* **What is Full Stack Development?**
  + **Answer:** Full Stack Development involves building both frontend (client-side) and backend (server-side) of an application.
* **How does a programming language communicate with machines?**
  + **Answer:** Programming languages are converted into machine language (binary code) using compilers or interpreters for execution.
* **Who created Java, and who owns it now?**
  + **Answer:** Java was created by James Gosling at Sun Microsystems, and it is now owned by Oracle Corporation.
* **Why is Java platform-independent?**
  + **Answer:** Java programs are compiled into bytecode, which can be executed on any platform that has a JVM (Java Virtual Machine).
* **What is garbage collection in Java?**
  + **Answer:** Garbage collection is the process of automatically reclaiming memory by removing unused objects.
* **What is the purpose of JDK, JRE, and JVM?**
  + **Answer:**
    1. **JDK:** Contains tools for developing Java applications.
    2. **JRE:** Provides runtime libraries for running Java programs.
    3. **JVM:** Executes Java bytecode and provides platform independence.
* **How do you compile and run a Java program?**
  + **Answer:**
    1. Compile: javac filename.java
    2. Run: java classname.
* **What command do you use to check the Java version installed?**
  + **Answer:** Use java -version.
* **What are the main editions of Java, and what are they used for?**
  + **Answer:**
    1. **JSE (Java Standard Edition):** For standalone/desktop applications.
    2. **JEE (Java Enterprise Edition):** For web and distributed applications.
    3. **JME (Java Micro Edition):** For mobile applications.
* **Give examples of applications for each Java edition.**
  + **Answer:**
    1. **JSE:** Calculator, Eclipse, VLC Media Player.
    2. **JEE:** Gmail, Instagram, YouTube.
    3. **JME:** Android apps.
* **What are identifiers in Java?**
  + **Answer:** Identifiers are names for variables, classes, and methods, composed of letters (a-z, A-Z), digits (0-9), underscores (\_), and dollar signs ($).
* **What are local and global variables in Java?**
  + **Answer:**
    1. **Local:** Defined inside a method and only accessible within it.
    2. **Global:** Declared inside a class; can be static or instance variables.
* **What is the syntax for declaring a variable in Java?**
  + **Answer:** datatype varname = value; (e.g., int eid = 123;).
* **What are primitive and non-primitive data types in Java?**
  + **Answer:**
    1. **Primitive:** Single values (e.g., byte, int, char).
    2. **Non-primitive:** Objects and arrays that store multiple values.
* **What is the range of** byte **data type in Java?**
  + **Answer:** -128 to 127.
* **What is the difference between** float **and** double**?**
  + **Answer:**
    1. **Float:** 4 bytes, precision up to 6-7 digits.
    2. **Double:** 8 bytes, precision up to 15-16 digits.
* **What is the purpose of methods in Java?**
  + **Answer:** Methods promote code reusability and modularity.
* **What is the syntax for defining a method?**
  + **Answer:**
  + accessModifier static/instance returnType methodName(parameters) {
    1. // method body
  + }
* **What are control structures in Java?**
  + **Answer:**
    1. **Conditional:** if, else, switch.
    2. **Looping:** for, while, do-while.
* **What is the default package for Java classes?**
  + **Answer:** java.lang.
* **What is the difference between instance and static variables?**
  + **Answer:**
    1. **Instance:** Memory allocated when the object is created.
    2. **Static:** Memory allocated once, at the time of class loading.
* **How can static properties be accessed in Java?**
  + **Answer:** By using the class name (e.g., ClassName.propertyName).
* **What are the four pillars of OOPS?**
  + **Answer:** Encapsulation, Abstraction, Inheritance, Polymorphism.
* **What is the difference between an abstract class and an interface?**
  + **Answer:**
    1. **Abstract Class:** Can contain both abstract and concrete methods.
    2. **Interface:** Contains only abstract methods (before Java 8).
* **How does Java support multiple inheritance?**
  + **Answer:** Through interfaces.
* **Why is exception handling important?**
  + **Answer:** It prevents abnormal program termination and provides user-friendly error handling.
* **What are the keywords used in exception handling?**
  + **Answer:** try, catch, finally, throw, throws.
* **What is the advantage of Collections over arrays?**
  + **Answer:** Collections are growable, can handle heterogeneous data, and provide utility methods for data manipulation.
* **What are the main interfaces in the Collections framework?**
  + **Answer:** List, Set, Queue.
* **What is the difference between** ArrayList **and** LinkedList**?**
  + **Answer:**
    1. **ArrayList:** Better for random access.
    2. **LinkedList:** Better for insertions and deletions.
* **What are lambda expressions in Java?**
  + **Answer:** Lambda expressions provide a concise way to write anonymous functions using (parameters) -> {expression/block}.
* **What is the purpose of Stream API?**
  + **Answer:** Stream API simplifies operations on collections like filtering, mapping, and reducing.
* **What is the Optional class?**
  + **Answer:** It helps in avoiding NullPointerExceptions.
