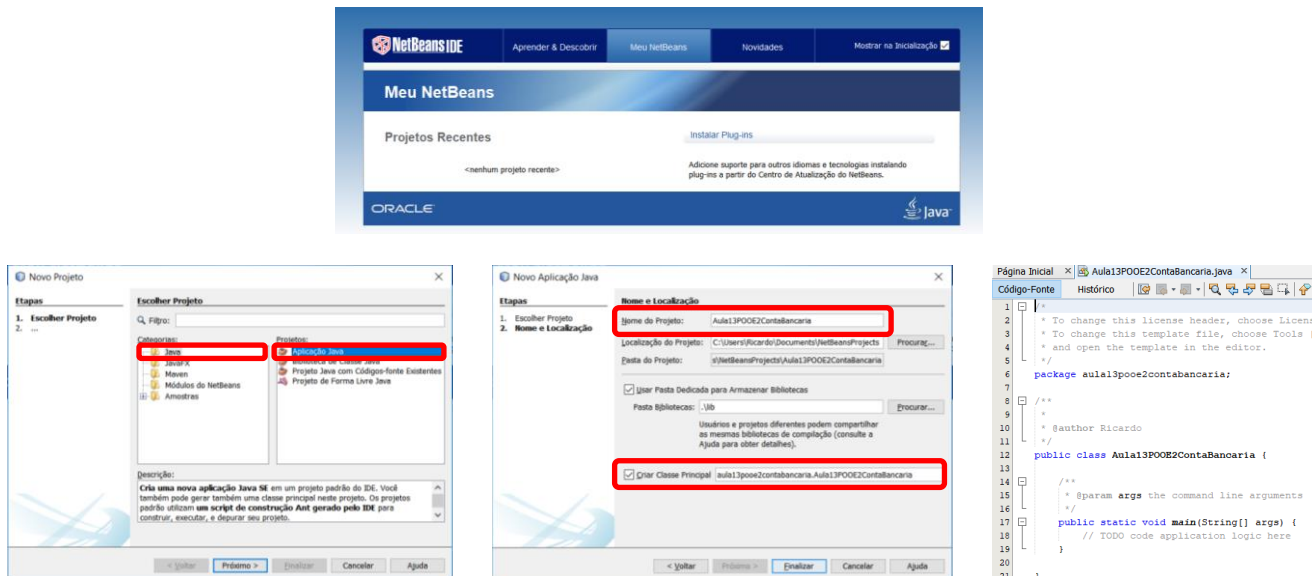


Solução do Desafio!

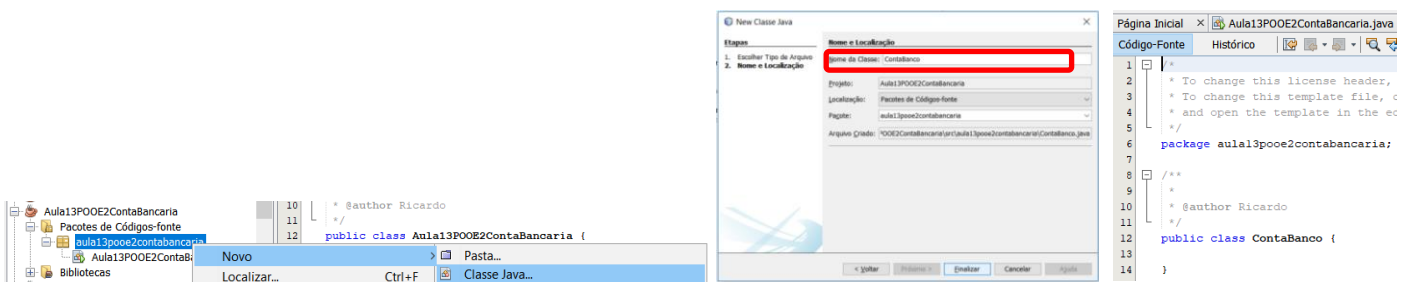
Dados da Conta Bancária

Crie um software similar aos utilizados em Bancos para gerenciamentos de Contas Correntes e Contas Poupanças.

