

Python - Classes and Objects

٩٩ Class إيش هو الا

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:~/cat 0-main.py
<class '0-square.Square'>
()
guillaume@ubuntu:~/cat
```

هو مثل قالب أو مخطط نكتبه عشان نصنع منه أشياء
كين يعفي ٤٤ :

تحذل لو عندك قالب حديه يضع لك مرباع من كونون الملاس هو القالب والمربيات الى تطلع منه هي "الأشياء" أو "الكائنات" (Objects) اللي تستخدموها بعد ما تكتب.

لقولك أكتب كلاس فامي أسمه `Square` يعني مربع

لعيي يفينا شوي لشيء كلاس أسمه `Square` بس ما فيه أي شيء داخله (بلا دفاتر ولا دوال)
ممنوع تستخدم أي `import` يعني لا تستخدمو مكتبات باجهزه .

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

له مولين مربع جديه باستخدام الكلاس اللي كتبته

لعيي استخدم المقالب اللي أسمه `Square` ومويل مربع جديه وأخذه في معي أسمه
النتيجة هار عندك فرم جديه بس ما فيه أي ميزه لايج ولازم لأن الملاس نفسه فامي

`Print(type(my_square))`

له هذا بطبع نوع متغير `my_square` يعني كأنه يتعل دريني ويش ذفع المستير هدا `my_square` من أول كلاس أنا افتح

والت نتيجه بمحن :

`<Class '0-square.Square'>`

هذا اولاً object مفتوح عن كلاس `Square` اللي موجود في ملف `square.py`

`Print(my_square.__dict__)`

له هذا بطبع إنها اشيء الكائن (`object`)

طبعاً هو حالياً ما فيه شي لأن الكلاس فامي

فارجعيلج : ٤٣

الاختبار الكود

المطلوب من السؤال

Class Car :
Pass

`My_Car = Car()`
`Print(type(my_car))`

`#!/usr/bin/python3`

Class Square:
→ هنا موينا كلاس
Square
سبياه
Pass
→

هتمام أقدر لباليدينا
لرماعنى ليقول أكتب هنا اللي
بس خلني أكمل الكود وما يطبقي خطأ
فالكلاس عمله فامي ولا فيه ولا سطر
لأن زدك `Pass` عشان الباقيون ما يصب علينا 😊

1. Square with size

Score: 100.00% (Checks completed: 100.00%)

- 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 this size value.

السؤال اللي قبل كنا كاتين : Class square: pass

لیٹی الکلاس فامنی

Private size و تكون خامه جديدة أو مهمها الآن في هذا السؤال حروفه ذاته جديدة

نکتہ گلائس ائچے Square بس ہالمرہ ہو فاضی

لذلك يكتن فيه خاصية **attribute** (الخاصية) **Size** و **وخطها بطريقة خاصة** (Private) يعني إذا أردت استخدام الكلاس من ماتقدر بتحمل لمح المروج على طوله . و **وخطه بحثة** (Private) يعني أن المحتوى أو القيمة المكتوب **بمسافة بين كل متغير**

المطلوب من السؤال بـ بـ

1. Square with size

التطبيق

my_square = Square(3)
 له ميوزي هوي هي في قيمه
 my_square = my_square.set_color("red")
 لم يغير الميوزي
 Print(**type(my_square))**
 my_square
 لم يغير الميوزي
 <class 't_square.Square'>
 Print(**my_square.__dict__**)
 له ميوزي هوي هي في قيمه
 my_square
 لم يغير الميوزي
 {'_Square__size': 3}

Class Square:

```
def __init__(self, size):
    self.size = size
    self.square = Square(size)
    self.area = self.size * self.size
```

Self --- Size --- square --- area

نحو امثلة

١- طلب ليت حصلنا بعدين قيل size 99

لذلك ليت تخصيصها عن الذات الى برست هي المكتاب

٢- طلب ليت يحصلنا لها مساحة الذات 99

طبع ليت نشوهها

٣- المساحة 99 المربع احتقر جيم

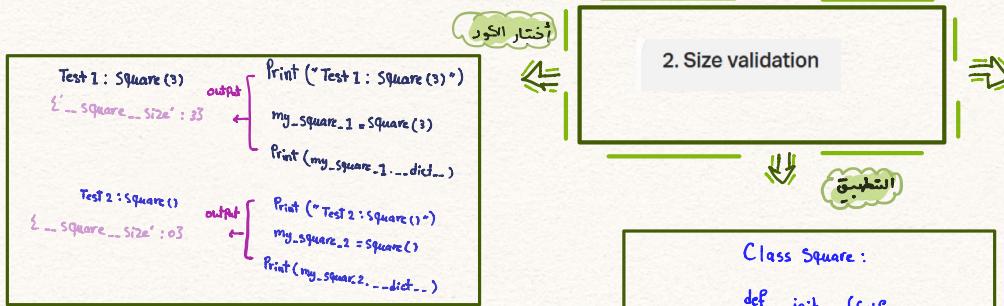
٤- ليت احذر برست او يشوهها دون ذاته

Score: 100.00% (Checks completed: 100.00%)

Write a class `Square` that defines a square by: (based on `1-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`
 - You are not allowed to import any module

المطلوب هنا السؤال



Class Square :

