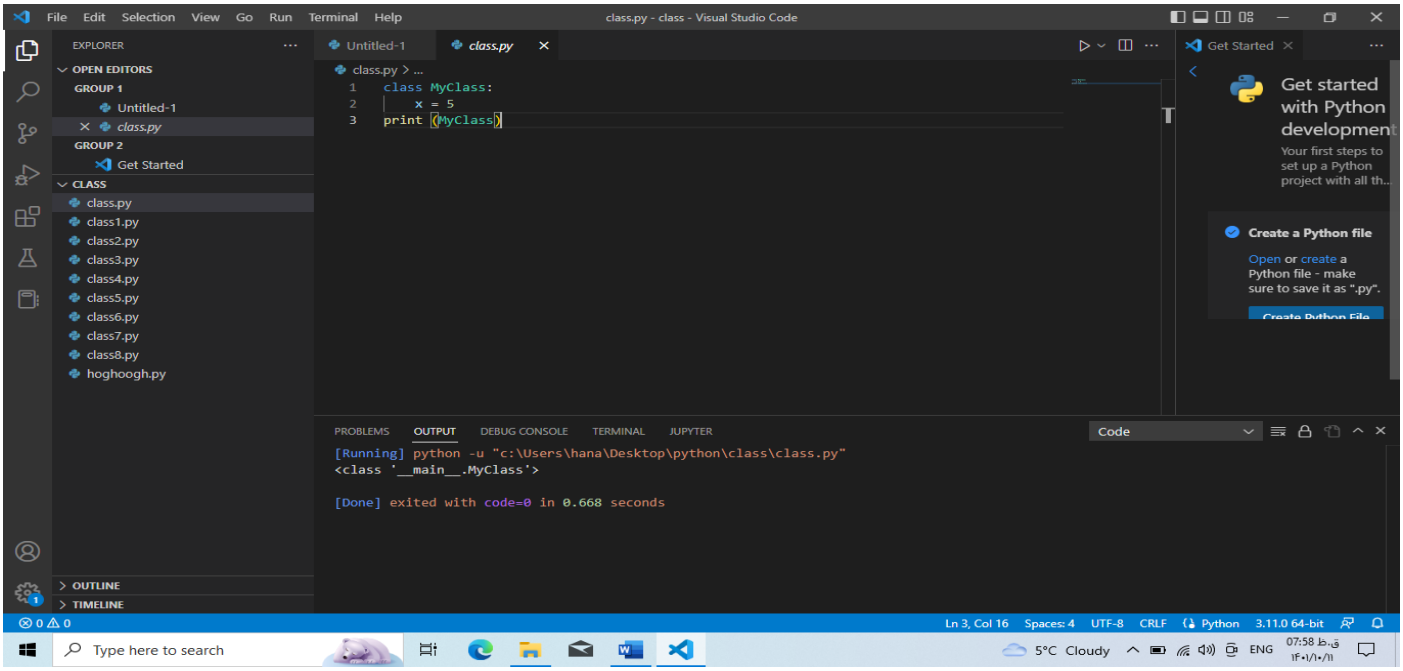
[C1113-OOP-EX5][TahrehGholami-401114037180030]

پایتون یک زبان برنامه نویسی شی گرا است .تقریباً همه چیز در پایتون یک شی است، با خواص و متدهای آن .یک کلاس مانند یک سازنده شی یا یک "طرح" برای ایجاد اشیا است.

