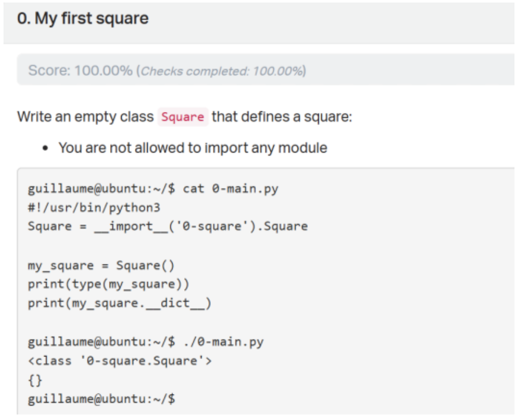
Python

Classes &Objects

Leen Mohammed Alsaleh

Task 0.My First Square



0. My first square

Score: 100.00% (*Checks completed: 100.00%*)

Write an empty class `Square` that defines a square:

- You are not allowed to import any module

```
guillaume@ubuntu:~/$ cat 0-main.py
#!/usr/bin/python3
Square = __import__('0-square').Square

my_square = Square()
print(type(my_square))
print(my_square.__dict__)

guillaume@ubuntu:~/$ ./0-main.py
<class '0-square.Square'>
{}
guillaume@ubuntu:~/$
```

السؤال طالب نسوي كلاس اسمه:
square
مافي ولا شي بداخله لا خصائص ولا دوال وممنوع
نستخدم مكتبات جاهزه

وش يعني كلاس ؟؟؟

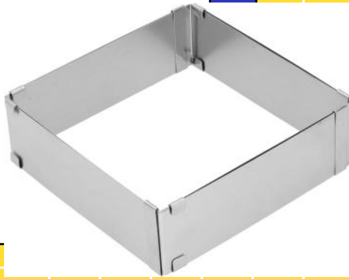
What is Class?

هو مثل قالب او مخطط عشان نصنع منه اشياء

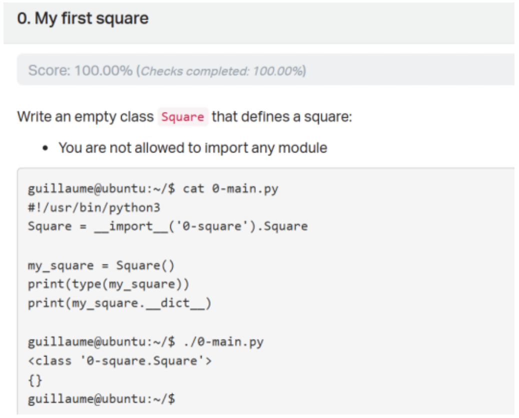
كيف يعني ؟؟

لو عندك قالب حديد يصنع لك مربعات زي هذا

الكلاس هو القالب والمربعات اللي تطلع منه هي الكائنات اللي هي
Object



Task 0.My Ftist Square



0. My first square

Score: 100.00% (*Checks completed: 100.00%*)

Write an empty class `Square` that defines a square:

- You are not allowed to import any module

```
guillaume@ubuntu:~/$ cat 0-main.py
#!/usr/bin/python3
Square = __import__('0-square').Square

my_square = Square()
print(type(my_square))
print(my_square.__dict__)

guillaume@ubuntu:~/$ ./0-main.py
<class '0-square.Square'>
{}
guillaume@ubuntu:~/$
```

السؤال طالب نسوي كلاس اسمه:

square

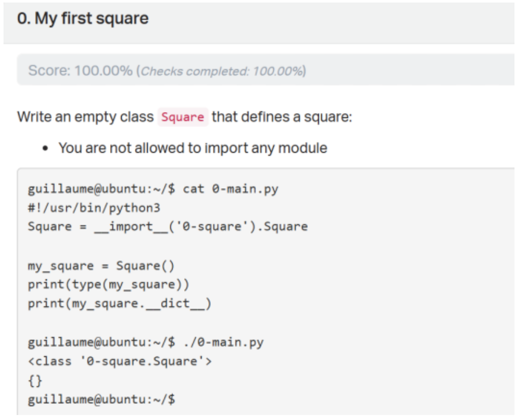
مافي ولا شي بداخله لا خصائص ولا دوال وممنوع نستخدم مكتبات جاهزه

```
#!/user/bin/python3
Class square:
pass
```