* **What is the JVM architecture?**
  + **Answer:** The JVM consists of the class loader, memory areas (heap, stack), and the execution engine.
* **What is the significance of** System.out.println()**?**
  + **Answer:** It is used to print output to the console.
* **What are the four features of OOPS in Java?**
  + **Answer:**
    1. **Inheritance:** Allows a class to acquire properties of another class.
    2. **Polymorphism:** Allows methods to take multiple forms (e.g., method overriding and overloading).
    3. **Abstraction:** Hides implementation details and exposes only the functionality.
    4. **Encapsulation:** Binds data and methods into a single unit for security and modularity.
* **What is the difference between inheritance and multiple inheritance in Java?**
  + **Answer:**
    1. **Inheritance:** Achieved using the extends keyword for classes and interfaces.
    2. **Multiple Inheritance:** Not supported through classes but achieved using interfaces.
* **Why does Java not support multiple inheritance through classes?**
  + **Answer:** To avoid ambiguity caused by the diamond problem.
* **What is abstraction, and how is it achieved in Java?**
  + **Answer:**
    1. Abstraction hides implementation details.
    2. Achieved through abstract classes and interfaces.
* **What is encapsulation? Provide an example.**
  + **Answer:** Encapsulation wraps data (fields) and methods in a single unit (class). Example:
* java
* class Employee {
* private int empId;
* private String empName;
* public void setEmpId(int id) { empId = id; }
* public int getEmpId() { return empId; }
* }
* **What is the JVM, and why is it important for Java?**
  + **Answer:** JVM (Java Virtual Machine) is a runtime environment that executes bytecode and provides platform independence.
* **What are the roles of JDK, JRE, and JVM?**
  + **Answer:**
    1. **JDK:** Provides tools for Java development (compiler, debugger).
    2. **JRE:** Offers runtime libraries for Java applications.
    3. **JVM:** Executes Java bytecode and manages memory.
* **How do you compile and run a Java program?**
  + **Answer:**
    1. Compile: javac filename.java
    2. Run: java classname.
* **What are primitive and non-primitive data types?**
  + **Answer:**
    1. **Primitive:** byte, short, int, long, float, double, char, boolean.
    2. **Non-primitive:** Strings, arrays, custom objects.
* **What is the difference between local and global variables in Java?**
  + **Answer:**
    1. **Local:** Defined inside methods and accessible only within that scope.
    2. **Global:** Defined inside the class and accessible throughout.
* **What is method overloading and overriding?**
  + **Answer:**
    1. **Overloading:** Same method name, different parameters.
    2. **Overriding:** Redefining a parent class method in the child class.
* **What are the key components of exception handling in Java?**
  + **Answer:**
    1. try: Contains risky code.
    2. catch: Handles exceptions.
    3. finally: Executes cleanup code.
    4. throw: Used to throw exceptions.
    5. throws: Declares exceptions.
* **What is the difference between checked and unchecked exceptions?**
  + **Answer:**
    1. **Checked:** Exceptions checked at compile time (e.g., IOException).
    2. **Unchecked:** Exceptions checked at runtime (e.g., NullPointerException).
* **What is the advantage of Collections over arrays?**
  + **Answer:**
    1. Collections are dynamic in size, support heterogeneous data, and provide utility methods.
* **What are the main interfaces in the Collections framework?**
  + **Answer:**
    1. **List:** Allows duplicates; maintains order.
    2. **Set:** No duplicates; no order.
    3. **Queue:** FIFO order.
* **Explain the difference between** Comparable **and** Comparator**.**
  + **Answer:**
    1. **Comparable:** Provides natural sorting (compareTo()).
    2. **Comparator:** Provides custom sorting (compare()).
* **What are the major Java 8 features?**
  + **Answer:** Lambda expressions, functional interfaces, Stream API, Optional class, default/static methods, Date/Time API.
* **What are lambda expressions in Java 8?**
  + **Answer:** Lambda expressions simplify anonymous method creation using (parameters) -> {expression/block}.
* **What is the Optional class in Java 8?**
  + **Answer:** A container object used to handle nullable values to avoid NullPointerException.
* **What is the Stream API?**
  + **Answer:** Provides methods for filtering, mapping, and reducing data collections.
* **What is Spring Framework, and why is it lightweight?**
  + **Answer:** Spring is a lightweight framework for Java applications that provides dependency injection and simplifies application development.
* **What is Dependency Injection in Spring?**
  + **Answer:** A design pattern where the Spring container manages object creation and wiring dependencies.
* **What are the main modules in Spring Framework?**
  + **Answer:** Spring Core, Spring JDBC, Spring ORM, Spring MVC, Spring AOP.
* **What is the difference between BeanFactory and ApplicationContext?**
  + **Answer:**
    1. **BeanFactory:** Provides basic dependency injection.
    2. **ApplicationContext:** Provides additional features like event handling and AOP.
