Q1. What is the purpose of Python's OOP?

Ans: Object Oriented Programming is a method of structuring program into multiple modules so that each model have items which are related to each other.

OOPS uses class and objects where class is the blueprint for the model, object is the actual entity which is built on that blueprint.

This OOPS helps us to break down the actual project into multiple problems, which can be easily maintained and modified,

Q2. Where does an inheritance search look for an attribute?

Ans: Inheritance searches for an attribute in the bottom up approach.

If it is able to find in its own object then it will return . If it is not able to find then it moves to 1 level up , and searches up to the root class if the lower levels don’t have the data.

Q3. How do you distinguish between a class object and an instance object?

Ans:

|  |  |
| --- | --- |
| Class | Instance |
| Class is the blueprint of the object | Instance is the individual object based on that class |
| As this a blueprint, it does not consume any memory, save any data | Instance saves the data thus uses memory |
|  | From 1 class there can be multiple instances that can be created |

Q4. What makes the first argument in a class’s method function special?

Ans: the first argument is the pointer argument , which is used to point to class itself.

It can be represented in any way but usually by convention ‘self’ is used to denote the pointer.

Q5. What is the purpose of the \_\_init\_\_ method?

Ans: \_\_init\_\_ is the constructor.

This method is called when an object is created from a class and it allows the class to initialize the attributes of the class.

Q6. What is the process for creating a class instance?

Ans: mycar=Car(‘WagonR’,’Petrol’)

When we want to create an instance of the class we first mention the class name “Car”. Inside the brackets we mention the attributes if any (‘Wagon R’, ‘Petrol’)

Q7. What is the process for creating a class?

Ans: Syntax for creating a class is

class classname:

classvar= 1 # class variable

def \_\_init\_\_(self, key1): # constructor

self.key1= key1 # instance variable

def return key(): #instance method

return self.key1

Q8. How would you define the superclasses of a class?

Ans: A parent class which is not a child class of another class is known as super class.

The parent class will be sent as an argument to the child class so that the child class can inherit all of the attribute and the method of the parent class.

E.g. class child(parent):