هنا سوينا كلاس

هنا أقول للبايثون ماعندي شي اكتبه االلحين خليني اكمل الكود وماتعطيني غلط

Task 0.My Ftist Square



0. My first square

Score: 100.00% (*Checks completed: 100.00%*)

Write an empty class `Square` that defines a square:

- You are not allowed to import any module

```
guillaume@ubuntu:~/$ cat 0-main.py
#!/usr/bin/python3
Square = __import__('0-square').Square

my_square = Square()
print(type(my_square))
print(my_square.__dict__)

guillaume@ubuntu:~/$ ./0-main.py
<class '0-square.Square'>
{}
guillaume@ubuntu:~/$
```

```
#!/user/bin/python3
Class square:
pass
```

هنا سوينا كلاس

هنا أقول للبايثون ماعندي شي اكتبه االلحين خليني اكمل الكود وماتعطيني غلط

```
My_square = square()
```

My_square

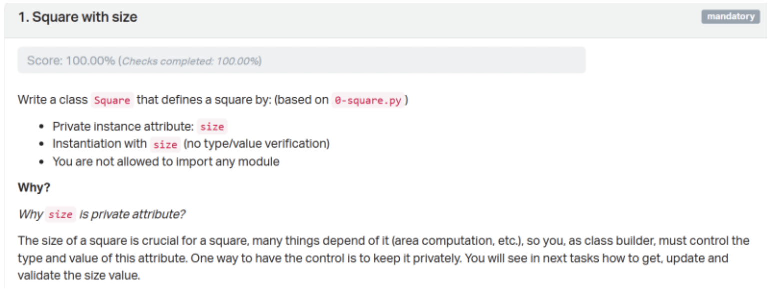
```
Print (type(My_square))
<class ‘0-square.square’>
```

هنا يوريني ايش نوع المتغير ومن أي كلاس انصنع ؟

```
Print (My_square.__dict)
{ }
```

هنا يطبع الخصائص بس ماعندنا شي فا راح يكون فاضي

Task 1. Square with size


1. Square with size

mandatory

Score: 100.00% (*Checks completed: 100.00%*)

Write a class `Square` that defines a square by: (based on `0-square.py`)

- Private instance attribute: `size`
- Instantiation with `size` (no type/value verification)
- You are not allowed to import any module

Why?

Why `size` is private attribute?

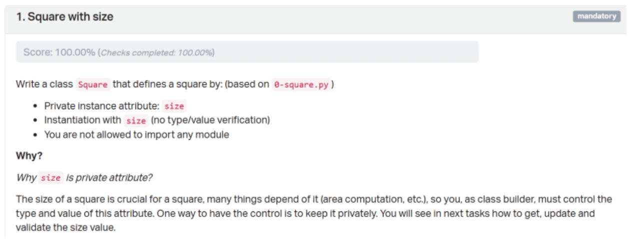
The size of a square is crucial for a square, many things depend of it (area computation, etc.), so you, as class builder, must control the type and value of this attribute. One way to have the control is to keep it privately. You will see in next tasks how to get, update and validate the size value.

هنا مطلوب نضيف خاصية جديده اسمها
Size
وتكون
Private

يعني ايش برايفت ؟

يعني مو أي احد يقدر يوصل لحجم المربع على طول

Task 1. Square with size



1. Square with size mandatory

Score: 100.00% (*Checks completed: 100.00%*)

Write a class `Square` that defines a square by: (based on `0-square.py`)

- Private instance attribute: `size`
- Instantiation with `size` (no type/value verification)
- You are not allowed to import any module

Why?

Why `size` is private attribute?

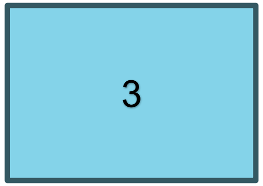
The size of a square is crucial for a square, many things depend of it (area computation, etc.), so you, as class builder, must control the type and value of this attribute. One way to have the control is to keep it privately. You will see in next tasks how to get, update and validate the size value.

