Q1. What is the difference between getattr and getattribute?

In [ ]:

```
Ans:__getattribute__ is used to find an attribute of a class. It raises an Attribute
Error of it fails to find an attribute of a class. __getattr__ is implemented latter if
Attribute Error is generated by__getattribute__, but for this __getattribute__ and
__getattr__ both has to be defined in same class.
If no attribute is found, __getattr__ returns a default value. So key difference is
that __getattr__ is called for attributes that don't actually exist on a class.
```

Q2. What is the difference between properties and descriptors?

In [ ]:

```
Ans: The differences between Properties and Descriptors is:

Properties: With Properties we can bind getter, setter and delete functions together
with an attribute name, using the built-in property function or @property decorator.
When we do this, each reference to an attribute looks like simple, direct access, but
involes the appropriate function of the object.

Descriptor: With Descriptor we can bind getter, setter and delete functions into a
seperate class. We then assign an object of this class to the attribute name in our main
class. When we do this, each reference to an attribute looks like simple, direct access
but invokes an appropriate function of descriptor object.
```

Q3. What are the key differences in functionality between getattr and getattribute, as well as properties and descriptors?

In [ ]:

```
Ans:The Key Differences between __getattr__, __getattribute__, Properties and
Descriptors are:
__getattr__: Python will call this method whenever you request an attribute that
hasn't already been defined
__getattribute__ : This method will invoked before looking at the actual attributes
on the object.
Means,if we have __getattribute__ method in our class, python invokes this method for
every attribute regardless whether it exists or not.

Properties: With Properties we can bind getter, setter and delete functions together
with an attribute name, using the built-in property function or @property decorator.
When we do this, each reference to an attribute looks like simple, direct access, but
invokes the appropriate function of the object.

Descriptor: With Descriptor we can bind getter, setter and delete functions into a
seperate class.
we then assign an object of this class to the attribute name in our main class.
When we do this, each reference to an attribute looks like simple, direct access but
involves an appropriate function of descriptor object.
```