Abra o NetBeans e inicie um Novo Projeto.



Abra uma Nova Classe.



Inicie pela criação dos Atributos.

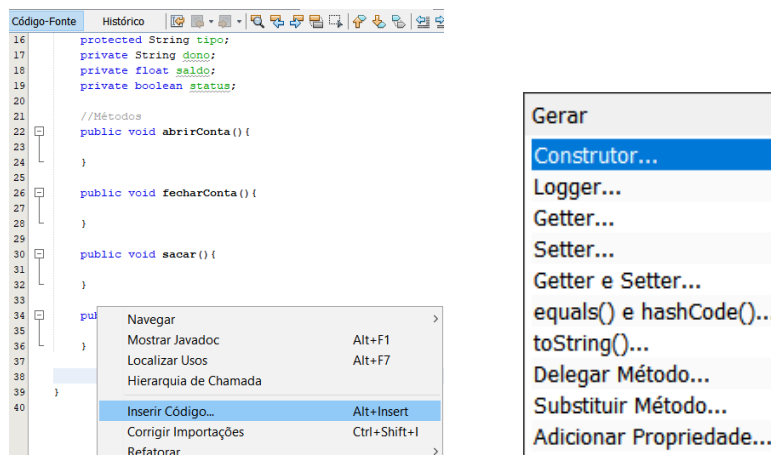
<pre> Classe ContaBanco // Atributos publico numConta: Inteiro protegido tipo: Caractere privado dono: Caractere privado saldo: Real privado status: Logico </pre>	<pre> # - - - + + + + + + </pre>	<pre> 12 public class ContaBanco { 13 14 //Atributos 15 public int numConta; 16 protected String tipo; 17 private String dono; 18 private float saldo; 19 private boolean status; 20 21 } </pre>
--	----------------------------------	--

Desenvolva os **Métodos especiais**.

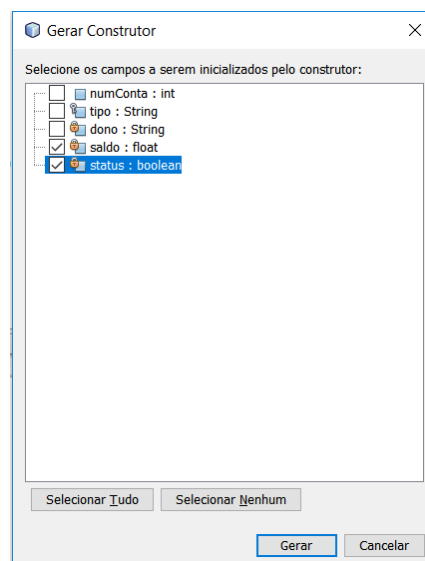
```
Classe ContaBanco
// Atributos
publico numConta: Inteiro
protegido tipo: Caractere
privado dono: Caractere
privado saldo: Real
privado status: Logico
// Métodos Especiais
publico Metodo setDono(d:Caractere)
    dono = d
FimMetodo
publico Metodo getDono()
    retorne dono
FimMetodo
publico Metodo setSaldo(s:Real)
    saldo = s
FimMetodo
publico Metodo getSaldo()
    retorne saldo
FimMetodo
publico Metodo setStatus(s:Logico)
    status = s
FimMetodo
publico Metodo getStatus()
    retorne status
FimMetodo
```

```
12 public class ContaBanco {
13
14     //Atributos
15     public int numConta;
16     protected String tipo;
17     private String dono;
18     private float saldo;
19     private boolean status;
20
21     //Métodos
22     public void abrirConta() {
23     }
24
25     public void fecharConta() {
26     }
27
28     public void sacar() {
29     }
30
31     public void pagarMensal() {
32     }
33
34 }
35
36
37
```

Crie o Construtor, clicando com o botão direito do mouse => Inserir Código => Construtor.