```
def __init__(self, size=0):
    self.size = size

    def area(self):
        return self.size * self.size

    def perimeter(self):
        return 4 * self.size
```

if not isinstance(size, int):

Raise TypeError ("size must be an integer")

elif size < 0:

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

Self -- size = size

لبروگريل كليني تمام مادل

بروكسيت بوكلي فوكس ميغان هد نيرز

نغيره من برا

LEEN ALSALEH

٦٩. وش فرق السؤال اللي قبل عن هذا؟

3. Area of a square

Score: 100.00% (Checks completed: 100.00%)

mandatory

Write a class `Square` that defines a square by:

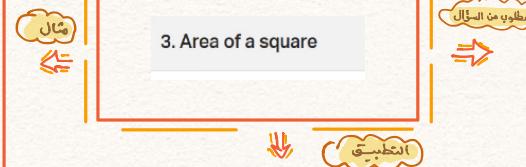
- Private instance attribute: `size`
- Instantiation with optional `size`: def `__init__(self, size=0)`:
 - `size` must be an integer, otherwise raise a `TypeError` exception with the message `size must be an integer`
 - If `size` is less than `0`, raise a `ValueError` exception with the message `size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

السؤال اللي قبل كان يتحقق من قيمة `size` لو هي `>= 0` ويفحظها
هنا الفكرة نفرين والدالة جبارة `area` ترجع مساحة المربع

```
my_square = square (-3) ↗
Print(my_square.area())
↓ output
Size must be >= 0

my_square = square ("49") ↗
Print(my_square.area())
↓ output
Size must be integer

my_square = square (v) ↗
Print(my_square.area())
↓ output
16
```



يبقى من هنا شوي زي السابق نكتب كلام
أنواع square تشن المربع ويكون فيها
أنواع متغير خاص اسمه `size` . فيش جم المربع
لما ننشئه object من الكلاس نكتب متغير `size`
تتأكد أن القيمة اللي داخله عدد صحيح `int` وإذا غيره
أومر ساب يطلع للرسان `ValueError`
بنبني دالة `area` (المساحة) ترجع المساحة دقت المربع

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):
        هذي دالة عاشه افترتنا فيها الملاس ترجع للمساحة
        return self.__size * self.__size

    المساحة = القطع × القطع
```

LEEN ALSALEH

4. Access and update private attribute

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.

Class Square :

```

def __init__(self, size=0):
    @property
    def size(self):
        getsize = property(fget=self.getsize)
        return getsize

    @size.setter
    def size(self, value):
        if not isinstance(size, int):
            raise TypeError("size must be an integer")
        if size < 0:
            raise ValueError("size must be >= 0")
        self.__size = size

    def getsize(self):
        return self.__size

    def setsize(self, value):
        if not isinstance(value, int):
            raise TypeError("size must be an integer")
        if value < 0:
            raise ValueError("size must be >= 0")
        self.__size = value

```

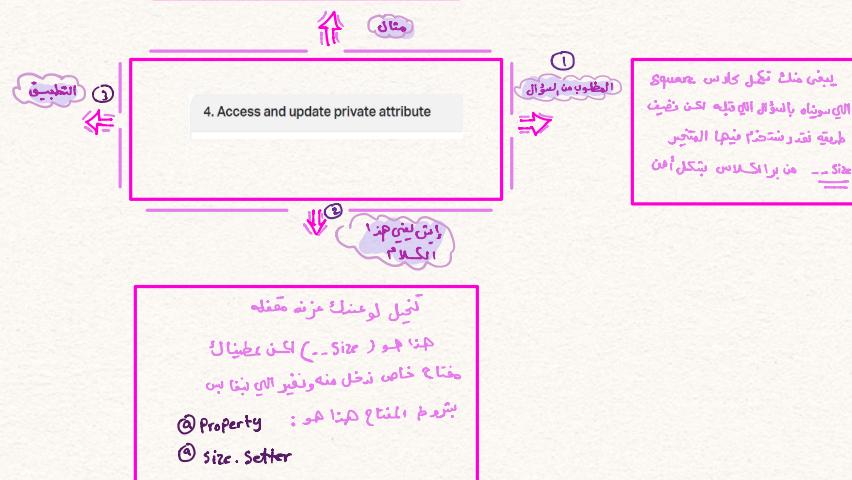
السؤال الذي قبل ما كان يسمى بالتأهيل
أكنت هنا دارج نقدر عمل لها ولطريقه أ منه
عن طريق Setter property عشان تقدر تعدل
ونقدر من هنا الحالات بطيئه

```

my_square = Square(89)
Print(my_square.area())
        ↓ output
89x89 = 7921

my_square.size = 3
Print(my_square.area())
        ↓ output
3x3 = 9
    → Seller C. can
    print this

