Python OOP Assignment

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

--- OOP is used to structure your code in a better way and bind the data and the functions that manipulate the data together and provides us the way to control unwanted access to the data.

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

--- an inheritance search looks at the namespace tree,Bottom up and left to right for an attribute.

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

--- class objects are shared across all the instances of the class and instance objects are unique to every instance created.

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

--- first argument in a class's method is used to represent an active instance of the class internally.generally self keyword is used but any other name other than self can also be used.It is used to access and modify the attributes of an active instance inside the class's method.

Q5. What is the purpose of the init method?

--- the init method acts like a constructor for a class in python and it initializes the attributes for the instances of the class i.e. the instance variables as soon as an instance or an object gets created.

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

--- A class instance can be created as follows:

a.) obj=class(parameter1,parameter2,....) where obj is the class instance

b.) class instances can also be created by using @classmethod decorator, creating a class method within the class itself which does the same thing as done above.

Q7. What is the process for creating a class?

--- the process for creating a class can be as follows:

a.) creating the __init__ method by passing all the instance variables(or attributes of the class objects) as parameters.

b.) various methods can be created which represents various functions that the objects can persorm.so in short it represents the behaviour of the class objects.

c.) various helper functions or utility functions can be created to perform certain tasks which are common and repeatedly done.

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

--- Superclass acts like a parent class for a class from which it inherits certain properties and methods of the superclass.So the class which inherits is known as child class and the class of which properties are inherited is called as superclass.In short, a Parent- child relationship exists between a superclass and class.

Q9. What is the relationship between classes and modules?

--- Classes can be used to build modules in python and are stored in python files(.py) and can be imported as modules where and when required in your code.

classes can have methods and attributes which can be accessed after importing modules and can be used to perform operations,functionalities etc. in your code.

Q10. How do you make instances and classes?

--- instances are made from classes and classes can be build using the CLASS keyword.

e.g.

```
#CREATING A CLASS
class person:
        def __init__(self,name):
                self.name=name
        def print(self):
                print("hello {}".format(self.name))
#CREATING INSTANCES
obj=person("Rahul")
```

Q11. Where and how should be class attributes created?

--- class attributes should be created within the class body and outside any class or instance method or constructors preferably at the very top before the __init__ method.

e.g.

```
class Person:
    attr1="Man"
    def __init__(self,gender,age,height):
```

```
        self.gender=gender
        self.age=age
        self.height=height
```

Q12. Where and how are instance attributes created?

--- instance attributes are preferably created in the __init__ method.so whenever a class with a couple of parameters gets called these parameters gets

passed to the __init__ method as well and the instance variables are created and initialized.All instance variables start with the self keyword and then the dot operator and then comes the variable name.

e.g.
```
class Person:
    attr1="Man"
        def __init__(self,gender,age,height):
            self.gender=gender
            self.age=age
            self.height=height

        cl=Person("M",45,"6ft")//creating instance variables
```

Q13. What does the term "self" in a Python class mean?

--- the term self is used to represent the current instance of the class internally.it is used by the instances of the class to access its attributes and methods.

Q14. How does a Python class handle operator overloading?

--- operator overloading can be handled by overriding special functions like __add__,__mul__,__eq__ etc inside a class.

Q15. When do you consider allowing operator overloading of your classes?

--- Operator overloading is mostly useful when you're making a new class that falls into an existing "Abstract Base Class" (ABC).eg. when you are trying to create a new class for Roman Numbers then it makes sense to allow operator overloading for all the arithmetic operators.

Q16. What is the most popular form of operator overloading?

--- the most popular form of operator overloading is overloading the add(+) operator.

Q17. What are the two most important concepts to grasp in order to comprehend Python OOP code?

--- the two important concepts are objects(instance) and classes.

Q18. Describe three applications for exception processing.

--- a.) used for efficiently handling of runtime exception arising within the programm.

b.) can be used to define user defined exceptions.

c.) used to better organize the code and to structure the code and to make it more robust and immune of any runtime exceptions.

Q19. What happens if you don't do something extra to treat an exception?

--- If you don't handle exceptions properly i.e using try and except blocks then the program execution would stop completely the moments some exception occurs and next statements in the program won't get executed.

Q20. What are your options for recovering from an exception in your script?

--- a.) try to catch that exception using except keyword.

b.) rerun the programm by changing the code in the right way.

Q21. Describe two methods for triggering exceptions in your script.

--- exceptions can be triggered using raise and assert statements.

Q22. Identify two methods for specifying actions to be executed at termination time, regardless of whether or not an exception exists.

--- the two methods are try-finally block and try-except block.

Q23. What is the purpose of the try statement?

--- try statement is used to test the code for exceptions so that any exception occurs that will get caught in the except statement without actually stopping the programm execution abruptly.It is used to make your code exception proof.

Q24. What are the two most popular try statement variations?

--- the two most popular try statement variations are :

a.) try-except

b.) try-except-finally

Q25. What is the purpose of the raise statement?

-- The purpose of the RAISE statement is to raise exceptions manually.

Q26. What does the assert statement do, and what other statement is it like?

--- assert statement evaluates a logical condition and if it turns out to be false then it raises an AssertError and if it is true then normal program execution occurs. It is mainly used for debugging purposes and validating data types of variables.It acts just like an IF-ELSE statement.

Q27. What is the purpose of the with/as argument, and what other statement is it like?

--- with/as argument is used for closing resources right after using them.A function or a class that supports with/as statement is known as context manager.

A context manager allows you to open and close resources right when you want to.It acts just like an try-catch block of code.

Q28. What are *args, **kwargs?

--- *args and **kwargs is used when we do not know the number of incoming function arguments. Hence *args is used to capture all positional arguments in a tuple and **kwargs is used to capture all keyword arguments in a dictionary.

e.g.: def func(a,*args,**kwargs):

```
    b=kwargs["height"]
    funcb(*args,**kwargs)
```

Q29. How can I pass optional or keyword parameters from one function to another?

--- optional arguments can be collected using **kwargs keyword and the same can be passed to another function.**kwargs collects parameters in a dictionary.

Q30. What are Lambda Functions?

--- Lambda functions are one line functions which needs input and return value.they are used in applications like filtering,transforming elements within an iterator etc.

e.g. lst=map(lambda x:x%2==0,[2,3,4,5,22,1,19.18])

Q31. Explain Inheritance in Python with an example?

--- Inheritance in python replicates the very idea of children inheriting values,behaviour and other materialistic things from their parents.When we say class a inherits class b it means that class b acts as a super class and class a as child class and class a inherits properties(attributes) and methods of class b by default.

Q32. Suppose class C inherits from classes A and B as class C(A,B).Classes A and B both have their own versions of method func(). If we call func() from an object of

class C, which version gets invoked?

--- the function func() from class A will get invked accordinh to mro(method resolution order).

Q33. Which methods/functions do we use to determine the type of instance and inheritance?

--- we use the type() and isinstance() functions to determine the type of instance and the inheritance(check whether an object is an instance of some class and one or more parent class,if any, of the same class .

Q34.Explain the use of the 'nonlocal' keyword in Python.

--- nonlocal keyword is used to modify variables in case of nested functions.In case of nested functions,variables which are local to outer functions can be modified in the scope of inner functions using nonlocal keyword.

Q35. What is the global keyword?

--- global keyword is used to modify global variables(defined outside function defination) inside the scope of functions i.e local scope.