Selecione “saldo” e “status”.



Ainda no Construtor, preencha “ContaBanco” de modo que saldo receba 0 e status receba falso.

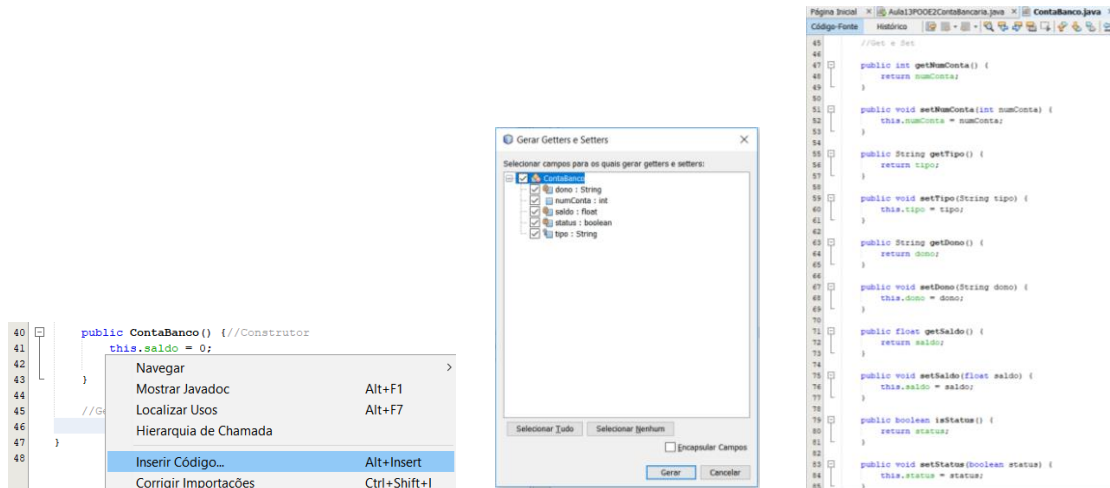
```
40 public ContaBanco(float saldo, boolean status) {
41     this.saldo = saldo;
42     this.status = status;
43 }

public Metodo Construtor()
    saldo = 0
    status = falso
FimMetodo

40 public ContaBanco() {
41     this.saldo = 0;
42     this.status = false;
43 }
```

Insira os Métodos Especiais Getter e Setter.

Clique com o botão direito do mouse => Inserir Código => Getter e Setter. Selecione tudo e clique em “Gerar”.



Desenvolva o Método necessário para abrir uma nova conta, incluindo o Atributo Tipo “String t” e fazendo o acesso através do Método, “this.setSatus(true);” e NÃO, diretamente, através do Atributo “this.status = true;”.

```
public Metodo abrirConta(t:Caractere)
    setTipo(t)
    setStatus(verdadeiro)
    se (t = "CC") entao
        saldo = 50
    senao se (t = "CP") entao
        saldo = 150
    FimSe
FimMetodo

22 public void abrirConta(String t) {
23     this.setTipo(t);
24     this.setStatus(true);
25     if (t == "CC") {
26         this.setSaldo(50);
27     } else {
28         this.setSaldo(150);
29     }
30     System.out.println("Abertura de conta realizado com sucesso");
31 }
```

Desenvolva o Método necessário para encerrar uma conta.

```
public Metodo fecharConta()
    se (saldo > 0) entao
        Escreva("Conta com dinheiro")
    senao se (saldo < 0) entao
        Escreva("Conta em débito")
    senao
        setStatus(false)
    FimSe
FimMetodo

33 public void fecharConta() {
34     if (this.getSaldo() > 0) {
35         System.out.println("Erro. Saldo em Conta");
36     } else if (this.getSaldo() < 0) {
37         System.out.println("Erro. Saldo negativo em conta.");
38     } else {
39         this.setStatus(false);
40         System.out.println("Conta encerrada com Sucesso.");
41     }
42 }
```

Desenvolva o Método necessário para depósito de dinheiro.

