

# Programação Orientada a Objetos

Classes abertas

# Classes abertas

No Python podemos alterar nossas classes e objetos em tempo de execução adicionando novos métodos e atributos.

Isso é possível porque o Python é uma linguagem dinâmica.

Somente não podemos modificar classes ***built-ins*** (ou seja, as classes embutidas do Python).

# Classes abertas

Podemos adicionar novos atributos e métodos de forma dinâmica quando não quisermos modificar uma classe existente ou quando não possuímos o código-fonte da classe, tendo apenas seu ***bytecode***, por exemplo.

Vamos criar uma classe e gerar o ***bytecode*** para simular um caso real.

# Classes abertas

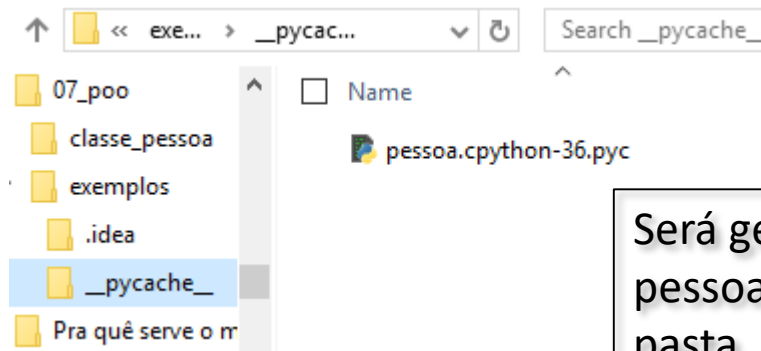
Primeiro vamos criar uma classe e compilar seu código.

```
class Pessoa(object):  
    def __init__(self, nome):  
        self.nome = nome
```

Temos que importar o módulo  
py\_compile para gerar o bytecode,  
arquivo pyc.

O comando para gerar o bytecode é:  
py\_compile.compile("arquivo.py")

```
Command Prompt - python  
6) [MSC v.1900 64 bit (AMD64)] on win32  
Type "help", "copyright", "credits" or "license" f  
or more information.  
>>> import py_compile  
>>> py_compile.compile("pessoa.py")  
'__pycache__\pessoa.cpython-36.pyc'  
>>> py_compile.compile("pessoa.py")  
'__pycache__\pessoa.cpython-36.pyc'  
>>>
```



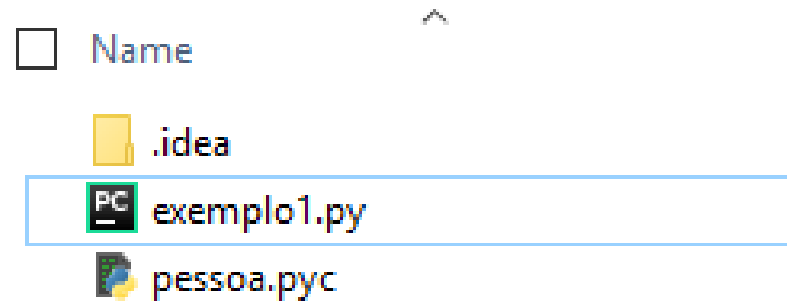
Será gerado o arquivo  
pessoa.cpython-36.pyc na  
pasta \_\_pycache\_\_

# Classes abertas

Vou excluir o `pessoa.py`, copiar o ***pessoa.cpython-36.pyc*** para a pasta do meu projeto e renomear para ***pessoa.pyc***.

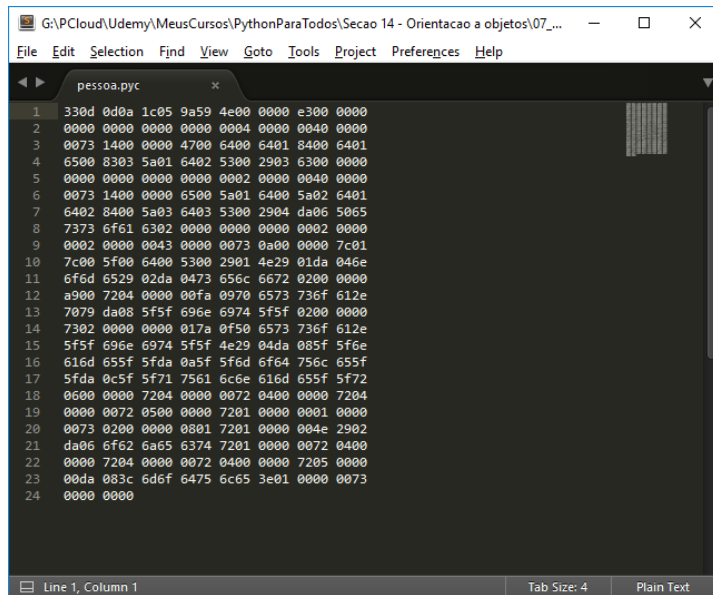
Em seguida vou apagar a pasta ***\_\_pycache\_\_***.

A pasta do projeto ficará assim. O arquivo `exemplo1.py` será nosso programa que usará a classe ***Pessoa*** de ***pessoa.pyc***.

