Topic 1: Classes, Objects, and Methods (Lab 2)

1. **What is a class?** A blueprint or template for creating objects.
2. **What is an object?** An instance of a class that occupies memory.
3. **How do you create an object in Java?** Using the new keyword: ClassName obj = new ClassName();.
4. **What is the difference between an instance variable and a local variable?** Instance variables belong to the class/object; local variables are declared inside a method and exist only during its execution.
5. **What are methods?** Functions defined inside a class that describe the behavior of an object.
6. **What is the return type of a method?** It specifies the type of value the method will return (e.g., int, void, String).
7. **What is method overloading?** Defining multiple methods with the same name but different parameters (number, type, or order).
8. **What is the static keyword?** It indicates that a member (variable or method) belongs to the class rather than a specific object.
9. **Can a static method access instance variables?** No, static methods can only access other static members directly.
10. **What is the purpose of the main method?** It is the entry point where the Java Virtual Machine (JVM) starts execution.
11. **How do you take user input in Java?** By using the Scanner class from the java.util package.
12. **What does the this keyword represent?** It refers to the current object instance.
13. **What is an array of objects?** An array where each element is a reference to an object of a class.

Topic 2: Constructors (Lab 2)

14. **What is a constructor?** A special method used to initialize objects, called automatically when an object is created.
15. **How does a constructor differ from a regular method?** It has no return type and its name must exactly match the class name.
16. **What is a default constructor?** A constructor with no parameters.
17. **What is a parameterized constructor?** A constructor that accepts arguments to initialize fields with specific values.
18. **Can you overload constructors?** Yes, by providing different parameter lists.
19. **What happens if you don't define a constructor in a class?** Java provides a hidden default no-argument constructor.
20. **What is constructor chaining?** Calling one constructor from another within the same class (using this()) or parent class.

Topic 3: Inheritance (Lab 3)

21. **What is inheritance?** A mechanism where one class (subclass) acquires the properties and behaviors of another (superclass).
22. **Which keyword is used for inheritance?** The extends keyword.
23. **What is Single Inheritance?** When one class extends only one other class.
24. **What is Multilevel Inheritance?** A chain of inheritance (e.g., Class C extends B, and B extends A).
25. **What is Hierarchical Inheritance?** When multiple classes extend the same superclass.
26. **Why does Java not support multiple inheritance with classes?** To avoid the "Diamond Problem" (ambiguity) and keep the language simple.
27. **What is the role of the super keyword?** It is used to refer to the immediate parent class's variables, methods, or constructors.
28. **What is the rule for using super() in a constructor?** It must be the very first statement in the subclass constructor.
29. **What is the "Is-A" relationship?** It describes inheritance (e.g., a "Car" *is-a* "Vehicle").

Topic 4: Encapsulation (Lab 3)

30. **What is encapsulation?** Wrapping data (variables) and code (methods) together as a single unit and restricting direct access.
31. **How do you achieve encapsulation?** By making fields private and providing public getter and setter methods.
32. **What are the benefits of encapsulation?** Data security, flexibility (read-only/write-only), and ease of maintenance.
33. **What is data hiding?** Protecting the internal state of an object from unauthorized access by external classes.

Topic 5: Polymorphism and Final (Lab 4)

34. **What is method overriding?** When a subclass provides its own specific implementation of a method already defined in the superclass.
35. **What is the @Override annotation?** A hint to the compiler that the method is intended to override a parent method; it helps catch errors.
36. **What is Compile-time Polymorphism?** Achieved through method overloading.
37. **What is Runtime Polymorphism?** Achieved through method overriding and dynamic method dispatch.
38. **What is Dynamic Method Dispatch?** The process where a call to an overridden method is resolved at runtime based on the object type.
39. **Difference between Reference Type and Object Type?** Reference type is the class of the variable (e.g., Parent p), while Object type is the actual instance created (e.g., new Child()).
40. **What happens if you declare a variable as final?** It becomes a constant and its value cannot be changed once initialized.

41. **What is a final method?** A method that cannot be overridden by subclasses.
42. **What is a final class?** A class that cannot be inherited (extended).

Topic 6: Abstraction and Interfaces (Lab 4)

43. **What is abstraction?** The process of hiding internal details and showing only the essential features of an object.
44. **What is an abstract class?** A class declared with the abstract keyword that cannot be instantiated and may contain abstract methods.
45. **What is an abstract method?** A method without a body (no implementation) that must be overridden by a subclass.
46. **Can an abstract class have a constructor?** Yes, it is used to initialize common fields during subclass instantiation.
47. **What is an interface?** A blueprint of a class that contains only abstract methods and static constants (provides 100% abstraction).
48. **Which keyword is used to use an interface?** The implements keyword.
49. **Can a class implement multiple interfaces?** Yes, this is how Java handles multiple inheritance of behavior.
50. **What is the "Diamond Problem" in interfaces?** It occurs when two interfaces have the same default method; the class implementing them must override the method to resolve the conflict.