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

Ans: It helps implements inheritance, polymorphism, encapsulation in the programming.

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

Ans: An inheritance search looks for an attribute first in the instance object, then in the class the instance was created from then in all higher super classes, progressing from left to right.

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

Ans: The class acts like a blue print, the object is an actual thing that is built based on the blue print. An instance is a virtual copy of the object.

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

Ans: self is not a keyword, but it is used to signify object itself.

Q5. What is the purpose of the init method?

Ans: It’s a constructor , its used to initialize the attributes of the class.

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

Ans: To create instances of a class , you all the class using class name and pass in whatever arguments its \_\_init\_\_ method accepts.

Q7. What is the process for creating a class?

Ans: class keyword is used to create a python class.

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

Ans: A superclass is class from which many sub class can be created.

Q9. What is the relationship between classes and modules?

Ans: Module in python is a simple way to organize the code, and it contains either python classes or just function.

Q10. How do you make instances and classes?

Ans: An instance is an object of the class, while a class is created by using the keyword class.

Q11. Where and how should be class attributes created?

Ans: class attributes are created in the init() , constructor , where we create attributes using the keyword self.

Q12. Where and how are instance attributes created?

Ans: Instance attributes are defined in the constructor, inside the constructor using self parameter.

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

Ans: self represents the instance of the class.

Q14. How does a Python class handle operator overloading?

Ans: Overloading is achieved by overriding the method which is specifically for that operator, in the user-defined class.

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

Ans: Operator overloading is mostly useful when you’re making a new class that falls into an existing “Abstracct Base class”.

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

Ans: Addition operator is the most popular form of operator overloading.

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

Ans: Inheritance and polymorphism are the two most important concepts in order to comprehend python OOP code.

Q18. Describe three applications for exception processing.

Ans: Syntax error, exceptions and logical errors.

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

Ans: Program terminates abruptly and the code past the line that caused the exception will not get executed.

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

Ans: By providing a generic except clause.

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

Ans: Try: this method catches the exceptions raised by the program.

Raise: Triggers an exception manually using custom exceptions.

Q22. Identify two methods for specifying actions to be executed at termination time, regardless of

whether or not an exception exists.

Ans: Finally and else block can handle actions to be executed at the termination time.

Q23. What is the purpose of the try statement?

Ans: The try statement allows you to define a block of code to be tested for errors while it’s is being executed.

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

Ans: else and finally the two most popular try statement variations.

Q25. What is the purpose of the raise statement?

Ans: Raise keyword is used to raise exceptions or errors.

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

Ans: assert keyword id used when debugging code, lets you test if a condition in your code returns true, if not the program will raise an assertion error.

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

Ans: with statement is a replacement for commonly used try/finally error handling statements.

Q28. What are \*args, \*\*kwargs?

Ans: We use \*args and \*\*kwargs as an argument when we are unsure about the number of arguments to pass in the funtions.

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

Ans: To pass optional or keyword parameters from one function to another, collect the arguments using the \* and \*\* specifiers in the function’s parameter list.

Q30. What are Lambda Functions?

Ans: Lamba functions in python is an anonymous function that can take any number of arguments but return only one expression.

Q31. Explain Inheritance in Python with an example?

Ans: Inheritance relationship defines the classes that inherit from other classes as derived, subclass, or sub-type classes.

class Person(object):

    def \_\_init\_\_(self, name, id):

        self.name = name

        self.id = id

    def Display(self):

        print(self.name, self.id)

emp = Person("Satyam", 102)

emp.Display()

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?

Ans: func() from class A gets called.

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

Ans: isinstance() function is used to test whether an object/ variable is an instance of the specified type or lass such as int or list.

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

Ans: The nonlocal keyword is used to work with variables inside nested functions, where the variable should not belong to the inner function.

Q35. What is the global keyword?

Ans : The global keyword is used to create global variable from a no-global scope like inside of a function.