```



وشن فرق السؤال اللي قبل عند هندا

6. Coordinates of a square

Score: 100.00% (Checks completed: 100.00%)

Write a class `Square` that defines a square by: (based on `5-square.py`)

- Private instance attribute: `size`:
 - property `def size(self)`: to retrieve it
 - property setter `def size(self, value)`: to set it:
 - if `size` is not 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`
- Private instance attribute: `position`:
 - property `def position(self)`: to retrieve it
 - property setter `def position(self, value)`: to set it:
 - `position` must be a tuple of 2 positive integers, otherwise raise a `TypeError` exception with the message `position must be a tuple of 2 positive integers`
- Instantiation with optional `size` and optional `position`: `def __init__(self, size=0, position=(0, 0))`:
- Public instance method: `def area(self)`: that returns the current square area
- Public instance method: `def my_print(self)`: that prints in stdout the square with the character `#`:
 - if `size` is equal to 0, print an empty line
 - `position` should be used by using space - `Don't fill lines by spaces` when `position[1] > 0`
- You are not allowed to import any module

في سؤال كي كنا نطبع مربع عادي بالحجم فقط يعني في موالاته يعني حار فيه ميزه خاصه اسمها `Position`
اللي انتجهت تفهم في مكان طلبه المربع على المساشه كم سطرياتي فوق وكم مساهه قبل السطر

`Square = Square(3, (0,0))`

`Square.my_print()`

ما في سطر

حاري فوق

وتحسانت

من اليسار

`Square = Square(2, (4,2))`

`Square.my_print()`

سطرين

حاري فوق

Position[1] = 2

دلونج مسانت قبل

Position[0] = 4

Class Square:

```
def __init__(self, size=0, position=(0,0)):
```

```
    self.size = size
```

```
    self.position = position
```

@property

```
def size(self):
```

```
    return self.__size
```

@size.setter

```
def size(self, value):
```

if not isinstance(self, Value):

```
Raise TypeError ("size must be an integer")
```

if value < 0:

```
Raise ValueError ("size must be >= 0")
```

```
self.__size = value
```

@position.setter

```
def position(self, value):
```

if not(isinstance(value, tuple) and len(value) == 2 and

all(isinstance(num, int) and num > 0 for num in value)):

Raise TypeError ("position must be a tuple of 2 positive integers")

```
self.__position = value
```

```
def area(self):
```

```
return self.__size * self.__size
```

```
def my.print(self):
```

if self.__size == 0:

```
Print ("")
```

else:

```
Print (" /n " + self.__position[1], end = "")
```

هذا السطر مسؤول عن المسافة اللي فوق `position[1]`

`Position = (3,2)`

كعي ادخل اعليه `##` مرتين لطبع مسافر ما بينه فوق المربع

و محتاجها لاظهر سطرياتي تبديهم من ذلك

`end = ##` في نفس المكان

```
for _ in range(self.__size):
```

لوب يشخون تبدي حجم المربع كل سطرين من المربع يتطبع

هذا مثال `size = 4` دا يطبع 4 اسطر

```
Print (" ## " * self.__position[0] + " ## " * self.__size)
```

طبع علامات `##` حسب حجم المربع

العنوان اللي من الممكن توي

حسب حجم المربع

`Position = (3,2)`

`Size = 4`

`output → #####`

و فراغات + ابريزان

`Square = Square(3, (0,0))`

ما في سطر

حاري فوق

وتحسانت

من اليسار

`Square = Square(2, (4,2))`

سطرين

حاري فوق

Position[1] = 2

دلونج مسانت قبل

Position[0] = 4

6. Coordinates of a square

رسوبي: مثل اكشن تفريغ خاصه جديده
دليلاً مبارئ: `Position` `size`
هذا هو ترتيب (الساقه، المساوه، والمساءه) اللي من الممكن توي
وانت شاكدر ان `size` هي مساهه
كل مساهه لها دلالة `size`
هذا المفهوم تأتمم بـ `my.print()`

نتحقق أ تناهاسكت وورته بيهجا
وأهنا بنتها نرسم مربع بإستخدام عالمه
بين بنتها :

(1) - حجم المربع يكون :

ـ معيكون في الواجهة لتنازله :

- سطرين فارغين فتحة

- وصافتين من الميسار قبل ما يبدا المربع

هناقول زد الحجم = 3 والوضع (2,1)

وشن راح يحصل لامطبع

راغ يحصل في سطرياتي فوق

و دلالة `Position[1] = 1`

ـ كل سطرين فيه سانته قبل

`Position[0] = 2` لـ

فاهردا لخورعنه 3

`Square = square(3,(2,1))`

`Square.my.print()`

