**Projeto BancarioHerança**

|  |
| --- |
| **Conta** |
| - saldo: double  - numero: int |
| + depositar(valor: double) : void  + sacar(valor: double): boolean  + transferir(valor: double, destino: Conta): void  + consultarSaldo(): String |

|  |
| --- |
| **ContaPoupança** |
| - reajusteMensal: double |
| +calcularRendimento(taxa: double): void  - atualizarSaldo(): void |

|  |
| --- |
| **Principal** |
|  |
| + main(args[]: String) : void |

|  |
| --- |
| **ContaCorrente** |
| - limite: double  - jurosChequeEspecial: double |
| + sacar(valor: double): boolean  + calcularJuros(taxa: double): void  + consultarSaldo(): String |

**Classe de Modelagem: Conta**

* Criar 3 construtores
  + Iniciando os atributos como 0.
  + Iniciando os atributo com valores recebidos como parâmetro
  + Iniciando o atributo numero com valor recebido por parâmetro e o saldo como zero.
* Método depositar - void
  + Recebe um valor positivo (testar se o valor é maior que 0) e adiciona ao saldo
* Método sacar - boleano
  + Recebe um valor positivo e subtrai do saldo
  + Verificar se o saldo é maior ou igual ao valor da retirada
  + Retornar true se a operação for realizada
  + Retornar false se a operação não for realizada
* Método transferir
  + Recebe dois parâmetros: o valor e a conta destino.
  + Saca o valor da conta origem e se conseguir, deposita na conta destino.
  + if (sacar(valor)) { destino.depositar(valor);}
* Método consultarSaldo
  + Exibe o saldo da conta.

**Classe de Modelagem: ContaCorrente**

* Criar 2 construtores
  + Iniciando os atributos como 0.
  + Iniciando os atributo com valores recebidos como parâmetro, passar os valores da super classe com o comando base(atributos)
* Método sacar
  + Recebe um valor positivo e subtrai do saldo+limite
  + Verificar se o saldo+limite é maior ou igual ao valor da retirada
  + Retornar true se a operação for realizada
  + Retornar false se a operação não for realizada
* Método calcularJuros
  + Recebe um valor positivo por parametro
  + Verificar se o saldo é menor do que 0.
  + Multiplica o valor negativo pela taxa de juros recebida por parâmetro, guarda o valor no atributo jurosChequeEspecial
* Método consultarSaldo
  + Exibe o saldo da conta, o limite e o juros do cheque especial

**Classe de Modelagem: ContaPoupanca**

* Criar 2 construtores
  + Iniciando os atributos como 0.
  + Iniciando os atributo com valores recebidos como parâmetro, passar os valores da super classe com o comando base(atributos)
* Método calcularRendimento
  + Recebe um valor positivo por parametro
  + Verificar se o o saldo é maior que 0.
  + Calcula o percentual recebido por parâmetro sobre o saldo e guarda no atributo reajusteMensal
  + Chama o método privado atualizarSaldo
* Método atualizarSaldo
  + Soma o rendimento mensal ao saldo

**Classe de Teste: Principal**

Instanciar dois objetos: do tipo ContaCorrente chamado cc e do tipo ContaPoupanca chamado cp.

Apresenta um menu com as opções:

1 – Conta Corrente

2 – Conta Poupança

0 - Sair

Em cc, apresentar um menu com as opções:

1 – Depositar

2 – Sacar

3 – Transferir

4 – Consultar saldo

5 – Calcular Juros Cheque Especial

0 – Sair

Realizar as chamadas aos métodos de acordo com a opção do usuário

Repetir as operações enquanto não for informada a opção zero

Em cp, apresentar um menu com as opções:

1 – Depositar

2 – Sacar

3 – Transferir

4 – Consultar saldo

5 – Calcular Reajuste Mensal

0 – Sair