هنا مطلوب نضيف خاصية جديده اسمها
Size
وتكون
Private

```
#!/user/bin/python3
Class square:

def__init__(self, size):
```

→ هنا صار عندنا شي اسمه
Constructor
يعني داله بنائيه
بحيث ان لما اجي اكتب
Square(3)
هذا السطر هو اللي بيشتغل
بشكل اخر اذا احد طلب مربع جديد خذ الرقم وحطه داخل المربع

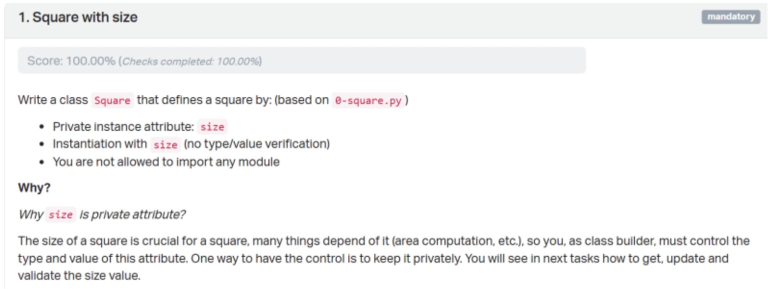


```
self.__size = size
```

→ نخزن الرقم داخلها __size

question

Task 1. Square with size



1. Square with size

mandatory

Score: 100.00% (*Checks completed: 100.00%*)

Write a class `Square` that defines a square by: (based on `0-square.py`)

- Private instance attribute: `size`
- Instantiation with `size` (no type/value verification)
- You are not allowed to import any module

Why?

Why `size` is private attribute?

The size of a square is crucial for a square, many things depend of it (area computation, etc.), so you, as class builder, must control the type and value of this attribute. One way to have the control is to keep it privately. You will see in next tasks how to get, update and validate the size value.

```
My_square= square(3)

Print(type(my_square))

→ <class '1-square.square'>

Print(my_square.__dicit__)

→ {'_square__size' :3}
```

Task 2.Size validation

- #!/user/bin/python3
- Class square:
- def__init__(self, size=0): →
- if not isinstance(size,int):
- raise TypeError(“size must be an integer”)
- elif size <0:
- raise ValueError(“ size must be >=0”)
- self.__size = size

لو احد سوا مربع جديد اعطيني الحجم واذا ماعطاني شي يكون صفر

ex:

M1= square(5) -> 5

M2= square() -> 0

question

Task 2.Size validation

What is the output ??

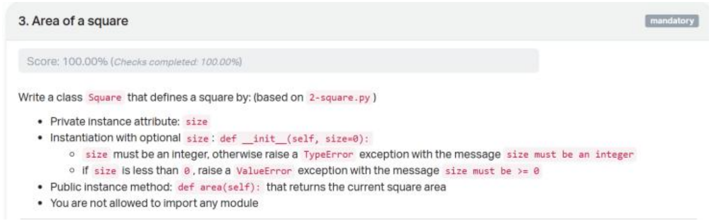
- Print(“ Test1: square(3)”)
- my_square_1 = square(3)
- Print(my_square_1.__dict__)

- Print(“ Test2 = : square()”)
- my_square_2 = square()
- Print(my_square_2.__dict__)

```
Test1 : square(3)
{‘__square__size’:3}
```

```
Test2 : square()
{‘__square__size’:0}
```

3.Area of square


3. Area of a square

Score: 100.00% (*Checks completed: 100.00%*)

Write a class `Square` that defines a square by: (based on `2-square.py`)

- Private instance attribute: `size`
- Instantiation with optional `size` : `def __init__(self, size=0):`
    - `size` must be an integer, otherwise raise a `TypeError` exception with the message `size must be an integer`
    - if `size` is less than `0` , raise a `ValueError` exception with the message `size must be >= 0`
- Public instance method: `def area(self):` that returns the current square area
- You are not allowed to import any module

نبني داله اسمها
Area()
ترجع المساحه حقت المربع

3.Area of square

```
#!/user/bin/python3
Class square:

def__init__(self, size=0):

if not isinstance(size,int):

raise TypeError("size must be an integer")

elif size <0:

raise ValueError(" size must be >=0")

self.__size = size

def area(self):
Return self.__size*self.__size
```

