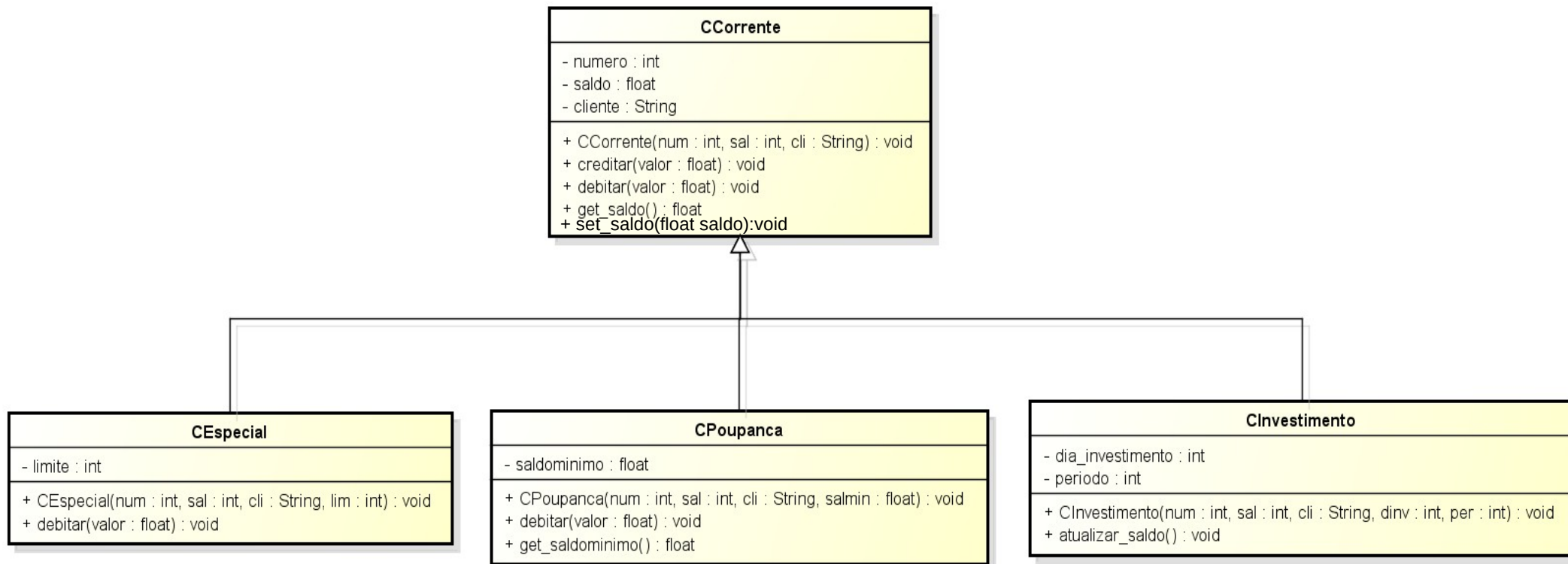


Prática de Orientação a Objetos

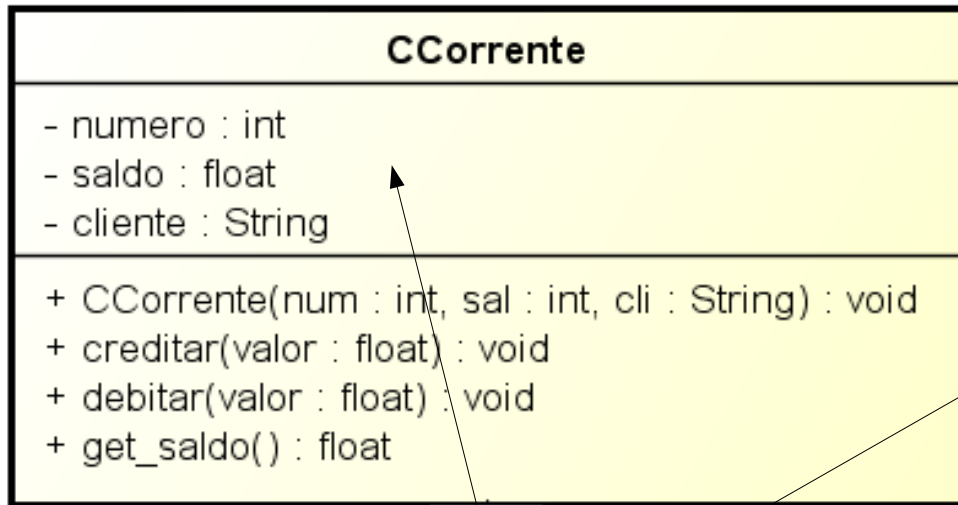
Implementando o Diagrama abaixo



Prática de Orientação a Objetos

- A Classe CCorrente é a Superclasse (classe pai) do diagrama apresentado
 - Contém três atributos: número, saldo e cliente
 - Contém quatro métodos: o construtor, creditar, debitar, get_saldo
 - O método construtor cria uma instância para a classe (um objeto) com os parâmetros fornecidos pelo usuário
 - O método creditar coloca dinheiro na conta, ou seja efetua um crédito no saldo da Conta Corrente
 - O método debitar faz um saque na conta, ou seja tira dinheiro do saldo. Lembrando que só é permitido sacar de uma conta corrente quando ela tem saldo
 - O método get_saldo apresenta o saldo atual

Implementando Conta Corrente



Esses são os atributos da classe

```
class CCorrente {
```

```
    private int numero;  
    private double saldo;  
    private String cliente;
```

