Faculdade IMPACTA

**Programação Orientada a Objetos**

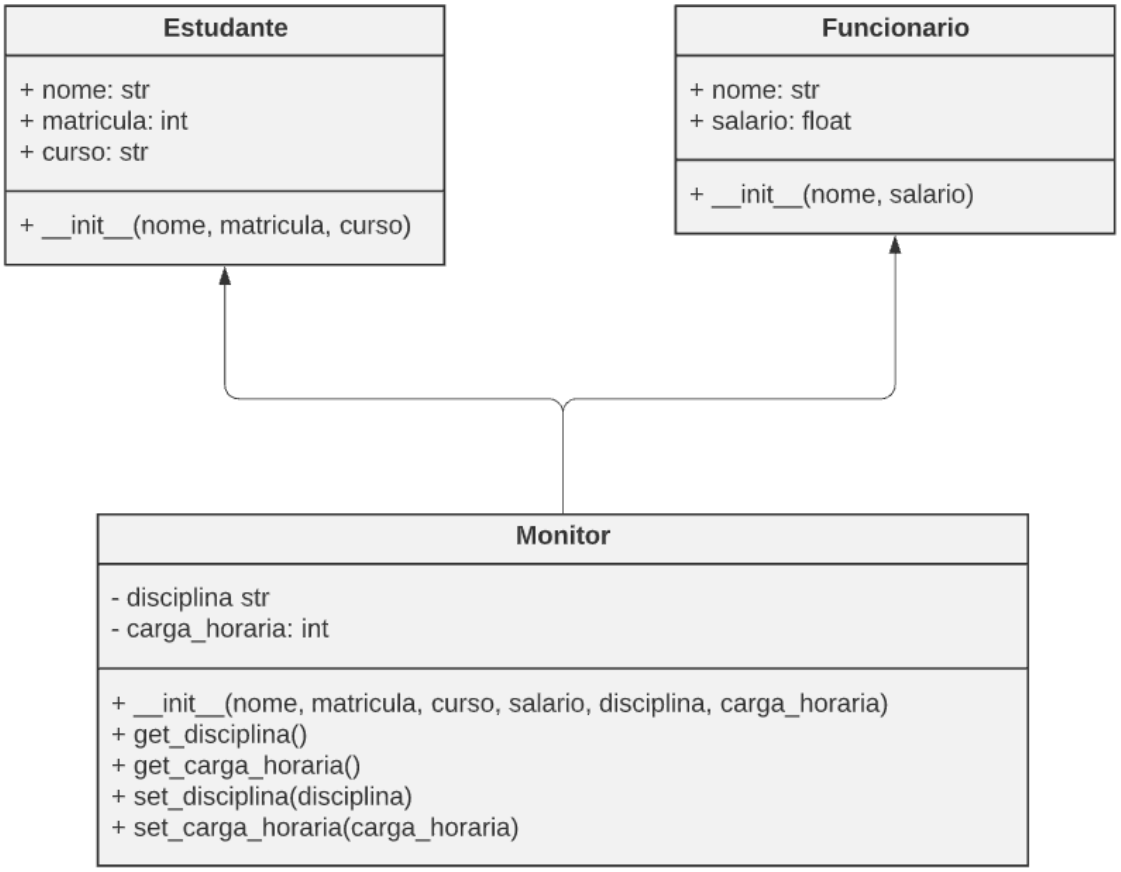
**Exercícios Complementares - Herança**

---

## Exercício 01

Implemente a hierarquia de herança múltipla definida pelo diagrama UML abaixo.

Em cada classe defina um contrutor e considere que os atributos da classe `Monitor` são privados.




*Utilize o programa abaixo para testar as classes:*

```
estudante = Estudante("Maria", 456789, "ADS")
funcionario = Funcionario("João", 2000)
monitor = Monitor("Paulo", 123456, "SI", 1000.0, "POO", 4)

print("Nome:", monitor.nome)                          # Paulo
print("Matricula:", monitor.matricula)                # 123456
print("Curso:", monitor.curso)                        # SI
print("Salario:", monitor.salario)                    # 1000.0
print("Disciplina:", monitor.get_disciplina())        # POO
print("Carga Horaria:", monitor.get_carga_horaria())  # 4
```

## Exercício 02

Uma universidade necessita de um sistema que facilite a sua gestão acadêmica. Sabe-se que um professor é um funcionário. Além de professores, há funcionários que são técnicos administrativos.

Para cada funcionário, independente de ser professor ou técnico, é necessário saber seu nome, endereço, telefone, cpf e salário.

