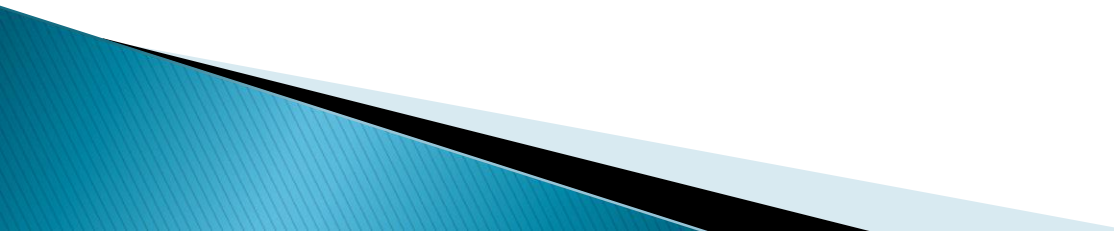


Aula 6 – Exercícios

Exercícios – Netbeans

- ▶ Crie uma pasta chamada Banco
 - Dentro desse diretório crie dois outros diretórios chamados cliente e conta
 - ▶ Cria as classes Pessoa, Conta Corrente e Conta Poupança nas respectivas pastas
 - Para cada classe adicione seus atributos e possíveis métodos
- 

Exercícios – Netbeans

- ▶ Para a classe ContaCorrente crie os atributos
 - String numAgencia;
 - String numConta;
 - Date dataCriacao;
 - float saldo;
 - float limiteChequeEspecial;
- ▶ Gere os métodos gets e sets
- ▶ Adicione também os Métodos:
 - sacar, transferirParaCorrente, transferirParaPoupanca e depositar
 - Implemente cada método

Exercícios – Netbeans

- ▶ Para a classe ContaPoupanca crie os atributos
 - String numAgencia;
 - String numConta;
 - Date dataCriacao;
 - float saldo;
 - Int variacao;
- ▶ Gere os métodos gets e sets
- ▶ Adicione também os Métodos:
 - sacar, transferirParaCorrente, transferirParaPoupanca e depositar
 - Implemente cada método

Exercício

- ▶ Para a classe Pessoa crie os atributos
 - String nome;
 - String login;
 - String senha;
 - String cpf;
 - String rg;
 - String endereco;
 - int idade;
 - ContaCorrente contaCorrente;
 - ContaPoupanca contaPoupanca;
- ▶ Gere os métodos gets e sets
- ▶ Adicione o construtor inicializando as variáveis: nome, login, senha, cpf, rg, endereco e idade

Exercício

► Construtor:

```
public Pessoa(String nome, String login, String senha,  
              String cpf, String rg, String endereco, int idade){  
  
    this.nome = nome;  
    this.login = login;  
    this.senha = senha;  
    this.cpf = cpf;  
    this.rg = rg;  
    this.endereco = endereco;  
    this.idade = idade;  
  
}
```

Exercício

```
public boolean sacar(float valor){  
    if((saldo+limiteChequeEspecial)>=valor){  
        saldo = saldo - valor;  
        return true;  
    }  
    return false;  
}
```

```
public void depositar(float valor){  
    this.saldo = this.saldo+valor;  
}
```

Exercício

```
public boolean transferirParaCorrente(float valor, ContaCorrente conta){
```

```
    if((this.saldo+limiteChequeEspecial)>=valor){
```

```
        this.saldo = this.saldo - valor;
```

```
        conta.depositar(valor);
```

```
        return true;
```

```
    }
```

```
    return false;
```

```
}
```

```
public boolean transferirParaPoupanca(float valor, ContaPoupanca conta){
```

```
    if((this.saldo+limiteChequeEspecial)>=valor){
```

```
        this.saldo = this.saldo - valor;
```

```
        conta.depositar(valor);
```