$\longrightarrow$ نناديها من برا الكلاس عشان ترجع المساحه

$\longrightarrow$ المساحه = الضلع * الضلع

question

3.Area of square

```
My_square = square(-3)
Print(my_square.area())
```

$\longrightarrow$ Size must be >= 0

```
My_square = square("4")
Print(my_square.area())
```

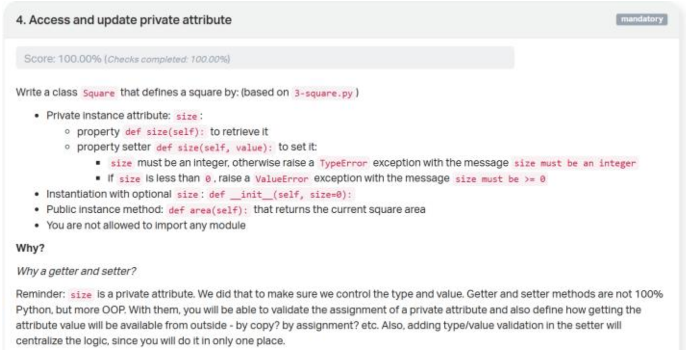
$\longrightarrow$ Size must be integer

```
My_square = square(4)
Print(my_square.area())
```

$\longrightarrow$ 16

4*4

4. Access and update private attribute



4. Access and update private attribute

mandatory

Score: 100.00% (*Checks completed: 100.00%*)

Write a class `Square` that defines a square by: (based on `3-square.py`)

- Private instance attribute: `size` :
  - property `def size(self):` to retrieve it
  - property setter `def size(self, value):` to set it:
    - `size` must be an integer, otherwise raise a `TypeError` exception with the message `size must be an integer`
    - if `size` is less than `0` , raise a `ValueError` exception with the message `size must be >= 0`
- Instantiation with optional `size` : `def __init__(self, size=0):`
- Public instance method: `def area(self):` that returns the current square area
- You are not allowed to import any module

Why?

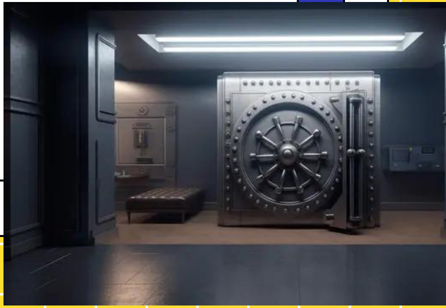
Why a getter and setter?

Reminder: `size` is a private attribute. We did that to make sure we control the type and value. Getter and setter methods are not 100% Python, but more OOP. With them, you will be able to validate the assignment of a private attribute and also define how getting the attribute value will be available from outside - by copy? by assignment? etc. Also, adding type/value validation in the setter will centralize the logic, since you will do it in only one place.

هنا الفكره نضيف طريقه نقدر نستخدم فيها المتغير
__size
من برا الكلاس بشكل امن

تخيلو لو عندنا غرفه مقفله
هذا هو ال السايز لكن عطيناكم مفتاح خاص ندخل
ونغير اللي نبغا بس بشروط المفتاح هذا هو :

@property
@size.setter



4. Access and update private attribute

```
#!/user/bin/python3
Class square:

def__init__(self, size=0):

@property
def size (self):
Return self.__size

@size.setter
def size(self, value):

if not isinstance(size,int):

raise TypeError("size must be an integer")

elif size <0:

raise ValueError(" size must be >=0")

self.__size = size

def area(self):
Return self.__size*self.__size
```

هذا نسميه
Getter
نستخدمه عشان نقدر نقرا فيه قيمه
__size
من برا الكلاس

هذا نسميه
Setter
يستخدم عشان نقدر نعدل قيمة
__size
من برا الكلاس لكن بشرط :
لازم يكون عدد صحيح
ولازم يكون اكبر او يساوي الصفر

question

4. Access and update private attribute

```
My_square = square(89)
Print(my_square.area())
```

$\longrightarrow$

7921

89*89

```
My_square = square(3)
Print(my_square.area())
```

$\longrightarrow$

9

3*3

Leen Mohammed Alsaleh

Thank you