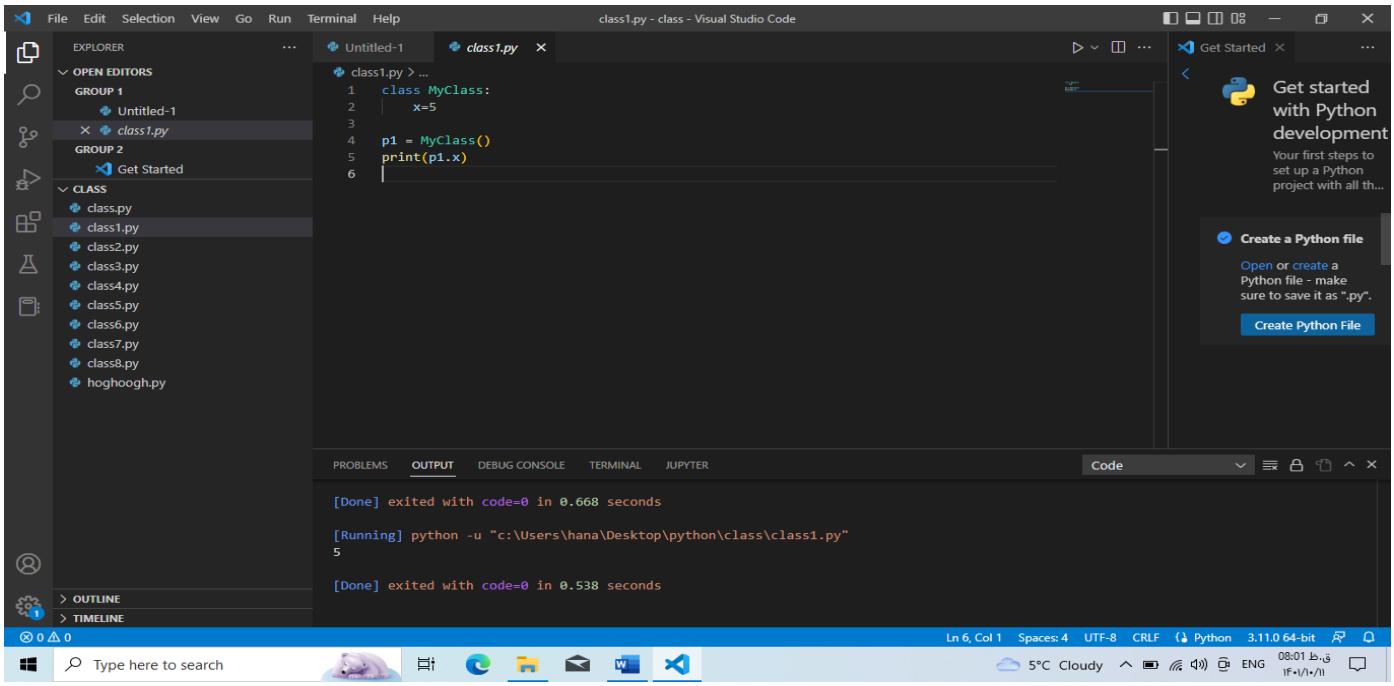
Create a Class: برای ایجاد یک کلاس، از کلمه کلیدی class استفاده کنید:

```
class MyClass:
    x = 5
print (MyClass)
```



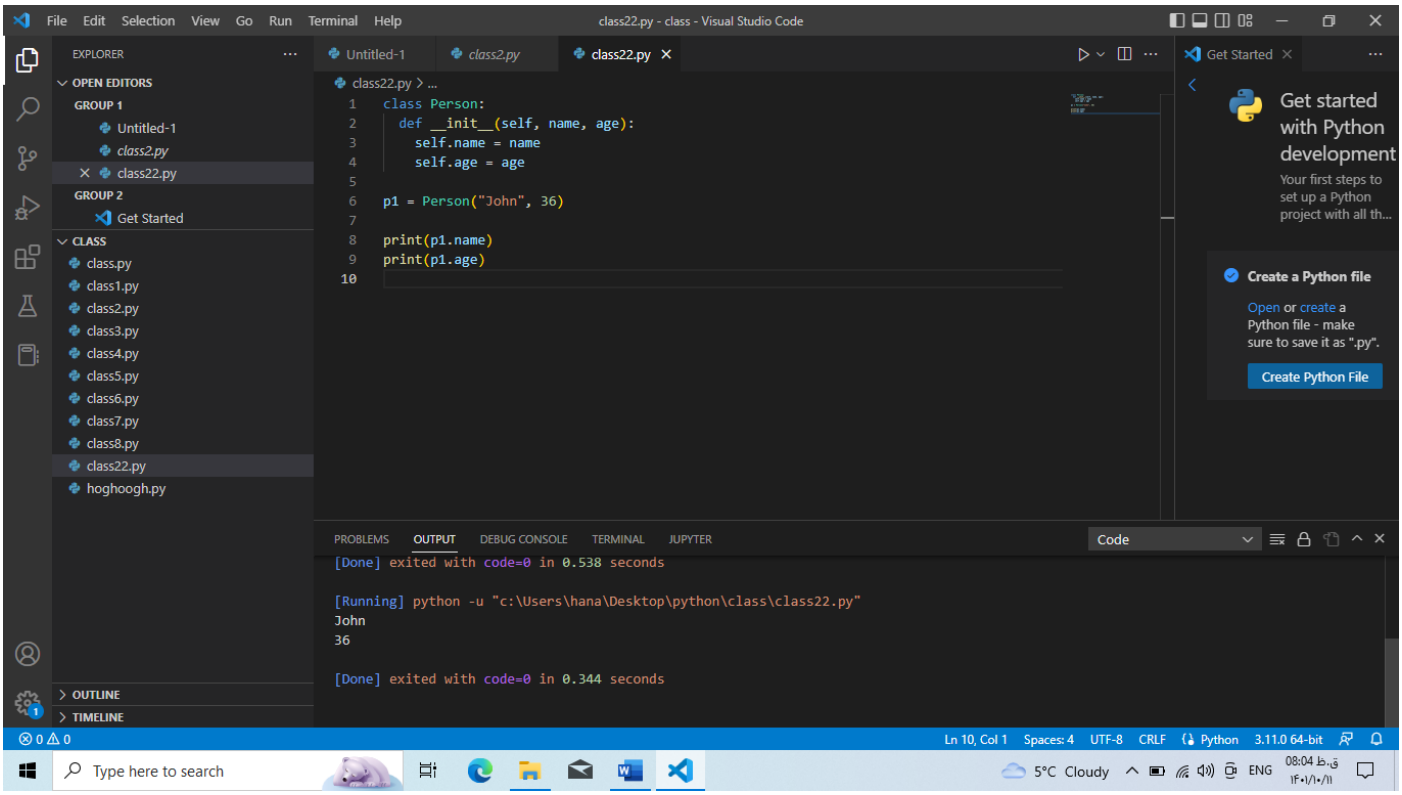
Create Object برای ساخت یک object از این دستور استفاده می شود

```
class MyClass:
    x=5
p1 = MyClass()
print(p1.x)
```