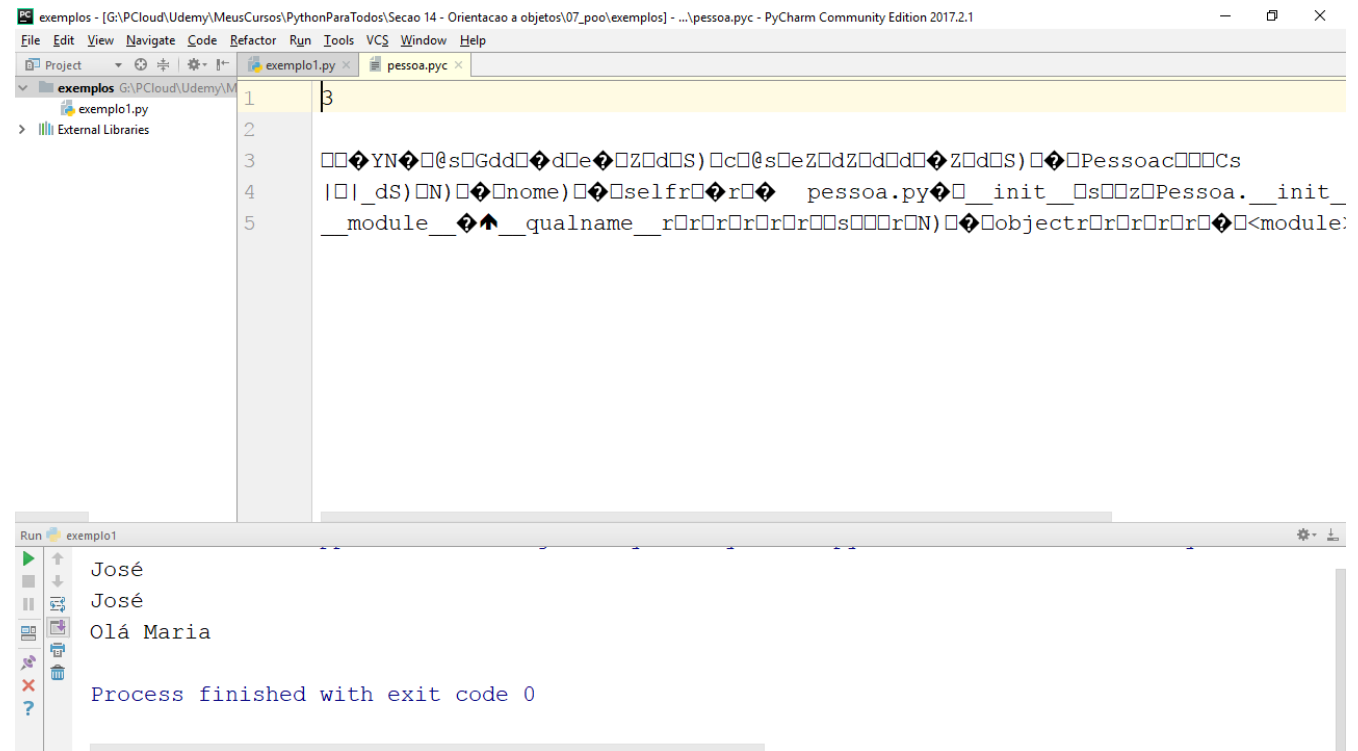


# Classes abertas

Se tentar editar o `pessoa.pyc` verá que ele é um binário (compilado). Veja o mesmo aberto nos editores Sublime e PyCharm.



The image shows the Sublime Text editor with the file `pessoa.pyc` open. The content is a list of 24 lines of hexadecimal byte sequences, representing the compiled Python bytecode. The status bar at the bottom indicates 'Line 1, Column 1', 'Tab Size: 4', and 'Plain Text'.



The image shows the PyCharm IDE with the file `pessoa.pyc` open. The content is the same hexadecimal byte sequences as seen in the Sublime Text editor. Below the editor, the Run window shows the output of a Python script, which includes the names 'José', 'José', and 'Olá Maria', followed by the message 'Process finished with exit code 0'.

# Classes abertas

Vamos ao programa que vai usar a classe Pessoa.

```
from pessoa import Pessoa

# Criado objeto p da classe pessoa passando 'José' para nome
p = Pessoa('José')
print(p.nome) # imprimindo o valor do atributo nome do objeto p

# Criando um método chamado retornar_nome que retorna a propriedade nome (de self)
def pegar_nome(self):
    return self.nome

# Criando o método pegar_nome na classe Pessoa com base no método pegar_nome que criei acima
Pessoa.pegar_nome = pegar_nome

# Executando o método pegar_nome do objeto p
print(p.pegar_nome())
```

# Classes abertas

Vamos ao programa que vai usar a classe Pessoa.

```
#Criando o método mudar_nome
def mudar_nome(self, novo_nome):
    self.nome = novo_nome

# Criando o método mudar_nome na classe Pessoa
Pessoa.mudar_nome = mudar_nome

# Executando o método mudar_nome no objeto p
p.mudar_nome('Maria')

# Criando um método denominado imprimir_ola para imprimir "Olá x", onde x é o valor da propriedade nome
def imprimir_ola(self):
    print("Olá", self.nome)

# Criando o método ola na classe Pessoa com base no método imprimir_ola
Pessoa.ola = imprimir_ola
# Executando o método ola do objeto p
p.ola()
```



# CONTINUA...