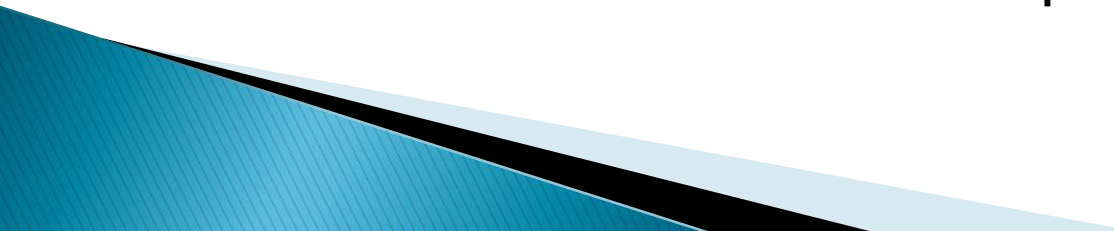
```
        return true;
```

```
    }
```

```
    return false;
```

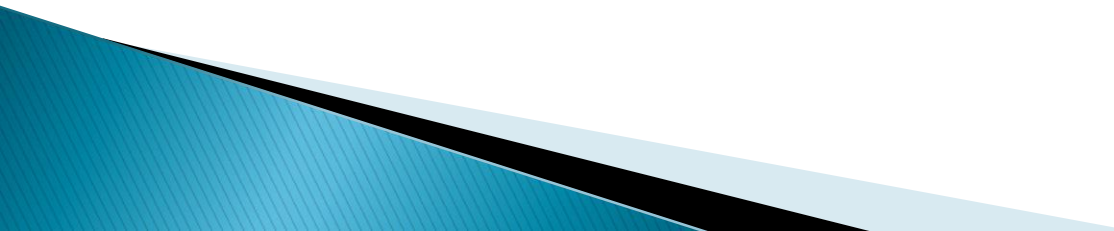
```
}
```


Exercício

- ▶ Pense em uma forma de reorganizar a estrutura de classes para permitir a reutilização de código
 - Crie a classe Conta
 - Faça com que as classes ContaCorrente e ContaPoupanca herdem de Conta
 - Transfira os atributos e métodos comuns para a classe Conta e remova-os das classes ContaCorrente e ContaPoupanca
- 

Exercício

```
public class ContaPoupanca extends Conta{  
    int variacao;  
  
    public int getVariacao() {  
        return variacao;  
    }  
  
    public void setVariacao(int variacao) {  
        this.variacao = variacao;  
    }  
}
```



Exercício

- ▶ Crie um método transferência na classe conta e não serão mais necessários os métodos de transferência nas demais classes

```
public boolean transferir(float valor, Conta conta){  
    if((this.saldo+limiteChequeEspecial)>=valor){  
        this.saldo = this.saldo - valor;  
        conta.depositar(valor);  
        return true;  
    }  
    return false;  
}
```

Exercício

- ▶ Crie uma pasta denominada view
- ▶ Crie um JFrame denominado CadastroPessoa
 - Deve permitir o cadastro de pessoas e suas contas utilizando abas

The image shows a Java Swing window titled "Visualização do Design [CadastroPessoa]". The window has a yellow title bar with standard Windows controls (minimize, maximize, close). Inside the window, there are three tabs: "Cadastrar Cliente" (selected), "Cadastrar Conta", and "Utilizar Conta". The form contains the following fields and controls:

- Nome:** A single-line text input field.
- Login:** A single-line text input field.
- Senha:** A single-line text input field with masked characters (dots).
- CPF:** A single-line text input field.
- R.G.:** A single-line text input field.
- Endereço:** A single-line text input field.
- Idade:** A single-line text input field.
- Buttons:** Two buttons at the bottom: "Salvar" and "Limpar".

Exercício

Visualização do Design [CadastroPessoa]