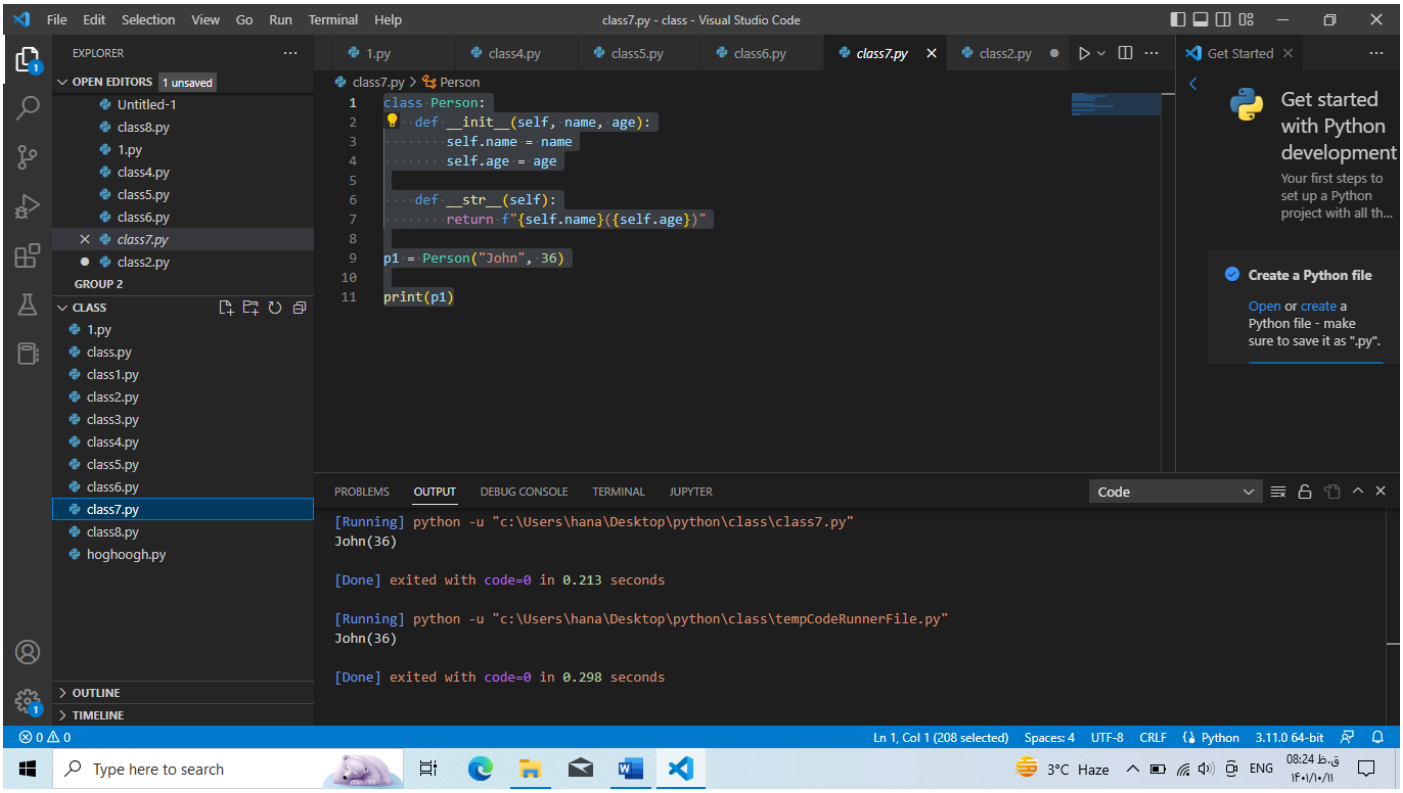
The __init__() Function برای درک معنای کلاس ها باید تابع __init__() داخلی را درک کنیم .همه کلاس ها دارای تابعی به نام __init__()هستند که همیشه زمانی که کلاس شروع می شود اجرا می شود .از تابع __init__() برای تخصیص مقادیر به خصوصیات شی یا سایر عملیاتی که هنگام ایجاد شیء لازم است انجام دهید استفاده کنید:

```
class Person:
 def __init__(self, name, age):
   self.name = name
   self.age = age

p1 = Person("John", 36)

print(p1.name)
print(p1.age)
```



