Polimorfismo

Polimorfismo

- Para entender o polimorfismo podemos nos apegar ao significado da palavra:
  - Poli: Múltiplos
  - Morfos: Forma
- Ela faz referência ao fato de que uma variável cujo tipo seja um tipo base (utilizado como herança em outras classes) pode assumir diferentes formas (referenciar objetos de diferentes tipos)

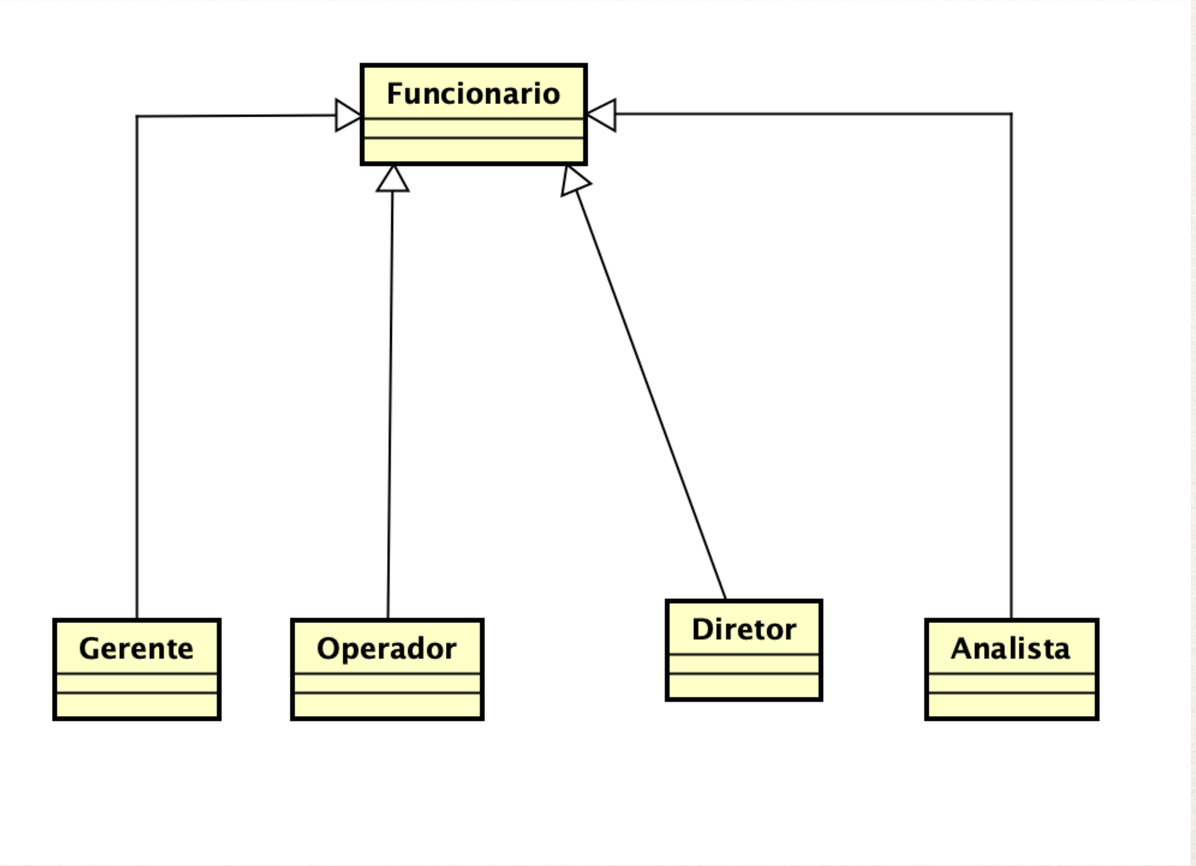
Polimorfismo

- O conceito é bastante simples, analise o seguinte exemplo:

- Em um sistema que gerencia o setor de RH de uma empresa, os funcionários são cadastrados como: Operdor, Analista, Assistente, Gerente e Diretor. Todos possuem atributos específicos da profissão e a fórmula para calcular o salário de cada um deles é diferente. Todos os meses o sistema deve, a partir de uma lista de funcionários