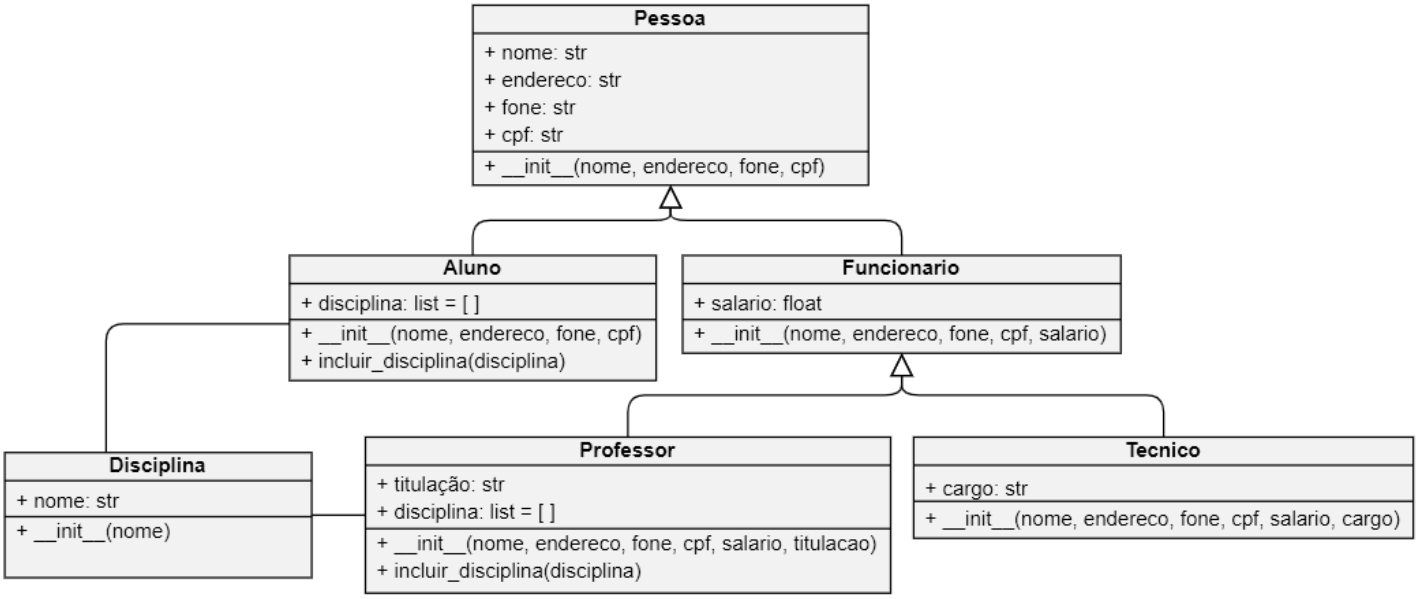
Especificamente para professores, é necessário saber sua titulação e as disciplinas que ele leciona.

Especificamente para técnicos administrativos, é necessário saber seu cargo.

Para cada aluno é necessário saber seu nome, endereco, telefone, cpf e disciplinas que ele cursa.

Para cada disciplina é necessário registrar seu nome.

Implemente a hierararquia de classes de acordo com o diagrama abaixo:




*Utilize o programa abaixo para testar as classes:*

```
disciplina1 = Disciplina("Programação")
disciplina2 = Disciplina("Banco de Dados")
professor1 = Professor("Joao", "Rua Silva, 456", "(11)99999-9555", "9999999",
                       2000, "Mestrado")
aluno1 = Aluno("Maria", "Avenida São Francisco, 239",
               "(11)98888-8435", "555555")
tecnico1 = Tecnico("Pedro", "Rua Rocha, 77",
                   "(11)93333-3333", "8787887", 1500, "Tecnico")

aluno1.incluir_disciplina(disciplina1)
aluno1.incluir_disciplina(disciplina2)
professor1.incluir_disciplina(disciplina1)

print('Disciplinas associadas ao aluno:')
for d in aluno1.disciplina:
    print(d.nome)

print('Disciplinas associadas ao Professor:')
for d in professor1.disciplina:
    print(d.nome)
```