The __str__() Function تابع __str__() کنترل می کند که وقتی شی کلاس به عنوان یک رشته نمایش داده می شود، چه چیزی باید برگردانده شود. اگر تابع __str__() تنظیم نشده باشد، نمایش رشته ای شی برگردانده می شود:

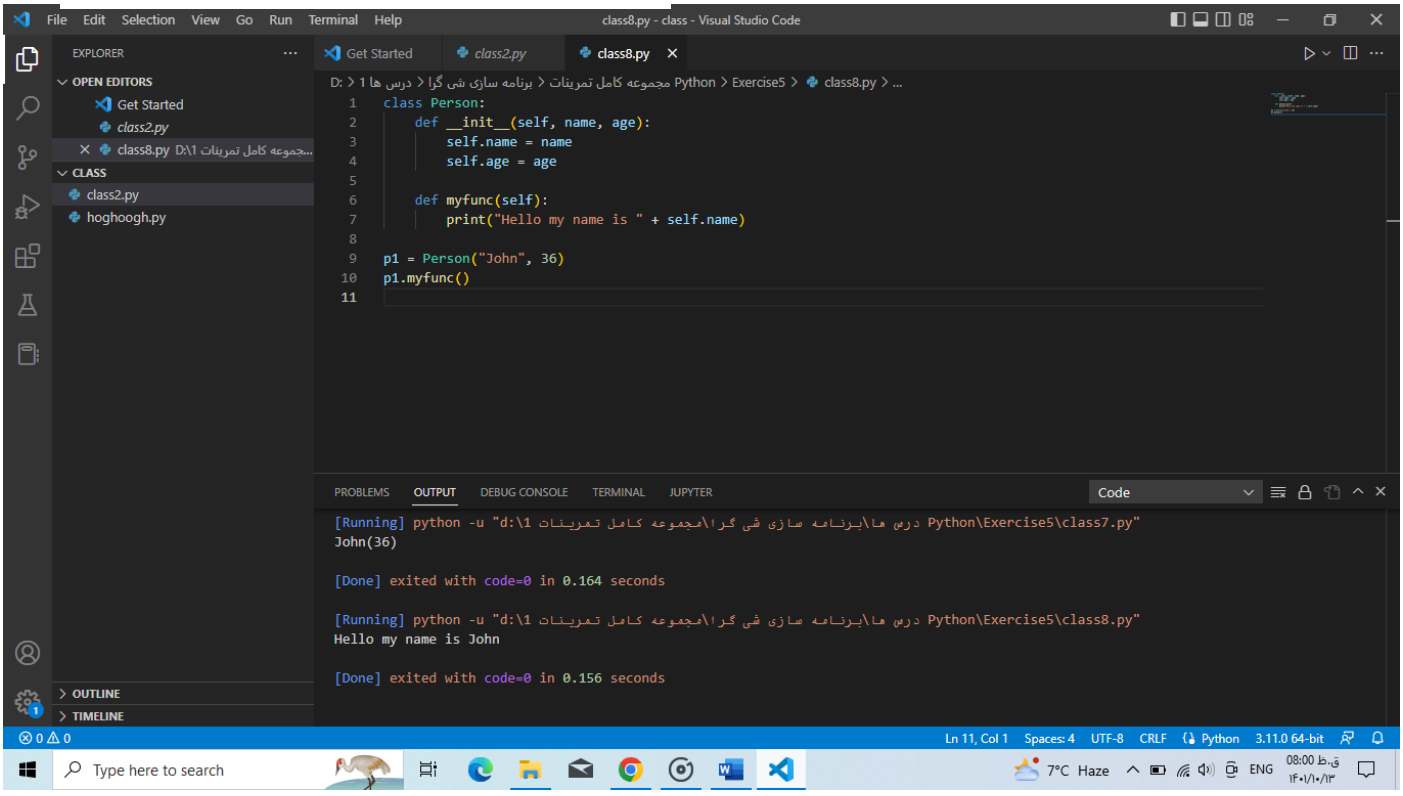
```
class Person:

  def __init__(self, name, age):

    self.name = name

    self.age = age


  def __str__(self):

    return f"{self.name}({self.age})"


p1 = Person("John", 36)


print(p1)
```



Object Methods اشیاء همچنین می توانند حاوی متدهایی باشند.