Acerte o Erro apresentado pela lâmpada editando o código inserido automaticamente. Isto ocorre devido a variável do "Status" ser do Tipo Boolean; desta forma, é inserido "is" ao invés de "get".

```
publico Metodo depositar(v:Real)
se (status = verdadeiro) entao
    setSaldo(getSaldo() + v)
senao
    Escreva("Impossível depositar")
FimSe
FimMetodo
```

```
44 public void depositar (float v){
45     if (this.getStatus()){
46         this.setSaldo(this.getSaldo()+v);
47         System.out.println("Depósito realizao em " + this.getDono());
48     } else {
49         System.out.println("Erro. Conta inválida.");
50     }
51 }
```

```
103 public boolean isStatus() {
104     return status;
}
```

```
102 public boolean getStatus() {
103     return status;
104 }
```

Desenvolva o Método necessário para realização de saques.

```
publico Metodo sacar(v:Real)
se (status = verdadeiro) entao
se (saldo > v) entao
    saldo = saldo - v
senao
    Escreva("Saldo insuficiente")
FimSe
senao
    Escreva("Impossível sacar")
FimSe
FimMetodo
```

```
53 public void sacar(float v){
54     if (this.getStatus()){
55         if (this.getSaldo() >= v){
56             this.setSaldo(this.getSaldo() - v);
57             System.out.println("Saque em nome de: " + this.getDono());
58         } else {
59             System.out.println("Saldo insuficiente.");
60         }
61     } else {
62         System.out.println("Erro. Conta encerrada.");
63     }
64 }
```

Desenvolva o Método necessário para pagamento de tarifa mensal.

```
publico Metodo pagarMensal()
var v: Real
se (tipo = "CC") entao
    v = 12
senao se (tipo = "CP") entao
    v = 20
fimse
se (status = verdadeiro) entao
se (saldo > v) então
    saldo = saldo - v
senao
    Escreva("Saldo insuficiente")
FimSe
senao
    Escreva("Impossível pagar")
FimSe
FimMetodo
```

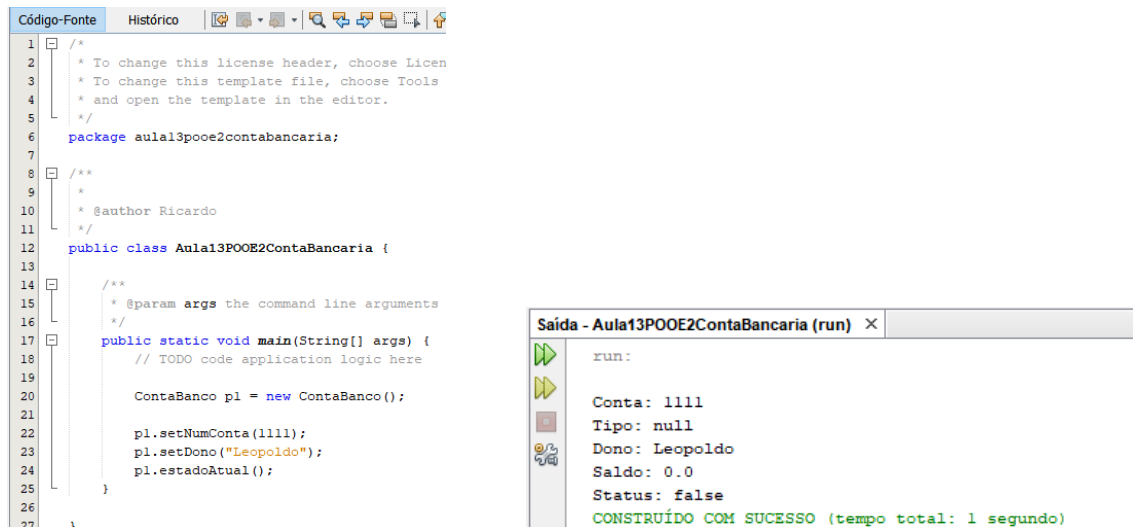
```
66 public void pagarMensal(){
67     int v = 0;
68
69     if (this.getTipo() == "CC"){
70         v = 12;
71     } else if (this.getTipo() == "CP") {
72         v = 20;
73     } else {
74         System.out.println("Erro desconhecido");
75     }
76
77     if (this.getStatus()){
78         this.setSaldo(this.getSaldo() - v);
79         System.out.println("Tarifa efetuada em nome de: " + this.dono);
80     } else {
81         System.out.println("Erro. Conta Desconhecidae.");
82     }
83 }
```

