



AV02 Paradigmas de Linguagem de Programação

05.10.2020

Nome do Aluno: LEONARDO AFONSO DA SILVA SOARES

Matrícula: 202009262988

Curso: REDES DE COMPUTADORES

Orientações

Façam as respostas Neste mesmo Documento

CODIFIQUE AS QUESTÕES

1. Crie uma classe em python para representação de uma FILA – FIFO
2. Crie um objeto pessoa com atributos: nome, sexo, idade e um objeto filho com nome estudante com os atributos nota e matrícula
3. Crie uma classe em python para representação de uma PILHA
4. Crie um código para instanciar um objeto em Fila com os seguintes valores 39,12,20,12,23, remova 3 valores da fila e mostre a soma dos elementos restantes
5. Adicione 6 valores em uma Pilha (10,40,20,1,3,3) remova os dois primeiros e diga qual o último elemento a sair.
6. Descreva o método de ordenação bubblesort.
7. Em uma árvore binária, o elemento central fica na esquerda, no meio ou à direita?
8. Defina o paradigma orientado a objetos
9. Para uma aplicação de fila de banco, qual a estrutura mais eficiente? A Pilha ou a Fila?

Professor Willys Campos AV01 – 23.11.2020

10. Sobre linguagens de programação e paradigma orienta a objetos é correto afirmar que: (Assinale a opção correta)



Objeto e Classe são a mesma coisa no paradigma



Classe Ocupa mais memória que objetos



A linguagem Orientada a Objeto serve para retratar o mundo real



O paradigma Estruturado deu inicio ao paradigma orientado a objetos



Polimorfismo é um objeto sem classe

1)

```
class Fila(object):  
    def __init__(self):  
        self.dados = []  
    def insere(self, elemento):  
        self.dados.append(elemento)  
    def remove(self):  
        self.dados.pop(0)  
    def lista(self):  
        return self.dados  
    def insere_com_prioridade(self, elemento, posicao):  
        self.dados.insert(posicao, elemento)
```

Professor Willys Campos AV01 – 23.11.2020

2)

peessoa.py

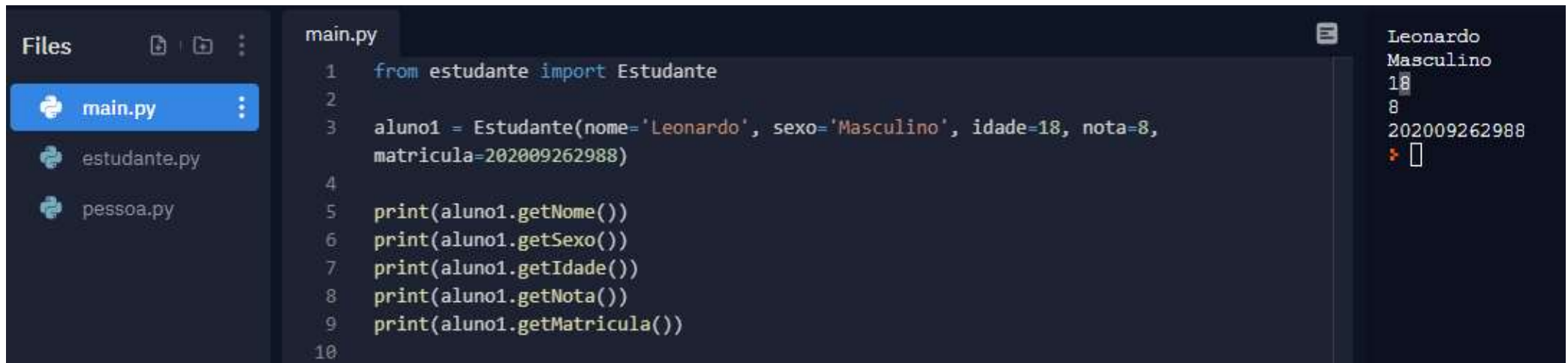
```
1 class Pessoa:
2     def __init__(self, nome, sexo, idade):
3         self.nome = nome
4         self.sexo = sexo
5         self.idade = idade
6
7     def setNome(self, nome):
8         self.nome = nome
9
10    def setSexo(self, sexo):
11        self.sexo = sexo
12
13    def setIdade(self, idade):
14        self.idade = idade
15
16    def getNome(self):
17        return self.nome
18
19    def getSexo(self):
20        return self.sexo
21
22    def getIdade(self):
23        return self.idade
```

estudante.py

```
1 from pessoa import Pessoa
2
3 class Estudante(Pessoa):
4     def __init__(self, nota, matricula, nome, sexo, idade):
5         super().__init__(nome, sexo, idade)
6         self.nota = nota
7         self.matricula = matricula
8
9     def setNota(self, nota):
10        self.nota = nota
11
12    def setMatricula(self, matricula):
13        self.matricula = matricula
14
15    def getNota(self):
16        return self.nota
17
18    def getMatricula(self):
19        return self.matricula
20
```