متدها در اشیا توابعی هستند که به شی تعلق دارند.

```
class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):
    print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```



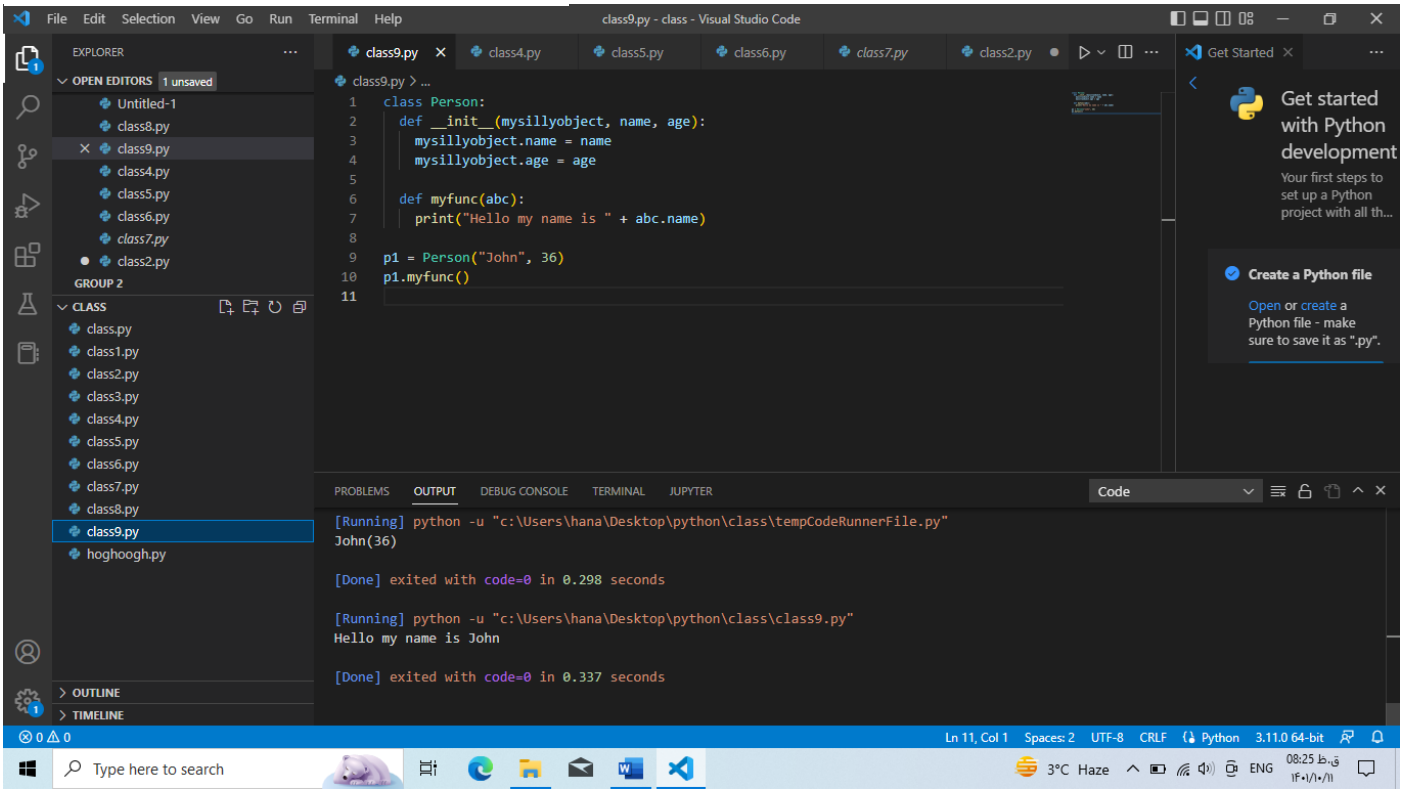
The self Parameter پارامتر `self` اشاره ای به نمونه فعلی کلاس است

برای دسترسی به متغیرهایی که متعلق به کلاس هستند استفاده می شود.

لازم نیست خود نامگذاری شود، می توانید آن را هر چه دوست دارید صدا بزنید،

اما باید اولین پارامتر هر تابع در کلاس باشد.

```
class Person:
  def __init__(mysillyobject, name, age):
    mysillyobject.name = name
    mysillyobject.age = age

  def myfunc(abc):
    print("Hello my name is " + abc.name)