Inclua um Método para impressão de informações gerais sobre a conta.

```
84
85 public void estadoAtual(){
86     System.out.println("");
87     System.out.println("Conta: "+this.getNumConta());
88     System.out.println("Tipo: "+this.getTipo());
89     System.out.println("Dono: "+this.getDono());
90     System.out.println("Saldo: "+this.getSaldo());
91     System.out.println("Status: "+this.getStatus());
92 }
```

Neste momento, iremos adicionar os códigos da Classe main e realizar testes durante a inserção.

Teste 1. Como a conta não possui Tipo CC ou CP, ela está com saldo 0 e status false.



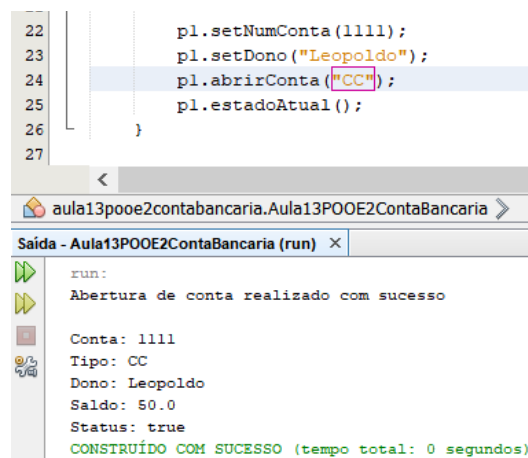
```
1  /*
2  * To change this license header, choose Licen
3  * To change this template file, choose Tools
4  * and open the template in the editor.
5  */
6  package aula13poe2contabancaria;
7
8  /**
9   *
10   * @author Ricardo
11   */
12  public class Aula13POOE2ContaBancaria {
13
14      /**
15       * @param args the command line arguments
16       */
17      public static void main(String[] args) {
18          // TODO code application logic here
19
20          ContaBanco p1 = new ContaBanco();
21
22          p1.setNumConta(1111);
23          p1.setDono("Leopoldo");
24          p1.estadoAtual();
25      }
26  }
```

Saída - Aula13POOE2ContaBancaria (run) X

run:

Conta: 1111
Tipo: null
Dono: Leopoldo
Saldo: 0.0
Status: false
CONSTRUÍDO COM SUCESSO (tempo total: 1 segundo)

Teste 2. Adicione o comando para abertura de uma CC e verifique que o saldo é alterado para “50.0” e o status para “true”.



```
22      p1.setNumConta(1111);
23      p1.setDono("Leopoldo");
24      p1.abrirConta("CC");
25      p1.estadoAtual();
26  }
```

