

Entendendo as definições de classe

- Atributos
- Construtores
- Métodos
- Parâmetros
- Atribuição
- Instruções condicionais

Universidade: um exame externo

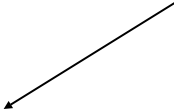
- Usaremos um exemplo de uma universidade. Nesta universidade, os professores ensinam alunos que são matriculados em cursos. Um curso é uma coleção de tópicos relacionados. Estudantes se matriculam para cursar diferentes cursos. Desde que existem limites da quantidade de alunos que um professor possa ensinar de uma só vez (a universidade tem como norma não possuir grandes turmas), quando muitos alunos precisando tomar um curso poderá haver várias turmas do mesmo curso ministradas pelo mesmo professor ou por mais de um professor.

Universidade: um exame interno

- Interagir com um objeto fornece dicas sobre seu comportamento.
- Examinar internamente permite determinar como esse comportamento é fornecido ou implementado.
- Todas as classes Python têm uma visualização interna semelhante.

Estrutura de uma classe básica

Envoltório
externo da
Estudante



```
class Estudante:
```

```
# Parte interna da classe omitida.
```

Python não possui uma palavra para o modificador de visibilidade: `private` ou `public`

Estrutura de uma classe básica

class *NomeDaClasse*:

Atributos

Construtores

Métodos

O conteúdo de uma classe

A ordem não importa, é uma questão de preferência.
Importante seguir um estilo

Atributos

- Atributos armazenam valores para um objeto utilizar.
- Eles também são conhecidos como **variáveis de instâncias**.
- Atributos definem o estado de um objeto.

Atributos

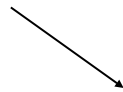
```
class Estudante:
    """attributes:
    nome
    matricula
    credits
    """

    # Construtor e métodos omitidos.
```

Atributos

Modificador de
visibilidade

Nome da
variável de
instância



creditos

Dois caracteres `__` informa que o modificador de visibilidade é um atributo privado

Comentários

- Os comentários são inseridos no código-fonte para fornecer explicações aos humanos
- Não têm nenhum efeito nas funcionalidades da classe

```
# quantidade de credits cursados  
self.__credits = 134
```

Comentários

- Comentários mais detalhados são escritos em múltiplas linhas
- Começam com três caracteres “””
- E terminam com o par “””
- “”” esse comentário é um mais longo, devido “””

Construtores

- Construtores inicializam um objeto.
- O construtor é um método especial chamado de `__init__`
- É executado na criação do objeto

Construtores

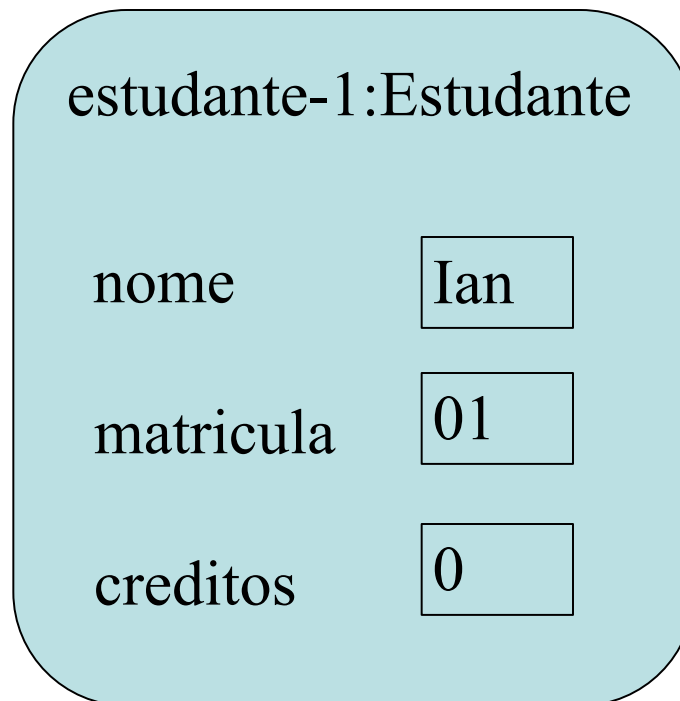
- Construtores armazenam valores iniciais nos atributos.
- Eles freqüentemente recebem valores de parâmetros externos nesses atributos.
- Os atributos também podem ser inicializados através de atribuições

Construtores

```
def __init__(self,name):  
    self.__nome = name  
    self.__matricula = "01"  
    self.__creditos = 0
```