p1 = Person("John", 36)
p1.myfunc()
```



در این کد در تعریف کلاس یه متغیر تعریف می کنیم به عنوان مثال x=5 در

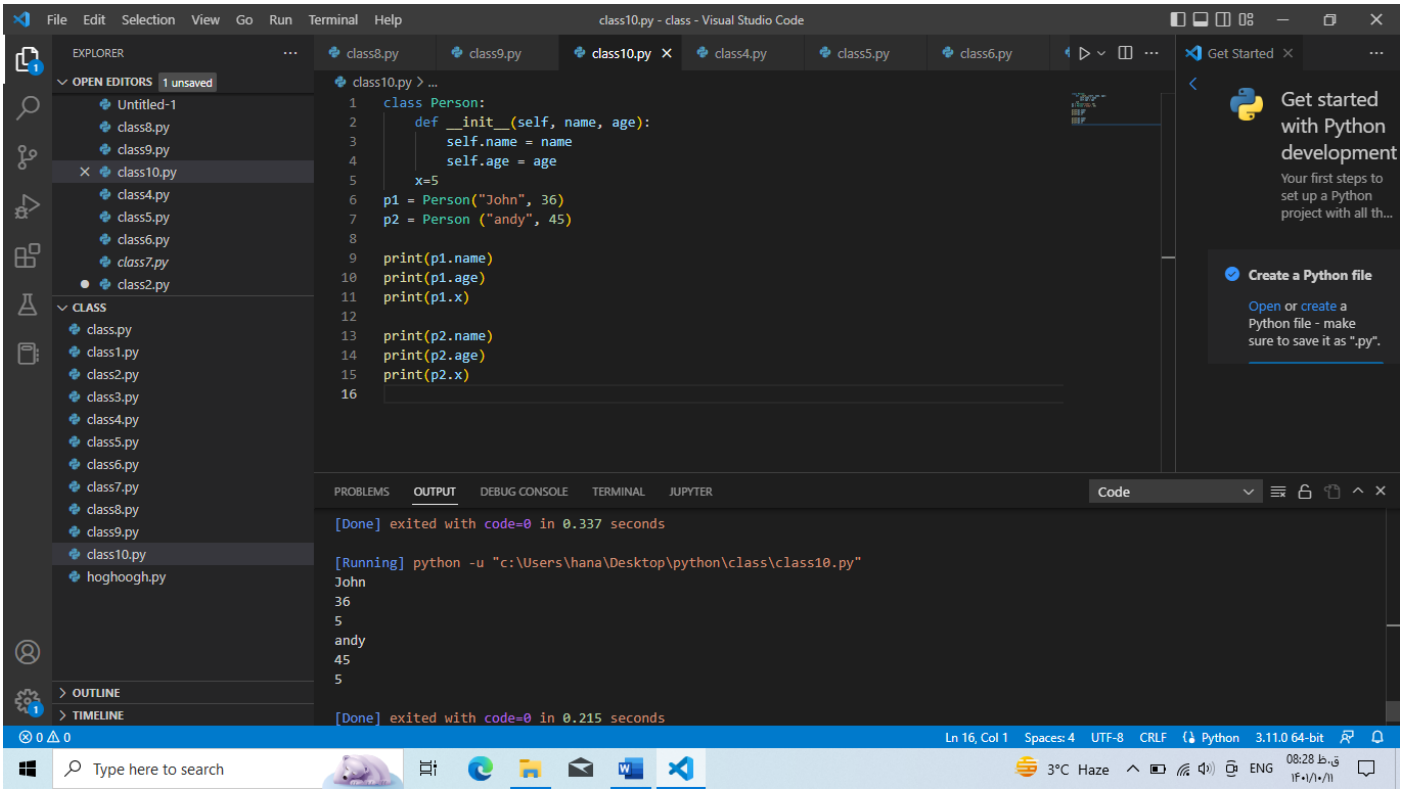
این کد که در پرینت بعد از هر `object` که اینجا شامل نام و سن است یکبار

عدد 5 پرینت می شود و به خط بعدی رفته و بعد `object` بعدی شامل نام