Cadastrar Cliente Cadastrar Conta Utilizar Conta

Número da Conta Agência

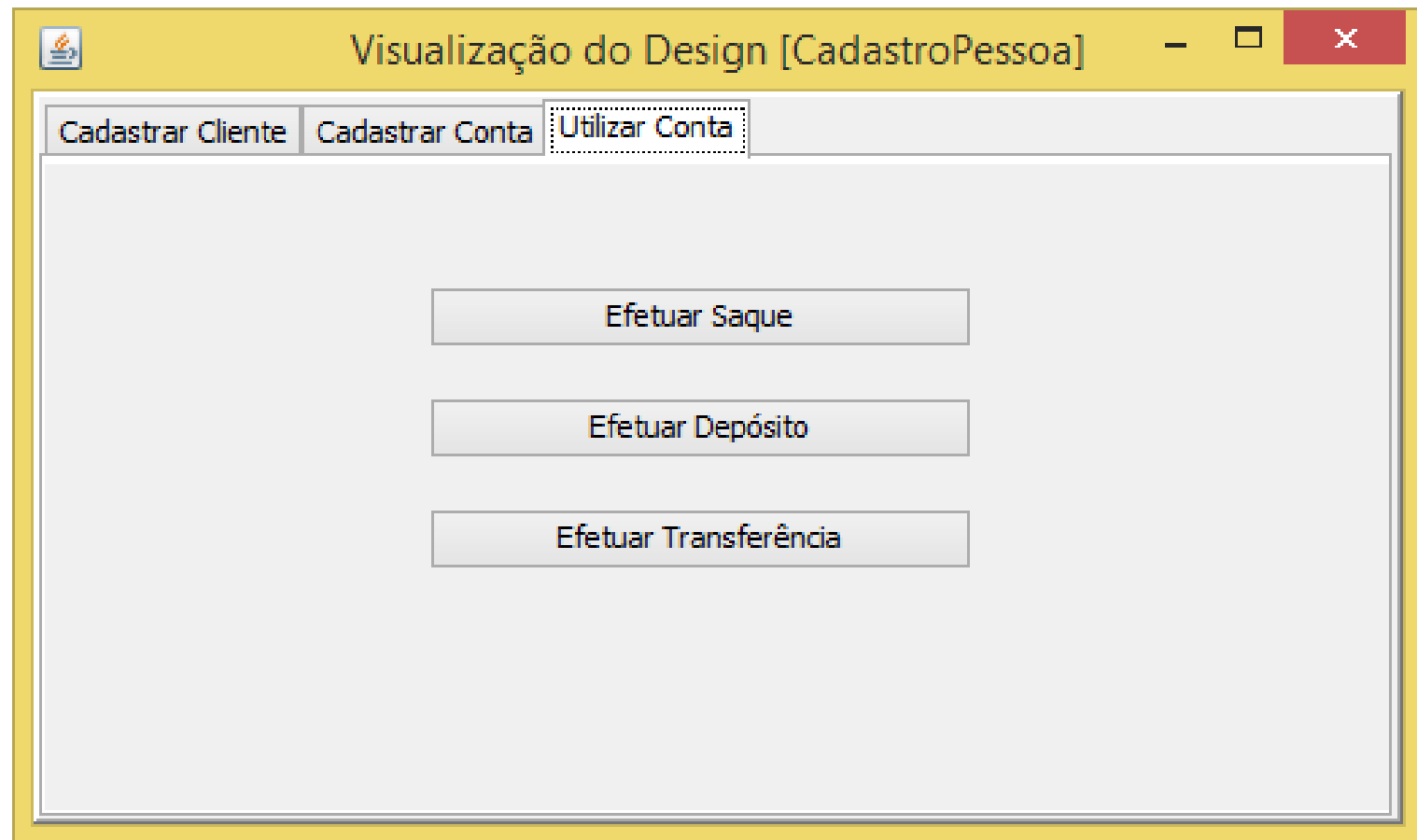
Saldo Inicial Limite do Cheque Especial

Variação

☒ Conta Corrente ☐ Conta Poupança

Salvar Limpar

Exercício



Exercício

- ▶ Para simular que já existem classes cadastradas serão criadas as classes Pessoa, Conta Corrente e Conta Poupança como variáveis globais

Pessoa pessoa1;

ContaCorrente contCorr1;

ContaPoupanca contPoup1;

Exercício

- ▶ Método Limpar

```
jTextField1.setText("");  
jTextField2.setText("");  
jTextField3.setText("");  
jTextField4.setText("");  
jTextField5.setText("");  
jTextField6.setText("");  
jPasswordField1.setText("*****");  
jTextField7.setText("");  
jTextField8.setText("");  
jTextField9.setText("");  
jTextField10.setText("");  
jTextField11.setText("");  
jRadioButton1.setSelected(true);
```


