Aula 02

Estrutura de Dados

<https://dontpad.com/estruturas2025>

https://meet.google.com/fqb-wmnv-ipj

Usar todas as Variáveis em inglês, e deixa o código o max possivel organizado

quando for usar uma variável booleana usar o “is” no inicio pois isso facilitará saber que ela e booleana

toda variável constante usar as letras maiusculas, as constantes nunca mundão por isso estão em letra maiuscula

usar demarcações com o ## para sinalizar o'que está ocorrendo

Codigo Usado

import time

class Elevator:

#Constants

MAX\_FLOOR = 10

MIN\_FLOOR = 0

#Atributes

currentFloor = 0

isDoorOpen = False

nextFloor = 0

#Setter methods

def set\_current\_floor(self, newFloor):

if (not(self.is\_door\_open()) and newFloor >= self.MIN\_FLOOR and newFloor <= self.MAX\_FLOOR):

self.currentFloor = newFloor

else:

print("beeeeeeppppp")

def set\_door\_state(self, newDoorState):

self.isDoorOpen = newDoorState

#getter methods

def get\_current\_floor(self):

return self.currentFloor

def is\_door\_open(self):

return self.isDoorOpen

#instancia de um objeto elevador

elevator = Elevator()

#chamada aos metodos set

elevator.set\_door\_state(False)

elevator.set\_current\_floor(8)

#chamada aos metodos get

print(elevator.is\_door\_open())

print(elevator.get\_current\_floor())

elevator.set\_current\_floor(4)

#chamada aos metodos get

print(elevator.is\_door\_open())

print(elevator.get\_current\_floor())

**Código 2**

class Elevador:

AndarAtual = 0

total\_andares = 9

def ir\_para\_andar(self, andar):

if 0 <= andar <= self.total\_andares:

if andar > self.AndarAtual:

print("\nFechando Porta")

print(f"Elevador subindo \ndo andar {self.AndarAtual} para o andar {andar}.")

print("Abrindo Porta")

elif andar < self.AndarAtual:

print("\nFechando Porta")

print(f"Elevador descendo \ndo andar {self.AndarAtual} para o andar {andar}.")

print("Abrindo Porta")

else:

print(f"O elevador está no andar {self.AndarAtual}.")

self.AndarAtual = andar

else:

print(f"O prédio só tem andares de 0 (térreo) a {self.total\_andares}.")

def status(self):

print(f"O elevador está no andar {self.AndarAtual}.")

elevador = Elevador()

while True:

print("\n################################")

print("# 1. Escolher andar #")

print("# 2. Ver andar atual #")

print("# 3. Sair #")

print("################################")

escolha = input("\nEscolha uma opção: ")

if escolha == "1":

print()

print("######################")

print("# 7 8 9 #")

print("# 4 5 6 #")

print("# 1 2 3 #")

print("# >|< 0 <|> #")

print("######################")

print()

andar = int(input(f"Precione o botao do andar que gostaria de ir (0 a {elevador.total\_andares}): "))

elevador.ir\_para\_andar(andar)

elif escolha == "2":

elevador.status()

elif escolha == "3":

print("Fim")

break

else:

print("Opção invalida. Tente novamente.")

**Exercício**

**Exercício 1:**

**python**

**Copy**

**class Car:**

**model = ""**

**car1 = Car()**

**car2 = car1**

**car3 = car2**

**car1.model = "Tesla"**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 2:**

**python**

**Copy**

**class Book:**

**title = ""**

**book1 = Book()**

**book2 = book1**

**book1 = None**

**book3 = Book()**

**book3.title = "Python Basics"**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 3:**

**python**

**Copy**

**class Animal:**

**species = ""**

**animal1 = Animal()**

**animal2 = animal1**

**animal1.species = "Lion"**

**animal2 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 4:**

**python**

**Copy**

**class Person:**

**name = ""**

**person1 = Person()**

**person2 = person1**

**person3 = person2**

**person4 = person3**

**person1 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 5:**

**python**

**Copy**

**class Laptop:**

**model = ""**

**laptop1 = Laptop()**

**laptop2 = laptop1**

**laptop3 = laptop2**

**laptop4 = laptop3**

**laptop1 = None**

**laptop2 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 6:**

**python**

**Copy**

**class Item:**

**name = ""**

**item1 = Item()**

**item2 = item1**

**item3 = item2**

**item1 = None**

**item3 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 7:**

**python**

**Copy**

**class User:**

**username = ""**

**user1 = User()**

**user2 = user1**

**user1 = None**

**user3 = user2**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 8:**

**python**

**Copy**

**class Vehicle:**

**type = ""**

**vehicle1 = Vehicle()**

**vehicle2 = vehicle1**

**vehicle3 = vehicle1**

**vehicle4 = vehicle2**

**vehicle5 = vehicle3**

**vehicle1 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 9:**

**python**

**Copy**

**class Product:**

**name = ""**

**product1 = Product()**

**product2 = product1**

**product1 = None**

**product3 = product2**

**product2 = None**

**# Quantos objetos o coletor de lixo irá recolher?**

**Exercício 10:**

**python**

**Copy**

**class Device:**

**brand = ""**

**device1 = Device()**

**device2 = device1**

**device3 = device2**

**device4 = device3**

**device5 = device4**

**device1 = None**

**device2 = None**

**# Quantos objetos o coletor de lixo irá recolher?**