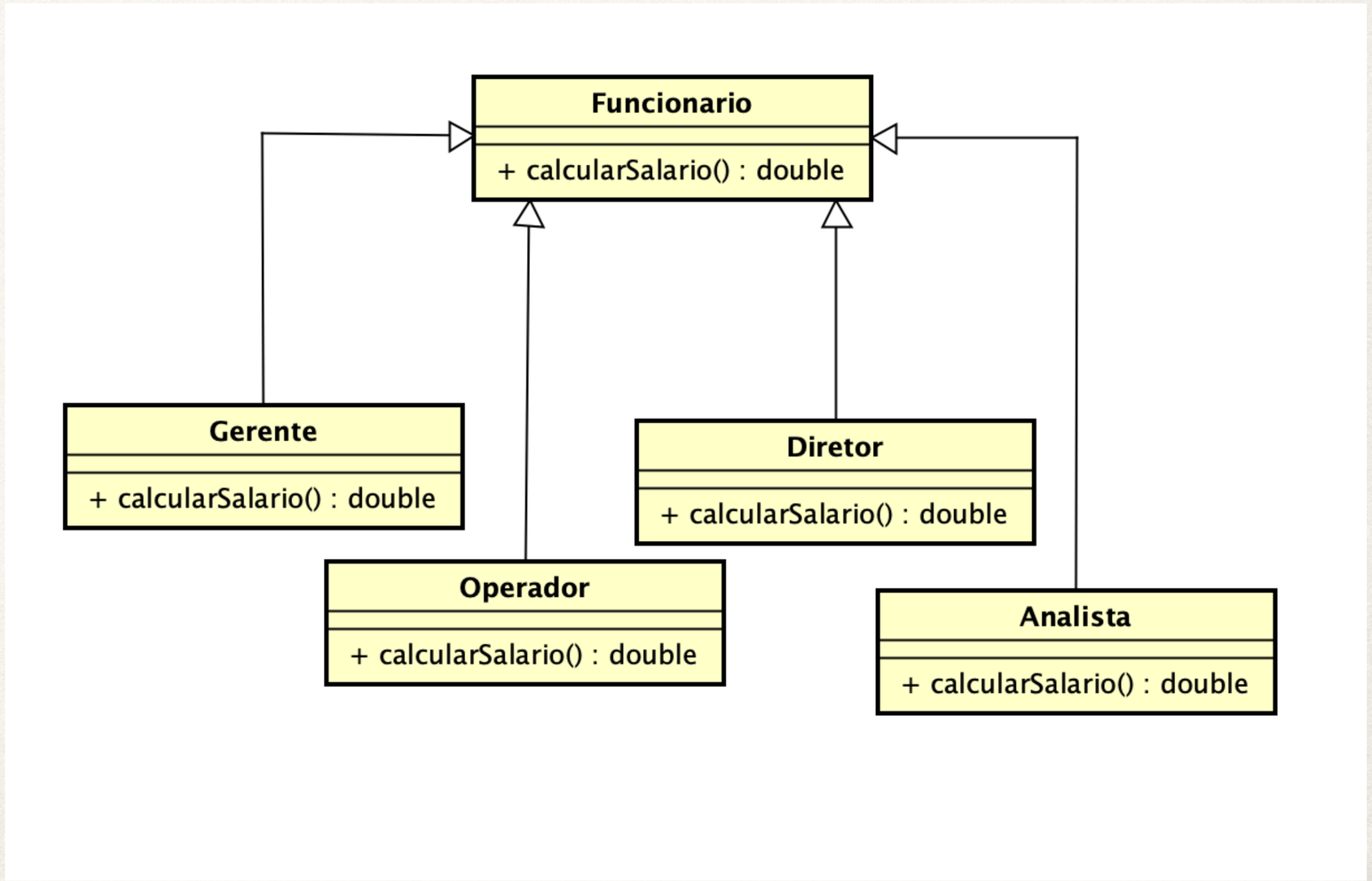
Polimorfismo

Sabemos de antemão que precisamos calcular o salário de cada um deles ao final do mês, logo precisamos de um método para isso.



Polimorfismo

O tipo base, sua superclasse Funcionario, possui um método calcularSalario, que é por sua vez sobrescrito em todas as classes que a herda, dessa forma cada uma calcula o salário.



Polimorfismo

- O polimorfismo toma forma, quando utilizamos uma referência de uma superclasse para fazer referência a objetos de uma ou mais de suas subclasses.
- O problema a ser resolvido no exemplo é:
  - Como calculamos a folha de pagamento no final do mês?
- Obviamente temos que nos certificar de chamar o método correto para cada tipo.

Polimorfismo

- Seria muito bom se pudéssemos ter uma Lista (ou vetor) com todos os funcionários, colocar em um loop e chamar o método calcularSalario de cada um na mesma estrutura.

- É exatamente isso que Polimorfismo nos permitir fazer, veja abaixo:

```
class FolhaDePagamento {

    private ArrayList<Funcionario> funcionarios = new ArrayList<Funcionario>();

    public void imprimirFolhaDePagamento() {

        for(funcionario: this.funcionarios) {
            System.out.prinln(funcionario.nome + " recebe " + funcionario.calcularSalario());
        }

    }
}
```

Polimorfismo

- Na lista de funcionários do exemplo anterior podemos colocar qualquer tipo que herde a classe Funcionario, pois Gerente, Analista, Diretor e Operador **também são** do tipo Funcionario.
- Estamos então utilizando uma referência do tipo funcionário para referenciar um outro tipo que o herda.
- E aqui a mágica acontece, pois o programa sabe disso e irá invocar o método da objeto que está sendo referenciado e não o método do tipo da referência.

Polimorfismo

- Portanto…

```
class FolhaDePagamento {

    private funcionarios = new ArrayList<Funcionario>();

    public void imprimirFolhaDePagamento() {

        for(funcionario: this.funcionarios) {
            System.out.prinln(funcionario.nome + " recebe " + funcionario.calcularSalario());
        }

    }
}
```

- Podemos colocar qualquer Funcionario na lista e o método calcularSalario chamado vai ser o método especifico de cada tipo de objeto.

Polimorfismo

- O exemplo visto em sala de aula pode ser encontrado no GitHub da disciplina.