Q1. Define the relationship between a class and its instances. Is it a one-to-one or a one-to-many partnership, for example?

A class is a blueprint which you use to create objects. An object is an instance of a class - it's a concrete 'thing' that you made using a specific class. So, 'object' and 'instance' are the same thing, but **the word 'instance' indicates the relationship of an object to its class**

Q2. What kind of data is held only in an instance?

An instance in a database is a combination of the program and memory used to access metadata and application data stored in physical files on a server.

Q3. What kind of knowledge is stored in a class?

About object

Q4. What exactly is a method, and how is it different from a regular function?

**A function doesn't need any object and is independent, while the method is a function, which is linked with any object**. We can directly call the function with its name, while the method is called by the object's name. Function is used to pass or return the data, while the method operates the data in a class.

Q5. Is inheritance supported in Python, and if so, what is the syntax?

It is a mechanism that allows you to create a hierarchy of classes that share a set of properties and methods by deriving a class from another class. Inheritance is the capability of one class to derive or inherit the properties from another class.

Class BaseClass:

{Body}

Class DerivedClass(BaseClass):

{Body}

Q6. How much encapsulation (making instance or class variables private) does Python support?

Q7. How do you distinguish between a class variable and an instance variable?

Class variable an instance variable

|  |  |
| --- | --- |
| It usually reserves memory for data that the class needs. | It usually maintains a single shared value for all instances of class even if no instance object of the class exists. |

Q8. When, if ever, can self be included in a class's method definitions?

*The* self is used to represent the instance of the class*. With this keyword, you can access the attributes and methods of the class in python*

Q9. What is the difference between the \_ \_add\_ \_ and the \_ \_radd\_ \_ methods?

The difference between x. \_\_add\_\_(y) and x. \_\_radd\_\_(y) is that the former calculates x + y whereas the latter calculates y + x.

Q10. When is it necessary to use a reflection method? When do you not need it, even though you support the operation in question?

reflection refers to the ability for code to be able to examine attributes about objects that might be passed as parameters to a function. For example, if we write type(obj) then Python will return an object which represents the type of obj.

Q11. What is the \_ \_iadd\_ \_ method called?

The Python \_\_iadd\_\_() magic method implements in-place addition x += y that adds together the operands and assigns the result to the left operand.

Q12. Is the \_ \_init\_ \_ method inherited by subclasses? What do you do if you need to customize its behavior within a subclass?

By this we can conclude that \_\_init\_subclass\_\_ method is used to alter the behavior of subclasses which may be created in future.