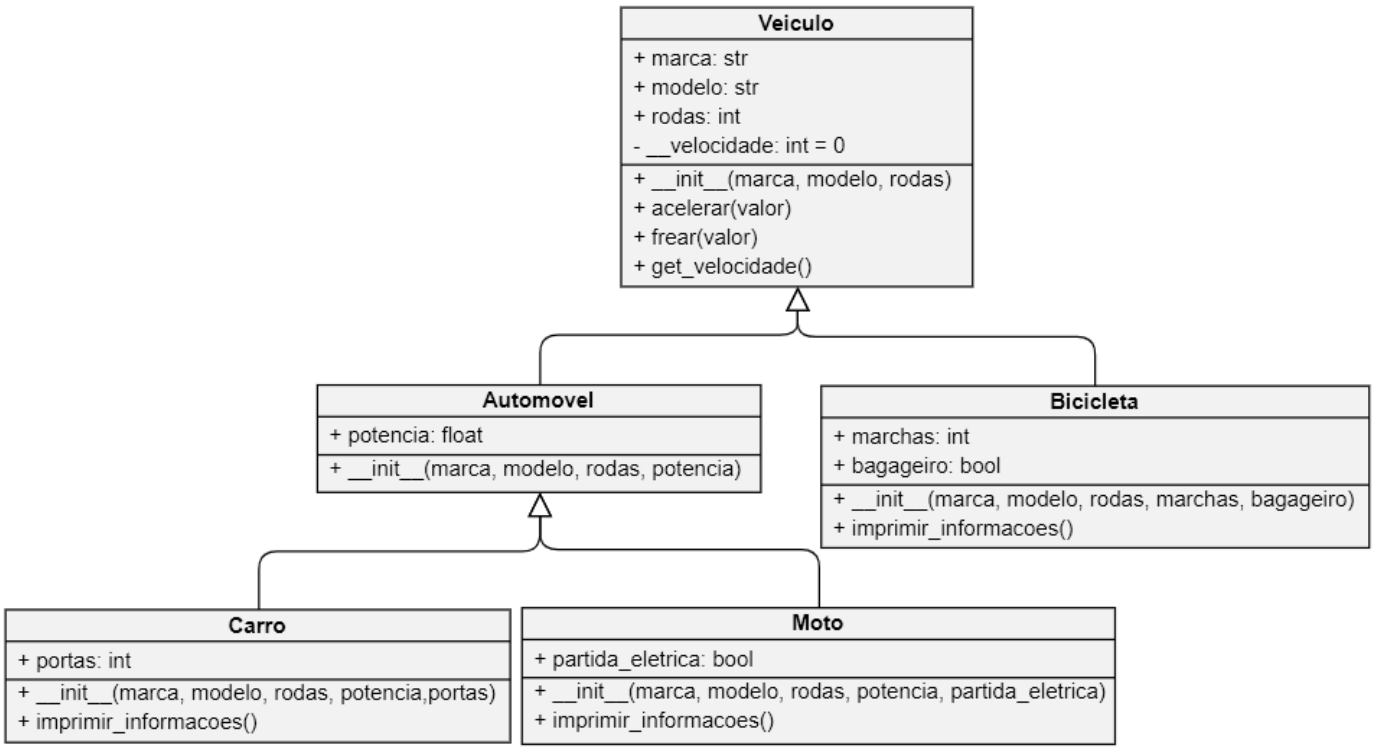
## Exercício 03

Implemente as classes de modo que obedeçam os relacionamentos apresentados no diagrama abaixo.

O método `acelerar` deve somar o valor passado por parâmetro à velocidade do veículo.

O método `frear` deve subtrair o valor passado por parâmetro da velocidade do veículo.

Os métodos `imprimir_informacoes` das classes `Carro`, `Moto` e `Bicicleta` deve exibir na tela o valor de cada um dos atributos da classe (inclusive os atributos herdados da superclasse)




*Utilize o programa abaixo para testar as classes*

```
carro = Carro("Ford", "Ka", 4, 85.0, 5)
moto = Moto("Honda", "Biz", 2, 9.2, True)
bike = Bicicleta("Caloi", "Elite", 2, 18, True)

carro.acelerar(30)
carro.frear(10)
moto.acelerar(100)
moto.frear(20)
bike.acelerar(20)
bike.frear(5)

carro.imprimir_informacoes()   # imprime os valores de todos os atributos do carro
bike.imprimir_informacoes()    # imprime os valores de todos os atributos da
bicicleta
moto.imprimir_informacoes()    # imprime os valores de todos os atributos da moto

# testar a velocidade atual
print("Velocidade atual do Carro:", carro.get_velocidade())       # 20
print("Velocidade atual da Moto:", moto.get_velocidade())         # 80
print("Velocidade atual da Bicicleta:", bike.get_velocidade())    # 15
```