Saída - Aula13POOE2ContaBancaria (run) X

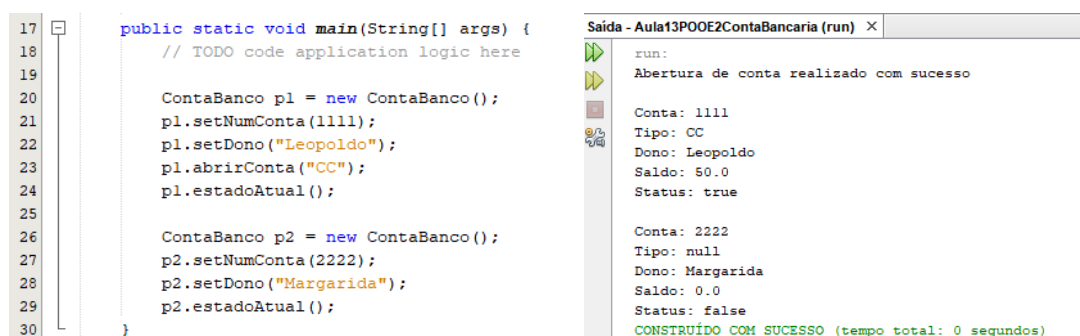
run:

Abertura de conta realizado com sucesso

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 50.0
Status: true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 3. Adicione uma nova conta.

Observe que, assim como anteriormente, como não foi aberta a conta, ainda não há um valor.



```
17      public static void main(String[] args) {
18          // TODO code application logic here
19
20          ContaBanco p1 = new ContaBanco();
21          p1.setNumConta(1111);
22          p1.setDono("Leopoldo");
23          p1.abrirConta("CC");
24          p1.estadoAtual();
25
26          ContaBanco p2 = new ContaBanco();
27          p2.setNumConta(2222);
28          p2.setDono("Margarida");
29          p2.estadoAtual();
30      }
```

Saída - Aula13POOE2ContaBancaria (run) X

run:

Abertura de conta realizado com sucesso

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 50.0
Status: true

Conta: 2222
Tipo: null
Dono: Margarida
Saldo: 0.0
Status: false
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 4. Faça com que seja aberta uma Conta Poupança (CP) para a Margarida e verifique que esta possui um saldo de “150.0” e o status está “true”.

Ou seja, para abrir a conta, é necessário informar qual o Tipo de Conta.

```
17 public static void main(String[] args) {
18     // TODO code application logic here
19
20     ContaBanco p1 = new ContaBanco();
21     p1.setNumConta(1111);
22     p1.setDono("Leopoldo");
23     p1.abrirConta("CC");
24     p1.estadoAtual();
25
26     ContaBanco p2 = new ContaBanco();
27     p2.setNumConta(2222);
28     p2.setDono("Margarida");
29     p2.abrirConta("CP");
30     p2.estadoAtual();
31 }
```

Saída - Aula13POOE2ContaBancaria (run) X

run:
Abertura de conta realizado com sucesso

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 50.0
Status: true
Abertura de conta realizado com sucesso

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 150.0
Status: true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 5. Faremos os depósitos de 100 para Leopoldo e 500 para Margarida.

```
17 public static void main(String[] args) {
18     // TODO code application logic here
19
20     ContaBanco p1 = new ContaBanco();
21     p1.setNumConta(1111);
22     p1.setDono("Leopoldo");
23     p1.abrirConta("CC");
24
25     ContaBanco p2 = new ContaBanco();
26     p2.setNumConta(2222);
27     p2.setDono("Margarida");
28     p2.abrirConta("CP");
29
30     p1.depositar(100);
31     p2.depositar(500);
32
33     p1.estadoAtual();
34     p2.estadoAtual();
35
36 }
```

Saída - Aula13POOE2ContaBancaria (run) X

run:
Abertura de conta realizado com sucesso
Abertura de conta realizado com sucesso
Depósito realizao em Leopoldo
Depósito realizao em Margarida

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 150.0
Status: true

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 650.0
Status: true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 6. Realizar um saque de “1000” da conta da Margarida.

```
17 public static void main(String[] args) {
18     // TODO code application logic here
19
20     ContaBanco p1 = new ContaBanco();
21     p1.setNumConta(1111);
22     p1.setDono("Leopoldo");
23     p1.abrirConta("CC");
24
25     ContaBanco p2 = new ContaBanco();
26     p2.setNumConta(2222);
27     p2.setDono("Margarida");
28     p2.abrirConta("CP");
29
30     p1.depositar(100);
31     p2.depositar(500);
32
33     p2.sacar(1000);
34
35     p1.estadoAtual();
36     p2.estadoAtual();
37 }
```

Saída - Aula13POOE2ContaBancaria (run) X

run:
Abertura de conta realizado com sucesso
Abertura de conta realizado com sucesso
Depósito realizao em Leopoldo
Depósito realizao em Margarida
Saldo insuficiente.

