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

Ans. OOP stands for object oriented programming , oops shares the same paradigm/purpose in every language as in python. OOPs is not indeed any language it is just the concept to make the code structured, reusable and easily manageable. As mentioned earlier it shares the same purpose with python as it do with other language such as encapsulation, abstraction, inheritance ,polymorphism , code reusability, easily manageable code , well organized code and so on.

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

Ans. The search for an attribute in inheritance goes as:

1. The instance object itself.
2. The class of the instance class
3. In base class/ parent classes

Example:

```
Class A:
        Num =10
Class B(A):
        Pass
Class C(B):
        Pass
# Creating an object of class C
Obj = C()
Obj.num
```

When we search for variable num , first it search for attribute num in "Obj" itself if not present there will look in Class C if not present in C then will look into Parent Class B and similarly in A. if not present in Class A also then will throw an error.

Q3. How do you distinguish between a class object and an instance object?
Ans.In general if we see a class object is template or blueprint for creating instances of objects, while an instance object is a specific occurrence of an object created from that class. The other way around is the object of the class is not proceeded with the self-keyword whereas the instance object is proceeded with the self-keyword e.g. self.anyfunction().

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

Ans. The first argument in class's method is usually "self" , mentioning that self is not any predefined keyword in python however it is a convention to use the self keyword with in the python community itself.

The key points about the special arguments are:

1. The first argument or self (generally used) is actually a pointer which points to the class.
2. It allows to access and manipulate the attributes and methods of the specific instance the class.
3. It's automatically called while passing the arguments we don't need to pass it explicitly.

Note that we can use any keyword instance of self but since its believed that python is an adults language so we should always follow the convention so that it should not be headache for the other developers to follow up the code.

Q5. What is the purpose of the __init__ method?

Ans) 1. __init__ is called the constructor which is called automatically when we create the object of the class. It receives the essential arguments (if required) during the execution of class, generally we use it for the configuration purposes in real world.

Example:
```
Classs exmp:
        def __init__(self, a,b):
            Self.a =a
            Self.b = b
    Obj = exmp(10,20)
```

2. The arguments received in this constructor can be utilized/called in any method of the class.

3. It's not necessary to have __init__ method in the class.

3. The first argument that we should pass in the init method is self.

4. During the time of inheritance there exists couple of cases, mentioned below

Case a) if child class have its own __init__ method then during object creation the __init__ method of child will be called

Case b) if __init__ method not present in child class then the __init__ method of parent will be called during object creation.

Case c) we can initialize/ call /utilize the __init__ methods of parent class as well through different techniques. One using the "Super" keyword ,2nd calling the __init__ method of parent class with in __init__ method of child class.

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

Ans. The purpose of creating the instance of class is to create an object that belongs to a specific class. Although each instance of a class has its own set of attributes and methods.

Creating a class instance allows us to use the different attributes and methods defined with in the class

Q7. What is the process for creating a class?
Ans) The process of creating a class is as follows:

1. Use the "Class" keyword and choose the name of the class using the 'CamelCase' convention.
2. Define the methods and attributes of the class.
3. Once the class is defined or created then we need to create the object of the class as obj = name_of_the_class(). Note obj name can be anything. The we can assess any public and protected method/function or variable using this obj.

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

Ans. The superclass also called parent class by passing the name of the superclass with in parenthesis of the child class as shown below.

```
Class A:
        Pass
Class B(A):
        pass
```

Here A is the super class while as B is the child class.

Also note that the first class which is passed in the parenthesis can overtake or overwrite the methods of other classes , in simple words the methods of first passed class will be preferred.