Q1. What is the purpose of Python's OOP?
A)OOP ensures code modularity, don't repeat yourself principles, hiding unnecessary implementation details from a user using encapsulation, abstraction, inheritance and polymorphism mechanisms.

Q2. Where does an inheritance search look for an attribute?
A) Inheritance searches for an attribute in the first parent/super class mentioned as arguments to the child/sub class and moves to the next if it is not found in the previous super class.

```
class A:
   a1 = 10

class B:
   a1 = 20

class C(A, B):
   def __init__(self):
      print(self.a1)

c = C()
```

The above code returns 10, since a1 = 10 in class A which appears first in the sub class arguments.

Q3. How do you distinguish between a class object and an instance object?
A) class attributes are declared as global variables in a class, while instance attributes are declared in __init__ method of a class. Class attributes can be accessed with or without creating an object, while instance attributes need an instance to access them.

```
class A:
        a = 10
        def __init__(self, b):
                self.b = b
```

In the above example, a is class attribute and b is instance attribute.

A.a returns 10. whereas A.b returns error since instance of the class is not created.

After creating an instance as shown below
aa = A(20)
aa.a retuen 10 and aa.b returns 20

Q4. What makes the first argument in a class's method function special?
A) first argument points to the class/instance itself.

Q5. What is the purpose of the __init__ method?
A) __init__ method acts as a constructor to initialize the attributes once an instance of class is created. It is the first method called upon object creation.

Q6. What is the process for creating a class instance?
A) instance = Class(attributes)

Q7. What is the process for creating a class?
A) class DefaultClass:
    pass

(or)

```
class DefaultClass:
    def __init__(self, a):
        self.a = a
```

Q8. How would you define the superclasses of a class?
A) super class is the parent class from which a parent class inherits the attrbutes and methods. Super classes reduces code duplication and eliminates the don't repeat yourself problem.

```
class SuperClass:
    pass

class SubClass(SuperClass):
    pass
```