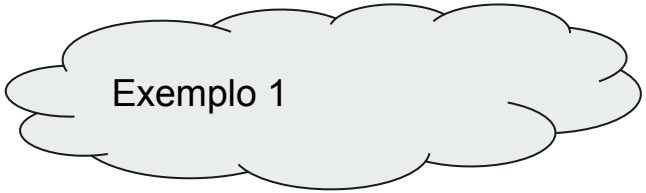
Tópico 13 - POO - Parte III

Atributos Privados, Classes Abstratas

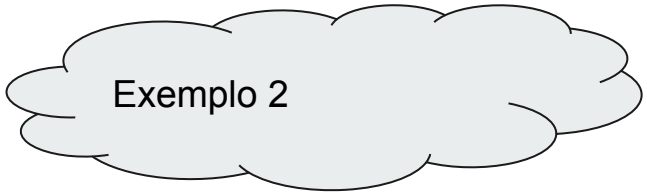
Atributos Privados

- Linguagens orientadas a objetos (Java, C++, etc.) controlam ou restrigem o acesso a atributos das classes.
- Nesse caso, os atributos da classe passam a ser classificados como público (*public*), privado (*private*) ou protegido (*protected*).
- Não há mecanismos em Python que impeçam de fato o acesso a qualquer variável ou método de uma classe.
- Todos os atributos de uma classe em Python são públicos por definição.
- **Atributos públicos** (*public*) podem ser acessados fora da classe.



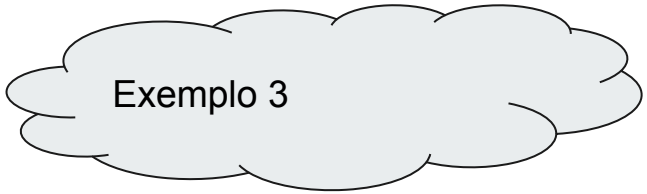
Atributos Privados

- Python permite simular acessos do tipo protegido ou privado através da adição de prefixos aos nomes de variáveis e métodos de uma classe.
- Atributos protegidos (protected) de uma classe podem ser acessados dentro da própria classe e pelas classes filhas.
- Atributos protegidos não podem ser acessados a partir de outros ambientes.
- Utiliza-se 1 underscore: _<nomeAtributo>



Atributos Privados

- Atributos privados têm seu acesso efetivamente impedido a partir de outros ambientes.
- Tentativas de acesso externo resultam em erro.
- Utiliza-se dois underscore: __<nomeAtributo>



Classes Abstratas

- As classes abstratas possuem um ou mais métodos do tipo abstrato.
- Um método abstrato é declarado, mas não é implementado.

```
@abstractmethod
def precisa_implementar(self):
    pass
```

- O decorador `@abstractmethod` torna o método abstrato.
- Python por default não fornece classes abstratas.
- Deve-se importar a Abstract Base Classes (ABCs)

```
from abc import ABC
```