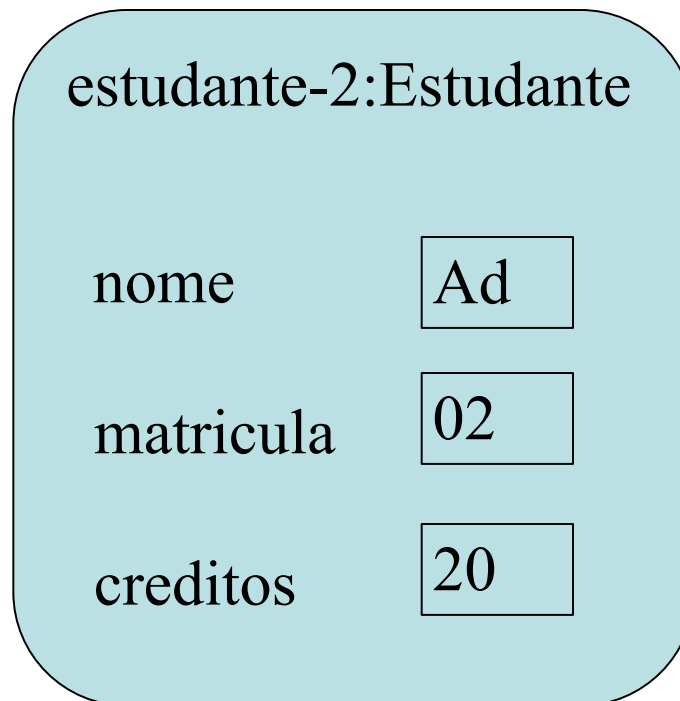
Construtores

estudante-1 = **Estudante**("Ian")



Transmitindo dados via parâmetros

estudante-2 = **Estudante**("Ad")



Atribuição

- Valores são armazenados em atributos (e outras variáveis) via instruções de atribuição:
variável = expressão
`self.__nome = name`
- Uma variável armazena um único valor, portanto, qualquer valor anterior é perdido.

Métodos de acesso (1)

- Métodos implementam o comportamento dos objetos.
- Métodos de acesso fornecem informações sobre um objeto.

Métodos de acesso (2)

- Métodos têm uma estrutura que consiste em um cabeçalho e um corpo.
- O cabeçalho define a *assinatura do método*.
`getCreditos(self)`
- O corpo engloba as instruções do método.

Métodos de acesso (3)

Nome do método

def getCreditos(self) :

Lista de
parâmetros
(vazia)

return self.__creditos

Instrução de retorno

Métodos modificadores (1)

- Eles têm uma estrutura de método semelhante: cabeçalho e corpo.
- Utilizados para *modificar* o estado de um objeto.
- Alcançados por meio da modificação do valor de um ou mais atributos.
 - Geralmente contêm instruções de atribuição.
 - Geralmente recebem parâmetros.

Métodos modificadores (2)

The diagram illustrates the components of a Python method definition. It shows the code `def setCredits(self, quantidade):` followed by an indented line `self.__credits = quantidade`. Four arrows point from text labels to specific parts of the code: 'Nome do método' points to `setCredits`, 'Parâmetro' points to `quantidade`, 'Atributo sendo alterado' points to `self.__credits`, and 'Instrução de atribuição' points to the assignment operator `=`.

```
def setCredits(self, quantidade):  
    self.__credits = quantidade
```

Nome do método

Parâmetro

Atributo sendo alterado

Instrução de atribuição

Operador +=

- `variavel = variavel + expressão`
- `creditos = creditos + quantidade`
- `creditos += quantidade`

Refletindo sobre o projeto universidade

- Seus comportamentos não são adequados por várias razões:
 - nenhuma verificação dos valores inseridos;
 - nenhuma verificação quanto a uma inicialização sensata.
- Como podemos melhorar isso?
 - Precisamos de um comportamento mais sofisticado.

Fazendo escolhas (1)

```
def addCreditos(self, quantidade):  
    if(quantidade > 0):  
        self.__creditos += quantidade  
  
    else :  
        print("Use uma quantidade positiva:  
", quantidade)
```


Fazendo escolhas (2)

palavra-chave 'if' condição booleana a ser testada — fornece um resultado *True* (verdadeiro) ou *False* (falso) ações se a condição for verdadeira

```
if(realiza algum teste) :  
    Segue as instruções aqui se o teste forneceu um resultado verdadeiro
```

else :
 Segue as instruções aqui se o teste forneceu um resultado falso

palavra-chave 'else' ações se a condição for falsa

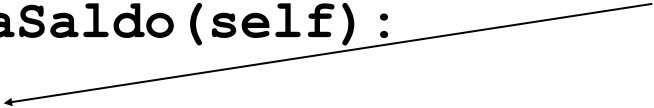
Variáveis locais

- Atributos são um tipo de variável. Elas:
 - armazenam valores por toda a vida de um objeto; e
 - são acessíveis por meio da classe.
- Métodos podem incluir variáveis de vida mais curta. Elas:
 - existem apenas enquanto o método está em execução; e
 - são acessíveis de dentro do método.

Variáveis locais

Uma variável local

```
def retornaSaldo(self):  
    quantidadeRetorno = self.__saldo  
    self.__saldo = 0  
    return quantidadeRetorno
```



Revisão (1)

- O corpo das classes contém atributos, construtores e métodos.
- Atributos armazenam valores que determinam o estado de um objeto.
- Construtores inicializam objetos.
- Métodos implementam o comportamento dos objetos.

Revisão (2)

- Atributos, parâmetros e variáveis locais são variáveis.
- Atributos persistem pelo tempo de vida de um objeto.
- Parâmetros são utilizados para receber valores em um construtor ou método.
- Variáveis locais são utilizadas para armazenamento temporário de curta duração.

Revisão (3)

- Objetos podem tomar decisões via atribuições condicionais (if).
- Um teste de verdadeiro ou falso permite que uma entre duas ações alternativas seja tomada.