Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)

Data Visualization using Python Lab

Academic year 2025 - 2026

Lab Practice Test - III

Faculty: Jamuna S Murthy

Sem : 4D

Part A – Conceptual Questions (MCQ & True/False)

Q1. Which of the following defines an object in Python?

a) A real-world entity
b) Instance of a class
c) A data type
d) A function

Q2. What is inheritance in Python?

a) Wrapping data and functions
b) Hiding implementation details
c) Acquiring properties of another class
d) Creating multiple objects

Q3. True or False: Encapsulation allows direct access to private variables from outside the class.

Q4. Which keyword is used to define a class in Python?

a) function
b) define
c) class
d) object

Q5. Which method is used to initialize an object in Python?

a) __new__()
b) __init__()
c) __str__()
d) __create__()

Q6. True or False: Method overriding is an example of polymorphism.

Q7. What will be the output of the following code?

```
class A:
    def __init__(self):
        self.var = 'A'
class B(A):
    def __init__(self):
        super().__init__()
        self.var = 'B'
obj = B()
print(obj.var)
```

a) A
b) B
c) AB
d) None

Q8. Which of the following correctly describes multiple inheritance?

a) A class inherits from one parent class
b) A class inherits from more than one parent class
c) A class inherits from its own instance
d) A class inherits and overrides its parent methods

Q9. In Python, which of the following is true about abstract classes?

a) Can be instantiated directly
b) Cannot have concrete methods
c) Must inherit from ABC and have at least one abstract method
d) Require __init__ to be abstract

Q10. Which of these is not a valid form of polymorphism in Python?

a) Duck typing
b) Operator overloading
c) Method overriding
d) Class overinitialization

Q11. What will the following code print?

```
class Parent:
    def greet(self):
        print("Hello from Parent")
class Child(Parent):
    def greet(self):
        print("Hello from Child")
p = Parent()
c = Child()
p.greet()
c.greet()
```

a) Parent then Child
b) Child then Parent
c) Both Child
d) Error

Q12. Which access modifier makes a member private in Python?

a) __member
b) _member
c) public
d) #member

Q13. Which of the following best describes encapsulation?

a) Wrapping code into functions
b) Binding data and functions into a single unit and restricting access
c) Providing multiple methods with the same name
d) Using many classes in a single program

Q14. Which feature allows different classes to be treated through the same interface?

a) Inheritance
b) Polymorphism
c) Encapsulation
d) Abstraction

Q15. What is the output of the following code?

```
class A:
    def __init__(self):
        print("A init")
class B(A):
    pass
class C(B):
    pass
c = C()
```

a) C init
b) A init
c) B init
d) No output

Q16. Why is the `super()` function used in Python?

a) To access private methods
b) To call the parent class's constructor or method
c) To define class-level variables
d) To import superclass

Part B – Coding Problems

Q17. Define a class `Car` with attributes `brand`, `model`, and `year`. Create two objects and print their details.

Q18. Write a program that demonstrates single inheritance.

Q19. Create a class with a private attribute and access it using a public method.

Part C – Theory + Code Mix

Q20. Explain Polymorphism with code for method overriding.

Q21. Abstraction using `abc` module:

Q22. Encapsulation vs Abstraction