclasses de **EntidadeBanco**. Para realizar esta sobrescrita, deve ser criado um atributo chamado identificador do tipo Long juntamente com os métodos getter e setter.

```
private Long identificador;

@Override
public Long getIdentificador() {
        return identificador;
}


public void setIdentificador(Long identificador) {
        this.identificador = identificador;
}
```

Interface como um tipo

1 - Crie a Interface **Entidade** como definida abaixo:

```
public interface Entidade {
	Long getIdentificador();

}
```

2 – Modifique a classe **EntidadeBanco** de forma que implemente a interface Entidade.

```
public abstract class EntidadeBanco implements Entidade{...}
```

3 - Crie a **Interface IExtrato** como definida abaixo:

```
public interface IExtrato {

	/**
	 * Formata o movimento como string.
	*/
	public String formatar() ;

}
```

4 - Crie a classe **ExtratoTXT,** deve implementar a interface **IExtrato** e implemente o método **formartar()** conforme abaixo:

```
import java.util.Iterator;

public class ExtratoTXT implements IExtrato {

	protected Conta conta;

	public ExtratoTXT(Conta conta) {
		this.conta = conta;
	}

	public String formatar() {
		String newLine = System.getProperty("line.separator");

		String resultado = "Extrato de conta " + newLine;
		resultado += String.format("%-20.20s", "Data") + " "
		+ String.format("%7.7s", "Debito") + " "
		+ String.format("%7.7s", "Credito") + " "
		+ String.format("%15.15s", "Valor") + " "
		+ String.format("%s", "Descricao") + newLine;
		Iterator it = conta.getMovimento().iterator();
		while (it.hasNext()) {
			Transacao t = (Transacao) it.next();
			if(t.getTipoTransacao() == EnumTipoTransacao.TRANSFERENCIA){
				resultado += String.format("%-20.20s", UtilData.DDMMAAAAHHMM(t.get-
Data()))
				+ " "
				+ String.format("%7d", t.getContaDebito().getNumero())
				+ " "
				+ String.format("%7d", t.getContaCredito().getNumero())
				+ " " + String.format("%15.15s", t.getValor()) + " "
				+ String.format("%s", t.getDescricao()) + newLine;
			}
```

```
                }
                return resultado;
        }
}
```

5 - Crie a classe **ExtratoHTML** que implementa a interface **IExtrato**.

```
import java.util.Iterator;

public class ExtratoHTML implements IExtrato {

        protected Conta conta;

        public ExtratoHTML( Conta conta ) {

                this.conta = conta;
        }

        public String formatar() {

                String newLine = System.getProperty("line.separator");
                String resultado = "<html>" + newLine;
                resultado += "<head>" + newLine;
                resultado += "<title>Extrato de Conta</title>" + newLine;
                resultado += "<style type=\"text/css\">" + newLine;
                resultado += "<!--" + newLine;
                resultado += "body { font-family: Verdana, Arial,Helvetica," + "sans-serif; font-weight:
normal; font-variant: normal}" + newLine;
                resultado += ".clsIndex { }" + newLine;
                resultado += ".clsTitle { background-color: #CCCCCC;" + "text-align: center }" + new-
Line;
                resultado += "td { font-size: 9pt; font-family: Verdana, Arial," + "Helvetica, sans-serif;
background-color: #EEEEEE}" + newLine;
                resultado += "-->" + newLine;
                resultado += "</style>" + newLine;
                resultado += "</head>" + newLine;
                resultado += "<body>" + newLine;
                resultado += "<h2>Extrato de conta</h2>" + newLine;
                resultado += "<TABLE CLASS=\"clsIndex\">" + newLine;
                resultado += "<tr>" + newLine;
                resultado += "<TD CLASS=\"clsTitle\"><B>Data</B></TD>" + newLine;
                resultado += "<TD CLASS=\"clsTitle\"><B>Debito</B></TD>" + newLine;
                resultado += "<TD CLASS=\"clsTitle\"><B>Credito</B></TD>" + newLine;
                resultado += "<TD CLASS=\"clsTitle\"><B>Valor</B></TD>" + newLine;
                resultado += "<TD CLASS=\"clsTitle\"><B>Descricao</B></TD>" + newLine;
                resultado += "</tr>" + newLine;
                Iterator it = conta.getMovimento().iterator();

                while (it.hasNext()) {
                        Transacao t = (Transacao) it.next();
                        if(t.getTipoTransacao() == EnumTipoTransacao.TRANSFERENCIA){
                                resultado += "<tr>" + newLine;
                                resultado += "<TD align=\"left\">" + UtilData.DDMMAAAAHHMM(t.getDa-
ta()) + "</TD>" + newLine;
                                resultado += "<TD align=\"right\">" + t.getContaDebito().getNumero() +
"</TD>" + newLine;
                                resultado += "<TD align=\"right\">" + t.getContaCredito().getNumero() +
"</TD>" + newLine;
                                resultado += "<TD align=\"right\">" + t.getValor() + "</TD>" + newLine;
                                resultado += "<TD align=\"left\">" + t.getDescricao() + "</TD>" + new-
Line;
                                resultado += "</tr>" + newLine;
                        }
                }
```

```
                resultado += "</table>" + newLine;
                resultado += "</body>" + newLine;
                resultado += "</html>" + newLine;
                return resultado;
        }
}
```

6 - Crie a classe **ExtratoContaCorrente** para imprimir as movimentações da classe **Conta.java**.

```
public class ExtratoContaCorrente {

        public static void main(String[] args) {

                // Cria uma instância de ContaService onde está presente as operações para Objeto Conta
                ContaService operacoesConta = new ContaService();

                Conta correntista1 = new Conta("Aluno", 1001);

                Conta correntista2 = new Conta("Professor", 2002);

                // faz deposito
                operacoesConta.depositar(correntista1, 1000);

                // faz transferencia de correntista1 para correntista2 e salva em memoria a transação
                operacoesConta.transferir(correntista1, 450.00, correntista2);

                // faz transferencia de correntista1 para correntista2 e salva em memoria a transação
                operacoesConta.transferir(correntista2, 50.00, correntista1);

                //IExtrato movimento = new ExtratoTXT(correntista1);
                //System.out.println(movimento.formatar());

                IExtrato movimento1 = new ExtratoHTML(correntista1);
                System.out.println(movimento1.formatar());
        }

}
```

7 – Comente as duas últimas linhas, e logo acima descomente as duas antepenúltimas. Com isso o extrato do correntista será impresso através da classe **ExtratoTXT**.

8 - Melhore o extrato html, informe o trecho de código abaixo no método formatar. Isso permitirá que o nome do cliente, o numero da conta e a data de impressão sejam impressos no cabeçalho.

```
resultado += "Titular: " + conta.getTitular() + "  Conta: " + conta.getNumero() + " "+ newLine;
resultado += "Data de Impressão: " + UtilData.DDMMAAAAHHMM(UtilData.data()) + " "+ newLine;
```