و سن چاپ و بعد دوباره عدد 5 پرینت می شود

در بدنه کلاس، نام متغیر (ویژگی) را نوشته و مقداردهی می‌کنیم.

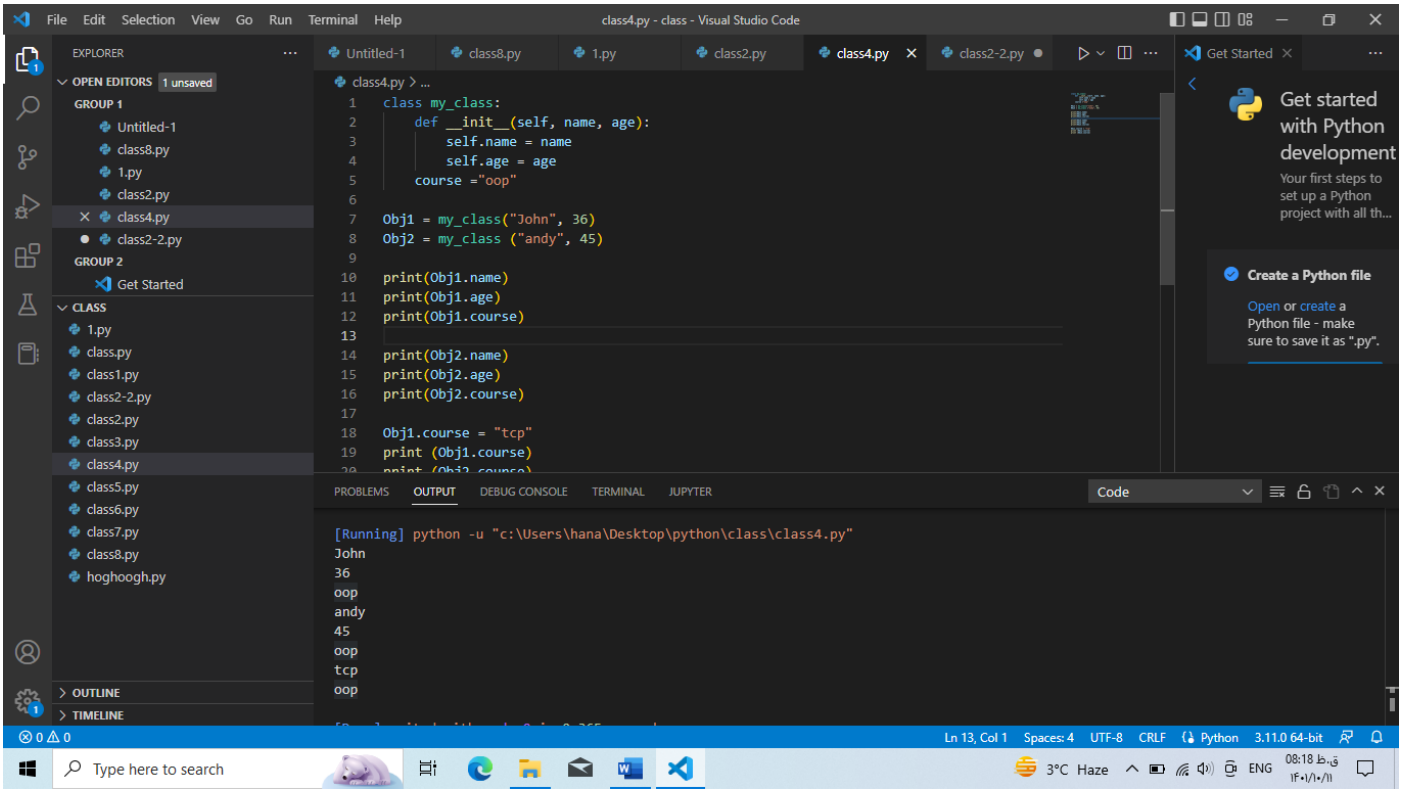
```
class Person:
   def __init__(self, name, age):
      self.name = name
      self.age = age
   x=5