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 150.0
Status: true

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 650.0
Status: true
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 7. Realizar um saque de “100” da conta da Margarida.

```
20 ContaBanco p1 = new ContaBanco();
21 p1.setNumConta(1111);
22 p1.setDono("Leopoldo");
23 p1.abrirConta("CC");
24
25 ContaBanco p2 = new ContaBanco();
26 p2.setNumConta(2222);
27 p2.setDono("Margarida");
28 p2.abrirConta("CP");
29
30 p1.depositar(100);
31 p2.depositar(500);
32
33 p2.sacar(100);
34
35 p1.estadoAtual();
36 p2.estadoAtual();
```

Saída - Aula13POOE2ContaBancaria (run) X

run:

- Abertura de conta realizado com sucesso
- Abertura de conta realizado com sucesso
- Depósito realizao em Leopoldo
- Depósito realizao em Margarida
- Saque em nome de: Margarida

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 150.0
Status: true

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 550.0
Status: true

CONSTRUÍDO COM SUCESSO (tempo total: 0 s

Teste 8. Tente realizar o encerramento da conta do Leopoldo. Neste caso, não será possível, já que ainda há valor depositado na conta.

```
17 public static void main(String[] args) {
18     // TODO code application logic here
19
20     ContaBanco p1 = new ContaBanco();
21     p1.setNumConta(1111);
22     p1.setDono("Leopoldo");
23     p1.abrirConta("CC");
24
25     ContaBanco p2 = new ContaBanco();
26     p2.setNumConta(2222);
27     p2.setDono("Margarida");
28     p2.abrirConta("CP");
29
30     p1.depositar(100);
31     p2.depositar(500);
32
33     p2.sacar(1000);
34
35     p1.fecharConta();
36
37     p1.estadoAtual();
38     p2.estadoAtual();
```

Saída - Aula13POOE2ContaBancaria (run) X

run:

- Abertura de conta realizado com sucesso
- Abertura de conta realizado com sucesso
- Depósito realizao em Leopoldo
- Depósito realizao em Margarida
- Saldo insuficiente.
- Erro. Saldo em Conta

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 150.0
Status: true

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 650.0
Status: true

CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Teste 9. Para encerrar a conta do Leopoldo, é necessário que antes ela faça o saque do valor total em conta.

```
17 public static void main(String[] args) {
18     // TODO code application logic here
19
20     ContaBanco p1 = new ContaBanco();
21     p1.setNumConta(1111);
22     p1.setDono("Leopoldo");
23     p1.abrirConta("CC");
24
25     ContaBanco p2 = new ContaBanco();
26     p2.setNumConta(2222);
27     p2.setDono("Margarida");
28     p2.abrirConta("CP");
29
30     p1.depositar(100);
31     p2.depositar(500);
32
33     p2.sacar(1000);
34     p1.sacar(150);
35     p1.fecharConta();
36
37     p1.estadoAtual();
38     p2.estadoAtual();
```

Saída - Aula13POOE2ContaBancaria (run) X

run:

- Abertura de conta realizado com sucesso
- Abertura de conta realizado com sucesso
- Depósito realizao em Leopoldo
- Depósito realizao em Margarida
- Saldo insuficiente.
- Saque em nome de: Leopoldo
- Conta encerrada com Sucesso.

Conta: 1111
Tipo: CC
Dono: Leopoldo
Saldo: 0.0
Status: false

Conta: 2222
Tipo: CP
Dono: Margarida
Saldo: 650.0
Status: true

CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)