Professor Willys Campos AV01 – 23.11.2020

2)Teste



The screenshot shows a Python IDE with a file explorer on the left, a code editor in the center, and a console on the right. The file explorer shows three files: `main.py` (selected), `estudante.py`, and `pessoa.py`. The code editor shows the following Python code in `main.py`:

```
1 from estudante import Estudante
2
3 aluno1 = Estudante(nome='Leonardo', sexo='Masculino', idade=18, nota=8,
4 matricula=202009262988)
5
6 print(aluno1.getNome())
7 print(aluno1.getSexo())
8 print(aluno1.getIdade())
9 print(aluno1.getNota())
10 print(aluno1.getMatricula())
```

The console on the right displays the output of the code:

```
Leonardo
Masculino
18
8
202009262988
[]
```

3)

```
class Pilha(object):  
    def __init__(self):  
        self.dados = []  
    def empilha(self, elemento):  
        self.dados.append(elemento)  
    def desempilha(self):  
        self.dados.pop()  
    def lista(self):  
        return self.dados
```


Professor Willys Campos AV01 – 23.11.2020

4)

```
import fila as lObjeto

NovaFila = lObjeto.Fila()

NovaFila.insere(39)
NovaFila.insere(12)
NovaFila.insere(20)
NovaFila.insere(12)
NovaFila.insere(23)

print(NovaFila.lista())

NovaFila.remove()
NovaFila.remove()
NovaFila.remove()

print(NovaFila.lista())

soma = 0
for x in NovaFila.lista():
    soma = soma + x

print(soma)
```

[39, 12, 20, 12, 23]
[12, 23]
35
[]

Professor Willys Campos AV01 – 23.11.2020

5)

```
py
import pilha as lobjeto

NovaPilha = lobjeto.Pilha()
#10,40,20,1,3,3

NovaPilha.empilha(10)
NovaPilha.empilha(40)
NovaPilha.empilha(20)
NovaPilha.empilha(1)
NovaPilha.empilha(3)
NovaPilha.empilha(3)

print(NovaPilha.lista())

NovaPilha.desempilha()
NovaPilha.desempilha()

print(NovaPilha.lista())

ultimo ()

print(NovaPilha.lista())
```

[10, 40, 20, 1, 3, 3]
[20, 1, 3, 3]
[3]
[]

Professor Willys Campos AV01 – 23.11.2020

6) O Bubble Sort vai ordenando de par em par. Ele pega os dois primeiros elementos e pergunta se o primeiro é maior que o segundo. Se sim, os elementos são trocados (swap), se não, são mantidos. Vai repetindo o processo até o final do vetor. Obviamente que ele não consegue ordenar todo o vetor em uma única rodada, ele terá que passar pelo vetor um certo número de vezes.

De maneira mais formal podemos destacar:

1. Percorra o vetor inteiro comparando elementos adjacentes (dois a dois)
2. Troque as posições dos elementos se eles estiverem fora de ordem
3. Repita os dois passos acima ($n - 1$) vezes, onde n é igual ao tamanho do vetor

Tem-se o seguinte vetor: **5, 3, 2, 4, 7, 1, 0, 6**

Sabemos que iremos repetir o vetor $n - 1$ vezes. O tamanho do vetor é 8, logo iremos repetir 7 vezes o vetor (8-1).

Então, na primeira iteração, pegamos os dois primeiros valores e trocamos se estiverem fora de ordem. Acompanhe na imagem ao lado.

Chegamos ao fim da primeira iteração e, como dito, não foi suficiente para ordenar o vetor.

Teremos que reiniciar, só que agora sabemos que, pelo menos, o último valor (7) já está em seu devido lugar.

Assim ele será marcado para não percorrer todo o vetor na segunda iteração. Veja na imagem abaixo o exemplo.

3 2 4 5 1 0 6 [7]

```
(5 3) 2 4 7 1 0 6  pegamos o primeiro par
3--5 2 4 7 1 0 6  trocamos

3 (5 2) 4 7 1 0 6  pegamos o próximo par
3 2--5 4 7 1 0 6  trocamos

3 2 (5 4) 7 1 0 6  pegamos o próximo par
3 2 4--5 7 1 0 6  trocamos

3 2 4 (5 7) 1 0 6  pegamos o próximo par
3 2 4 5--7 1 0 6  mantemos <----

3 2 4 5 (7 1) 0 6  pegamos o próximo par
3 2 4 5 1--7 0 6  trocamos

3 2 4 5 1 (7 0) 6  pegamos o próximo par
3 2 4 5 1 0--7 6  trocamos

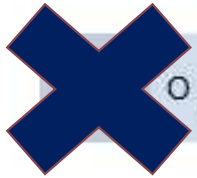
3 2 4 5 1 0 (7 6)  pegamos último par
3 2 4 5 1 0 6 7  trocamos
```

7) O elemento central fica no meio.

8) Conhecida como POO (Programação Orientada a Objetos), é uma paradigma da programação Que utiliza abstração para criar modelos baseados no mundo real. POO usa várias técnicas, incluindo modularidade, polimorfismo e encapsulamento.

9) A fila é mais justa (primeiro a entrar é o primeiro a sair)

10)



O paradigma Estruturado deu início ao paradigma orientado a objetos



**RESPONDA NO PRÓPRIO PPT SALVE EM PDF E ENTREGUE
NO TEAMS**

Obrigado e Boa Sorte!