Exercício

Inserir no botão salvar do cadastro de cliente

```
String nome = jTextField1.getText();  
String login = jTextField2.getText();  
String senha = jPasswordField1.getText();  
String cpf = jTextField3.getText();  
String rg = jTextField4.getText();  
String endereco = jTextField5.getText();  
int idade = Integer.parseInt(jTextField6.getText());  
pessoa1 = new Pessoa(nome, login, senha, cpf, rg,  
                        endereco, idade);  
JOptionPane.showMessageDialog(null, "Cadastro  
realizado com sucesso");  
limpar();
```

Exercício

Inserir no botão salvar do cadastro de conta

```
if(jRadioButton1.isSelected()){
    contCorr1 = new ContaCorrente();
    contCorr1.setNumConta(jTextField7.getText());
    contCorr1.setNumAgencia(jTextField8.getText());
    contCorr1.setSaldo(Integer.parseInt(jTextField9.getText()));
    contCorr1.setLimiteChequeEspecial(Integer.parseInt(jTextField10.getText()));
    contCorr1.setDataCriacao(new Date());
    pessoa1.setContaCorrente(contCorr1);
} else{
    contPoup1 = new ContaPoupanca();
    contPoup1.setNumConta(jTextField7.getText());
    contPoup1.setNumAgencia(jTextField8.getText());
    contPoup1.setSaldo(Float.parseFloat(jTextField9.getText()));
    contPoup1.setLimiteChequeEspecial(Float.parseFloat(jTextField10.getText()));
    contPoup1.setDataCriacao(new Date());
    contPoup1.setVariacao(Integer.parseInt(jTextField11.getText()));
    pessoa1.setContaPoupanca(contPoup1);
}
JOptionPane.showMessageDialog(null, "Cadastro realizado com sucesso");
limpar();
```

Exercício

- ▶ Inserir no botão Efetuar Saque

```
String v = JOptionPane.showInputDialog("Forneça o valor a  
ser sacado");  
if(v != ""){  
    boolean sacar =  
        pessoa1.getContaCorrente().sacar(Float.parseFloat(v));  
    if(sacar){  
        float novoSaldo =  
            pessoa1.getContaCorrente().getSaldo();  
        JOptionPane.showMessageDialog(null, "Saque  
realizado com sucesso. Novo saldo:  
"+novoSaldo);  
    }else JOptionPane.showMessageDialog(null, "Saldo Insuficiente");  
}else JOptionPane.showMessageDialog(null, "Valor não fornecido");
```

Exercício

- ▶ Inserir no botão Efetuar Depósito

```
String v = JOptionPane.showInputDialog("Forneça o valor a  
ser depositado");  
if(v != ""){  
    pessoa1.getContaCorrente().depositar(Float.parseFloat(v));  
    float novoSaldo = pessoa1.getContaCorrente().getSaldo();  
    JOptionPane.showMessageDialog(null, "Deposito realizado  
com sucesso. Novo saldo: "+novoSaldo);  
}  
else JOptionPane.showMessageDialog(null, "Valor não fornecido");
```

Exercício

- ▶ Inserir no botão Efetuar Transferência

```
String v = JOptionPane.showInputDialog("Forneça o valor a ser transferido");
String agenciaDestino = JOptionPane.showInputDialog("Forneça a agência da conta
                                                    para qual deseja transferir");
String contaDestino = JOptionPane.showInputDialog("Forneça o número da conta
                                                    para qual deseja transferir");
if((v != "") && (agenciaDestino != "") && (contaDestino != "")){
    //Deveria buscar no banco a conta fornecida,
    //mas como não tem banco utiliza a conta poupança
    boolean transferencia =
    pessoa1.getContaCorrente().transferir(Float.parseFloat(v), contPoup1);
    if(transferencia){
        float novoSaldo = pessoa1.getContaCorrente().getSaldo();
        JOptionPane.showMessageDialog(null, "Transferência realizado com
                                           sucesso. Novo saldo: "+novoSaldo);
    }else JOptionPane.showMessageDialog(null, "Saldo Insuficiente");
}else JOptionPane.showMessageDialog(null, "Valor não fornecido");
```

Exercício

- ▶ Executar o sistema
 - ▶ Cadastrar um cliente
 - ▶ Cadastrar uma conta corrente e uma poupança
 - ▶ Efetuar saque, depósito e transferência
 - ▶ OBS: Como não temos tratamento de exceção então devem preencher tudo
- 