1BPLC105B/205B PYTHON PROGRAMMING

Module-5: *Object Oriented Programming, Inheritance, Exceptions*

Part A – 5 Marks Questions

Object Oriented Programming

1. Explain why objects are mutable in Python with suitable examples.
2. Differentiate between mutable and immutable objects in Python.
3. What is meant by "Sameness" in Python objects? Explain using == and is operators.
4. Distinguish between equality and identity in Python.
5. Explain aliasing and object references with examples.
6. What is copying in Python? Explain shallow copy with an example.
7. Differentiate between shallow copy and deep copy.
8. Write a Python program demonstrating object copying using the copy module.
9. Explain the importance of copying mutable objects.
10. Discuss object mutability with list and dictionary examples.

Inheritance and Related Concepts

11. Define inheritance in Python. Mention its advantages.
12. Explain single inheritance with a suitable example.
13. Differentiate between base class and derived class.
14. What are pure functions? Explain with an example.
15. Define modifier functions in Python with examples.
16. Differentiate between pure functions and modifier functions.
17. Explain the concept of generalization in object-oriented programming.
18. Write short notes on code reusability using inheritance.
19. What is operator overloading in Python? Explain with examples.
20. Explain polymorphism in Python with suitable examples.
21. Differentiate between compile-time and run-time polymorphism.
22. Explain method overriding with an example.
23. Write a Python program demonstrating operator overloading using + operator.
24. Explain how polymorphism improves flexibility in programs.
25. Discuss the role of inheritance in achieving polymorphism.

Exceptions

26. What is an exception in Python? Explain common runtime exceptions.
27. Explain the need for exception handling in Python.
28. Describe the syntax of try, except, and finally blocks.
29. Write a Python program to handle division-by-zero exception.
30. Differentiate between syntax errors and exceptions.
31. Explain multiple exception handling with an example.
32. What is the purpose of the finally block in exception handling?
33. Explain user-defined exceptions in Python.
34. Write a Python program to raise a custom exception.
35. Discuss the use of raise statement in Python.
36. Explain exception propagation in Python.
37. Write short notes on: **try, except, raise**
38. Explain how exception handling improves program robustness.
39. What happens when an exception is not handled in Python?
40. Write a Python program using nested try-except blocks.

Part B – 10 Marks Questions

Object Oriented Programming

1. Explain object mutability in Python with suitable examples of mutable and immutable objects.
2. Differentiate between equality and identity in Python using == and is operators with examples.
3. Explain the concept of copying in Python. Discuss shallow copy and deep copy with example programs.
4. Write a Python program demonstrating:
    - Object aliasing
    - Sameness
    - Copying of objects
5. Discuss the impact of mutability on function arguments in Python with examples.
6. Explain object references and memory sharing in Python with suitable illustrations.

Inheritance and Related Concepts

7. Explain inheritance in Python with syntax and examples.
8. Write a Python program demonstrating single inheritance and method overriding.
9. Discuss pure functions and modifier functions with suitable examples.
10. Explain generalization in object-oriented programming with real-world examples.
11. Explain operator overloading in Python. Write a program to overload arithmetic operators.
12. Discuss polymorphism in Python with suitable examples.
13. Differentiate between inheritance and polymorphism with examples.
14. Explain run-time polymorphism and method overriding using Python classes.
15. Write a Python program to demonstrate polymorphism using inheritance.
16. Explain how operator overloading supports polymorphism in Python.
17. Discuss the advantages and disadvantages of inheritance in object-oriented programming.
18. Write detailed notes on:
    - Generalization
    - Reusability
    - Extensibility in inheritance

Exceptions

19. Explain exception handling mechanisms in Python with syntax and examples.
20. Write a Python program to demonstrate multiple exception handling and the use of finally.
21. Discuss the hierarchy of exceptions in Python with examples.
22. Explain user-defined exceptions and demonstrate raising custom exceptions with a program.
23. Write a Python program to validate user input using exception handling.
24. Explain the use of try, except, else, and finally blocks with examples.
25. Discuss exception propagation and nested exception handling in Python.
26. Write a Python program to handle file-related exceptions.
27. Explain the importance of exception handling in developing reliable software systems.
28. Compare:
    - Syntax errors
    - Runtime errors
    - Logical errors with suitable examples.
29. Write a Python program demonstrating:
    - Catching exceptions
    - Raising exceptions
    - Custom exception classes
30. Explain how exception handling improves debugging and fault tolerance in Python applications.

****