Name:- Manthan Joshi

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

Advanced python programming or scripting is based on Object Oriented Programming Nature,

Where the core python is based on procedural oriented programming nature.

So Advanced python programming majorly depends on the following Object Oriented Programming concepts like

class, Instance, Ecapsulation, Inheritance, Polymorphism.

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

An inheritance search looks for an attribute initially in the instance object, then in the class the instance was created from,

then in all higher parent classes, starting from left to right normally. The search stops at the first place the attribute is found.

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

Class variables can only be assigned when a class has been defined.

Instance variables can be assigned or changed at any time.

Both class variables and instance variables store a value in a program, just like any other Python variable.

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

class method is used to process class variables however the first argument of class method must be 'cls'.

A class method may or may not contain arguments otherthan 'cls'.

Q5. What is the purpose of the init method?

init is a special method to initialize variables. Before and after there are 2 underscores which denotes this method is internal method

that means it cannot access externally.

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

To create instances of a class, we call the class using class name and pass in whatever arguments its __init__ method accepts.

Name:- Manthan Joshi

Q7. What is the process for creating a class?

A class is a collection of instances i.e, objects,where the instances shares the common attributes of class.

```
class classname(object):
variables
def __init__(self):
        self.variablename1 = value
    .
    .
    .
    self.variablenamen = value
    method1()
    method2()
    .
    .
    .
    method n()
```

Classes provide a means of bundling data and functionality together. Creating a new class creates a new type of object,

allowing new instances of that type to be made. Each class instance can have attributes attached to it for maintaining its state.

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

The class from which a class inherits is called the superclass. A class which inherits from a superclass is called a subclass,

also called child class.

```
class A:
    def __init__(self, name):
        self.name = name
    def demo(self):
```

Name:- Manthan Joshi

```
        print("Hi, I am ", self.name)
class B(A):
    pass


x = A("Marvin")
y = B("James")


y.demo()
```

Q9. What is the relationship between classes and modules?

A class is a plan or model used to create instances also called as objects. We can write a class with attriutes and actions.

A module can define functions, classes, and variables. A module can also include runnable code.

Grouping related code into a module makes the code easier to understand and use.

The difference between a class and a module in python is that a class is used to define a plan for a given object,

whereas a module is used to reuse a given piece of code inside another program.

Q10. How do you make instances and classes?

we can create a class by giving the classname and the variables that we declare inside the class are claaed as class variables.

An instance of a class is an object. It is also known as a class object or class instance.

Q11. Where and how should be class attributes created?

The attributes of a class can be declared inside the class i.e, outside the __init__ method.

When you access an attribute via an instance of the class, Python searches for the attribute in the instance attribute list.

Name:- Manthan Joshi

If the instance attribute list doesn’t have that attribute, Python continues looking up the attribute in the class attribute list.

Python returns the value of the attribute as long as it finds the attribute in the instance attribute list or class attribute list.

However, if we access an attribute, Python directly searches for the attribute in the class attribute list.

```
class Student:
        name = "Sehwag"
        course = "Data Analyst"
```

Q12. Where and how are instance attributes created?

Instance attributes are attributes or properties attached to an instance of a class. Instance attributes are defined in the constructor.

```
class Student:
    def __init__(self, name, age):
    self.name = name
        self.age = age
```

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

Self is a parameter or variable or argument of init method. The purpose of 'self' described as when we are creating instance for current class python

allocates memory block on heap. This memory will be allocated to 'self' variable to store initial data of instance variables.

Q14. How does a Python class handle operator overloading?

Python does not support function overloading. Suppose we still want to use the feature of functional overloading.

In that case, we can set the default values of parameters in the method as None, which won't give an error

if that specific value is not passed as an argument while calling the function.

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

The operator overloading in Python means provide extended meaning beyond their predefined operational meaning.

Name:- Manthan Joshi

Such as, we use the "+" operator for adding two integers as well as joining two strings or merging two lists.

We can achieve this as the "+" operator is overloaded by the "int" class and "str" class.

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

the most popular form of operator overloading is the Addition operator. the '+' operator operates on two numbers

and the same operator operates on two strings. It performs "Addition" on numbers whereas it performs "Concatenation" on strings.

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

The two key concepts of OOP which are inheritance and polymorphism.

Both inheritance and polymorphism are key concepts for designing robust, flexible, and easy to maintain the application.

Q18. Describe three applications for exception processing.

The main applications of exception processing is for syntax errors, exceptions and logical errors.

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

if we don't do something to treat exception the program will terminates and the exception will be raised. The program will not be executed successfully.

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

To recover from excetion we have to use try block that we think there the exception will be raised.

Then the after the exception is raised then we divert the exception code block to the except code block. so that we can recover from raising the exception.

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

Try and Raise are the methods that can trigger the exception in the block of code.

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

Name:- Manthan Joshi

Raise and Finally are the two methods can be executed at the termination time, regardless of whether or not an exception exists.

Q23. What is the purpose of the try statement?

The try statement allows us to define a block of code to be tested for errors while it is being executed.

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

The try block lets us test a block of code for errors. The except block lets us to handle the error.

Q25. What is the purpose of the raise statement?

The raise keyword raises an error and stops the control flow of the program. we can use the raise statement without specifying the exception object.

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

The assert keyword is used to debug the code. The assert keyword will test if a condition in our code returns True,

if not, the program will raise an AssertionError. we can also write a message to be written if the code returns False.

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

with argument is used in exception handling to simplify the process of try,except and finally block of codes.

We can also use this argumrnt in the file handling mechanism.

Q28. What are *args, **kwargs?

non-keyword arguments and keyword arguments are to used to declare while calling the function.

These are used for calling the function by giving the values dynamically by the user instead of giving the single value manually to the function.

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

we can declare the **kwargs in one function and we can call another function by giving the same keyword parameters while calling the function.

Name:- Manthan Joshi

Q30. What are Lambda Functions?

A user defined function without a function name is called as anonymous function.

'lambda' is a keyword which is used to create anonymous functions. so, it is called as 'lambda functions'.

it returns result implicitly that means these function can't required return statement. lambda functions contain single expressions.

Q31. Explain Inheritance in Python with an example?

acquiring the properties of parent or base class to the child or derived class properties is called as inheritance.

```
class Person(object):
        def __init__(self, name):
                self.name = name
        def getName(self):
                return self.name
    def isEmployee(self):
            return False


# Inherited or Subclass
class Employee(Person):
        def isEmployee(self):
                return True

emp = Person("Rakesh")
print(emp.getName())

emp = Employee("Prasad")
print(emp.getName())
```

Name:- Manthan Joshi

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 from the class A will be invoked first and printed.

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

The isinstance() method checks whether an object is an instance of a class whereas issubclass() method

asks whether one class is a subclass of another class.

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

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

we use the keyword nonlocal to declare that the variable is not local.

Q35. What is the global keyword?

global keyword will be accessed from anywhere of the function and another function. The variable can be declared by using the global keyword.