Esses parâmetros vem da interface do usuário

```
    public CCorrente(int num, double sal, String cli){  
        this.numero = num;  
        this.saldo = sal;  
        this.cliente = cli;
```

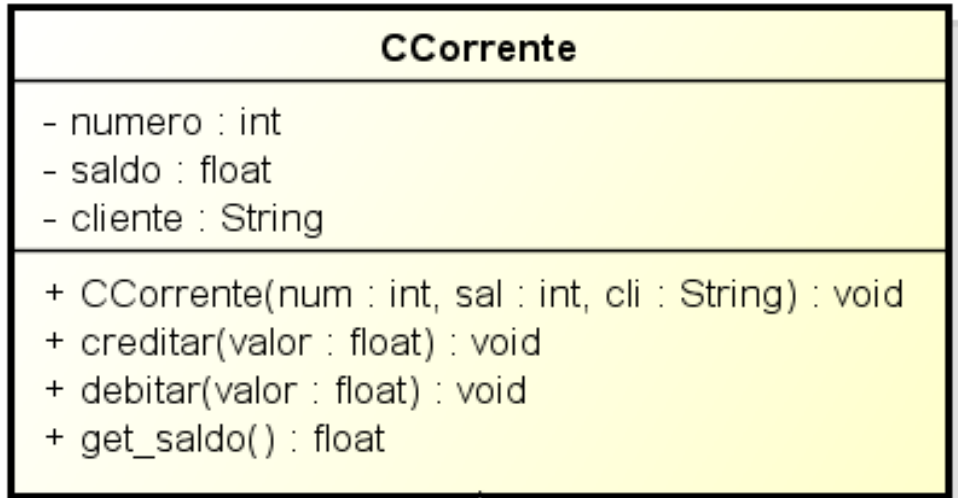
```
    }  
    public void creditar(double valor){  
        this.saldo = this.saldo + valor;
```

```
    }  
    public void debitar(double valor){  
        if (valor <= this.saldo) {  
            this.saldo = this.saldo - valor;  
        } else {  
            System.out.println("Saldo Insuficiente");  
        }  
    }
```

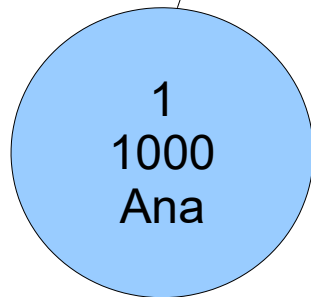
```
    public double get_saldo(){  
        return this.saldo;  
    }
```

```
    public double set_saldo(double saldo){  
        this.saldo = saldo;  
    }
```

Criando um objeto, ou seja, uma instância para essa classe



powered by Astah



CCorrente conta1 = new CCorrente(1, 1000, "Ana");

Valores passados para o construtor para a criação do objeto

```
public CCorrente (int num, double sal,String cli){  
    this.numero = num;  
    this.saldo = sal;  
    this.cliente = cli;  
}
```

Prática de Orientação a Objetos

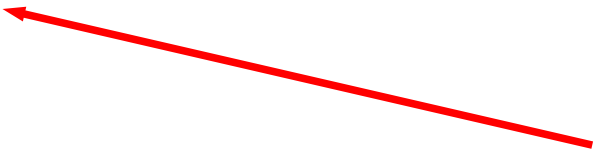
- A Classe CEspecial é Subclasse (classefilha) de Ccorrente, dessa forma pode herdar atributos e métodos já definidos:
 - Contém três atributos genéricos (herdados): número, saldo e cliente
 - Contém um atributo específico (próprio): limite
 - Contém dois métodos próprios: o construtor e debitar (reescrito)
 - O método construtor cria uma instância para a classe (um objeto) com os parâmetros fornecidos pelo usuário
 - O método debitar faz um saque na conta, ou seja tira dinheiro do saldo. Lembrando que quando não há saldo em uma conta especial é permitido retirar o valor do limite disponível

Implementando Conta Especial

```
class CEspecial extends CCorrente {  
    private int limite;
```

```
    public CEspecial(int num, double sal, String cli, int lim) {  
        super(num, sal, cli);  
        this.limite = lim;  
    }
```

**Implementação da herança,
Invocando o construtor da
superclasse CCorrente**



```
@Override
```

```
    public void debitar(double valor){  
        if (valor <=(super.get_saldo() + this.limite)){  
            super.set_saldo(super.get_saldo() - valor);  
        }
```

Implementação própria de debitar



```
        else{  
            System.out.print("Saldo Insuficiente");  
        }
```

```
    }
```

```
}
```

Testando nossa aplicação

Criando objetos

```
CCorrente conta1 = new CCorrente(1, 1000, "Ana");
```

```
CEspecial conta2 = new CEspecial(2, 2000, "Joao", 5000);
```

Testando nossa aplicação

Manipulando Conta Corrente

//Conta Corrente

```
conta1.debitar(2000);
```

```
conta1.debitar(500);
```

```
System.out.println("Seu saldo é: " + conta1.get_saldo());
```

```
conta1.creditar(1000);
```

```
System.out.println("Seu saldo é: " + conta1.get_saldo());
```

```
System.out.println("-----");
```


Testando nossa aplicação

Manipulando Conta Especial

//Conta Especial

```
conta2.debitar(10000);
```

```
conta2.debitar(3000);
```

```
System.out.println("Seu saldo é: " + conta2.get_saldo());
```

```
conta2.creditar(2000);
```

```
System.out.println("Seu saldo é: " + conta2.get_saldo());
```

```
System.out.println("-----");
```