`Q1`: Count number of instances of a class created in Python?

Example: Say `Car` is any class.

```
maruti = Car()
bmw = Car()
honda = Car()
```

So after creating above instances. We want to count how many instances are created of Car class.

For above example `no of instances = 3`.

Write a program for above problem.

```
#write your code here

class Car:

  __counter = 0

  def __init__(self):
    Car.__counter += 1

  def get_counter():
    return Car.__counter

o1 = Car()
o2 = Car()
o3 = Car()
o4 = Car()

print(Car.get_counter())
```

```
4
```

`Q-2`: Create a deck of cards class. Internally, the deck of cards should use another class, a card class. Your requirements are:

- The `Deck` class should have a deal method to deal a single card from the deck
- After a card is dealt, it is removed from the deck.
- There should be a shuffle method which makes sure the deck of cards has all 52 cards and then rearranges them randomly.
- The Card class should have a suit (Hearts, Diamonds, Clubs, Spades) and a value (A,2,3,4,5,6,7,8,9,10,J,Q,K)

`Deck` Class

- It is class of all possible cards in a deck. Total 52 cards.
- Methods - `deal()` it will take out one card from the deck of cards.
- Deck of cards should get shuffeled while creating the deck object.
- `no of cards remaining in deck - <number>` should dsiplay on printing any deck object.

`Card` class

- It is a class of card
- Atrributes - `suit` and `value`
- `<suit> of <value>` should dsiplay on printing any card object.

```
#Code Here
import random

class Card:

  def __init__(self,suit,value):
    self.suit = suit
    self.value = value

  def __repr__(self):
    return "{}->{}".format(self.suit,self.value)

class Deck:

  def __init__(self):
    suits = ['Hearts','Diamonds','Clubs','Spades']
    values = ['A','2','3','4','5','6','7','8','9','10','J','K','Q']
    self.cards = [Card(suit,value) for suit in suits for value in values]

  def __str__(self):
    return "Cards left " + str(len(self.cards))

  def shuffle(self):
    if len(self.cards) < 52:
      print('only full deck can be shuffled')
    else:
      random.shuffle(self.cards)
    return self.cards

  def deal(self):
    if len(self.cards) == 0:
      print('All cards have been dealt')
    return self.cards.pop()


deck = Deck()
deck.shuffle()
print(deck.deal())
print(deck.deal())
print(deck)
```

```
Diamonds->4
Spades->5
Cards left 50
```

Q-3 : Find the area of a rectangle.

Approach:

- The class name should be *Rectangle*.
- The constructor should accept two parameters *length* and *height* but you can't pass the values directly to it while creating the constructor. E.g., rectangle = Rectangle(length=10, height=8) <-- you can't do that while creating the instances.
- Create a method called *area()* which has no parameters.
- Create a method called *is_square()* which also has no parameters. Return True if the rectangle is a square otherwise return False.
- If you are using a if-else block inside the *is_square()* method, then use the one-linear syntax.

```
#Write your code here

class Rectangle:

  def __init__(self,l,b):
    self.length = l
    self.breadth = b

  @classmethod
  def property(cls,len,bre):
    return cls(len,bre)

  def area(self):
    return self.length*self.breadth

  def is_square(self):
    return True if self.length == self.breadth else False

r = Rectangle.property(4,4)
print(r.area())
print(r.is_square())
```

```
16
True
```

Q-4 : Problem 4

Statement: Write a program that uses datetime module within a class. Enter manufacturing date and expiry date of the product. The program must display the years, months and days that are left for expiry.

[ ] ↳ *1 cell hidden*

Q-5 : Problem 5

Statement: A university wants to automate their admission process. Students are admitted based on the marks scored in the qualifying exam. A student is identified by student id, age and marks in qualifying exam. Data are valid, if:

- Age is greater than 20
- Marks is between 0 and 100 (both inclusive)

A student qualifies for admission, if

- Age and marks are valid and
- Marks is 65 or more

Write a python program to represent the students seeking admission in the university. The details of student class are given below.

Class name: Student

Attributes (private)	student_id marks age	
Methods (public)	__init__()	Create and initialize all instance variables to None
	validate_marks()	If data is valid, return true. Else, return false
	validate_age()	

[ ] ↳ *1 cell hidden*

Q-6: Ice-Cream Scoops and Bowl shop

1. Create a class `Scoop` which has one public property `flavor` and one private proptery `price`. Take `flavor` values during object creation.
2. Create a class `Bowl` with private prperty `scoop_list` which will have list of scoopd object.
3. Create a method `add_scoops` in `Bowl` class which will add any no of `Scoop` objects given as parameter and store it in `scoops_list`.
4. Make getter and setter method for `price` property.
5. Make a method `display` to display `flavour` and `price` of each `Scoop` in `scoop_list` and print total price of the bowl by adding all flavour scoops prices.
6. Make a method `sold` in both `Scoop` class and `Bowl` class to print no of quantity sold.

[ ] ↳ *3 cells hidden*

Q-7: Ice-Cream Bowl continue..

Making advancement in the above classes. `Scoop` and `Bowl`

1. Introduce a property `max_scoops` in `Bowl` class to signify maximum scoops that a bowl can have, exceeding that it will display `Bowl is`