p1 = Person("John", 36)

p2 = Person ("andy", 45)

print(p1.name)

print(p1.age)

print(p1.x)


print(p2.name)

print(p2.age)
```



در این کد نیز مثل مثال بالا یک متغیر به نام course ="oop" بعد از هر object پرینت می شود و در خط 18 تغییر میدهیم به course = "tcp" و به object1 فقط نسبت می دهیم و از آن به بعد فقط هر دو course چاپ می شود.

در بدنه کلاس، نام متغیر (ویژگی) را نوشته و مقداردهی می‌کنیم.

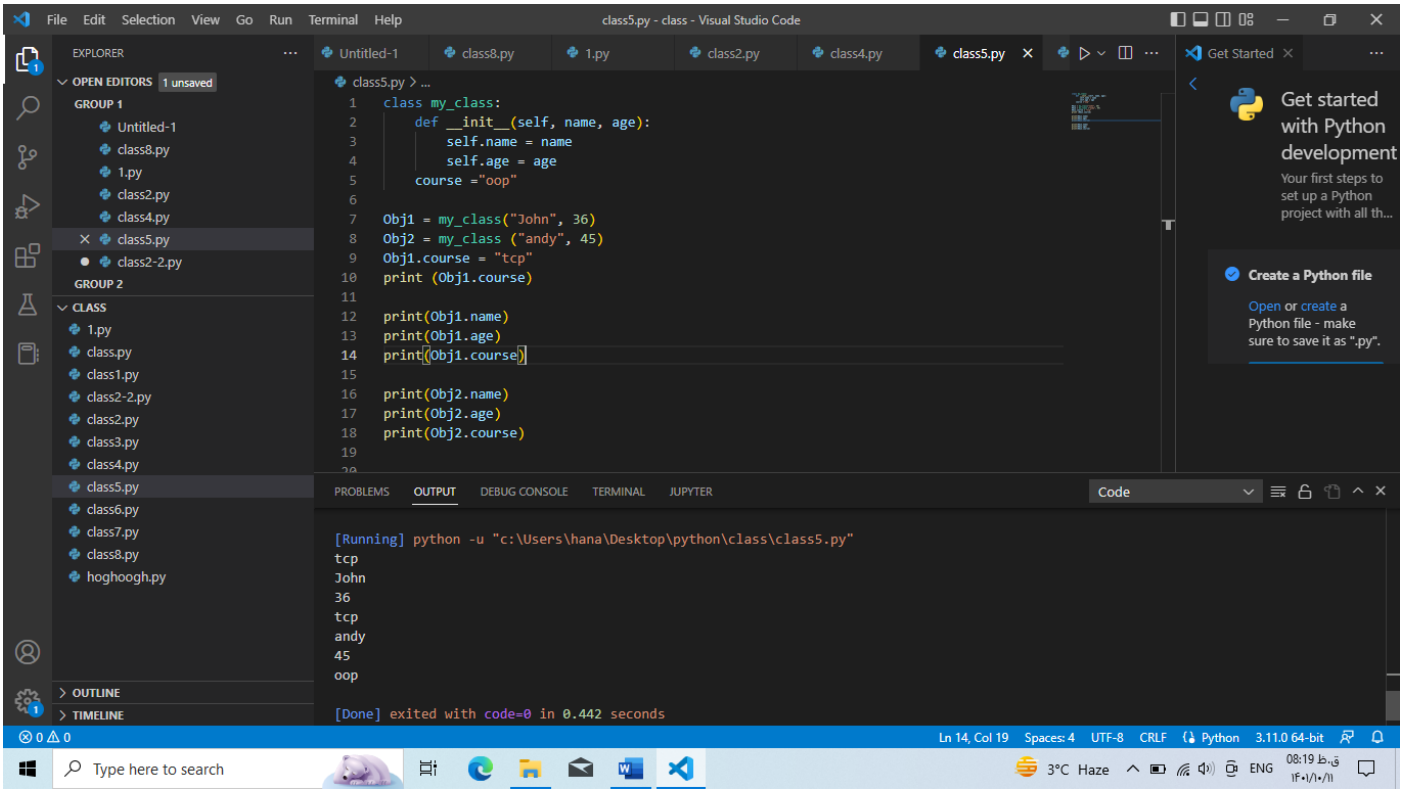
```
class my_class:
    def __init__(self, name, age):
        self.name = name
        self.age = age
    course ="oop"

Obj1 = my_class("John", 36)
Obj2 = my_class ("andy", 45)

print(Obj1.name)
print(Obj1.age)
print(Obj1.course)

print(Obj2.name)
print(Obj2.age)
print(Obj2.course)

Obj1.course = "tcp"
print (Obj1.course)
print (Obj2.course)
```



در این کد اول خواستیم که course = "tcp" جدید و بعد از روی course را در object1 , object2 چاپ کند.

در بدنه کلاس، نام متغیر (ویژگی) را نوشته و مقداردهی می‌کنیم و در خط 10 آن را تغییر می دهیم Obj1.course = "tcp"

```
class my_class:
    def __init__(self, name, age):
        self.name = name
        self.age = age
    course ="oop"

Obj1 = my_class("John", 36)
Obj2 = my_class ("andy", 45)
Obj1.course = "tcp"
print (Obj1.course)

print(Obj1.name)
print(Obj1.age)
print(Obj1.course)

print(Obj2.name)
print(Obj2.age)
print(Obj2.course)
```


File Edit Selection View Go Run Terminal Help
class5.py - class - Visual Studio Code
EXPLORER
OPEN EDITORS 1 unsaved
GROUP 1
Untitled-1
class8.py
1.py
class2.py
class4.py
class5.py
class2-2.py
GROUP 2
CLASS
1.py
class.py
class1.py
class2-2.py
class2.py
class3.py
class4.py
class5.py
class6.py
class7.py
class8.py
hoghoogh.py
OUTLINE
TIMELINE
Untitled-1
class8.py
1.py
class2.py
class4.py
class5.py
Get Started
class5.py > ...
class my_class:
def __init__(self, name, age):
self.name = name
self.age = age
course ="oop"
Obj1 = my_class("John", 36)
Obj2 = my_class ("andy", 45)
Obj1.course = "tcp"
print (Obj1.course)
print(Obj1.name)
print(Obj1.age)
print(Obj1.course)
print(Obj2.name)
print(Obj2.age)
print(Obj2.course)
Get started with Python development
Your first steps to set up a Python project with all th...
Create a Python file
Open or create a Python file - make sure to save it as ".py".
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Code
[Running] python -u "c:\Users\hana\Desktop\python\class\class5.py"
tcp
John
36
tcp
andy
45
oop
[Done] exited with code=0 in 0.442 seconds
Ln 14, Col 19 Spaces: 4 UTF-8 CRLF Python 3.11.0 64-bit
Type here to search
3°C Haze
ENG
08:19 ق.ظ