* **What are the advantages of Spring Boot?**
  + **Answer:** Provides auto-configuration, embedded servers, starters, Actuator, and production-ready applications.
* **What are Spring Boot starters?**
  + **Answer:** Pre-defined dependencies for specific functionalities like spring-boot-starter-data-jpa.
* **What is the role of Actuators in Spring Boot?**
  + **Answer:** Actuators provide endpoints to monitor and manage Spring Boot applications.
* **How does** application.properties **work in Spring Boot?**
  + **Answer:** It stores configuration properties like database connection, server port, and logging settings.
* **What are the steps in JDBC to connect to a database?**
  + **Answer:**
    1. Load the driver class.
    2. Create a connection.
    3. Create a statement.
    4. Execute the query.
    5. Close the connection.
* **What are the disadvantages of traditional JDBC?**
  + **Answer:** Boilerplate code, exception handling, and tight coupling.
* **How does Spring JDBC simplify database access?**
  + **Answer:** By eliminating boilerplate code and enabling loosely coupled operations using JdbcTemplate.
* **What is JPA, and how does it differ from JDBC?**
  + **Answer:**
    1. **JPA:** Java Persistence API automates database mapping and eliminates the need for SQL queries.
    2. **JDBC:** Requires manual SQL handling.
* **What are ORM tools, and how do they work?**
  + **Answer:** ORM tools like Hibernate map Java objects to database tables and vice versa.
* **What are SOLID principles in software development?**
  + **Answer:**
    1. **SRP:** Single Responsibility Principle.
    2. **OCP:** Open/Closed Principle.
    3. **LSP:** Liskov Substitution Principle.
    4. **ISP:** Interface Segregation Principle.
    5. **DIP:** Dependency Inversion Principle.
* **What is Agile methodology?**
  + **Answer:** Agile is an iterative approach to software development that emphasizes flexibility, collaboration, and incremental delivery.
* **What is the Single Responsibility Principle (SRP)?**
  + **Answer:** A class should only have one reason to change, meaning it should have one well-defined responsibility.
* **Explain the Open/Closed Principle (OCP).**
  + **Answer:** Classes should be open for extension but closed for modification. This means you can add new functionality but should not change existing code.
* **What is the Liskov Substitution Principle (LSP)?**
  + **Answer:** Subtypes should be substitutable for their base types without altering the behavior of the program.
* **What is the Interface Segregation Principle (ISP)?**
  + **Answer:** Clients should not be forced to depend on interfaces they do not use.
* **Explain the Dependency Inversion Principle (DIP).**
  + **Answer:** High-level modules should not depend on low-level modules; both should rely on abstractions.
* **What is Spring Boot?**
  + **Answer:** Spring Boot is a framework for building production-ready applications with minimal configurations and default settings.
* **What are the main features of Spring Boot?**
  + **Answer:**
    1. AutoConfiguration
    2. Starters
    3. Embedded server
    4. Actuators
    5. Inbuilt database support (H2)
* **What is AutoConfiguration in Spring Boot?**
  + **Answer:** Automatically configures most commonly used classes based on dependencies in the classpath.
* **What are Spring Boot starters?**
  + **Answer:** Starter packages simplify dependency management by including pre-configured libraries for specific functionalities (e.g., spring-boot-starter-web).
* **What is the role of Actuators in Spring Boot?**
  + **Answer:** Actuators provide endpoints to monitor and manage your Spring Boot application (e.g., health, metrics).
* **What are lambda expressions in Java 8?**
  + **Answer:** Lambda expressions provide a concise way to define anonymous functions using (parameters) -> expression.
* **What are functional interfaces in Java 8?**
  + **Answer:** Interfaces with a single abstract method, e.g., Predicate, Supplier, Consumer, Function.
* **What is the Stream API in Java?**
  + **Answer:** The Stream API simplifies data processing from collections with methods like filter(), map(), reduce().
* **What is the purpose of the Optional class in Java?**
  + **Answer:** Helps avoid NullPointerException by managing nullable values.
* **What are default and static methods in Java interfaces?**
  + **Answer:** Default methods allow implementations in interfaces, and static methods provide utility methods inside interfaces.
* **How would you design a system to fetch products priced above a given threshold using Spring JPA?**
  + **Answer:**
    1. Create an Entity for Product with fields like pid, pname, price, category, quantity.
    2. Use a custom query in the Repository:
* java
* @Query("SELECT p FROM Product p WHERE p.price > :price")
* List<Product> findProductsGreaterThanPrice(@Param("price") int price);
* **What are the benefits of using JPA over JDBC?**
  + **Answer:** JPA automates SQL query generation, offers cache support, and provides ORM (Object Relational Mapping).
* **What is microservices architecture?**
  + **Answer:** A design pattern where an application is divided into small independent services that communicate with each other.
* **What is the role of Eureka Server in microservices?**
  + **Answer:** Eureka acts as a service registry for microservices to discover and communicate with each other.
* **What is API Gateway in microservices?**
  + **Answer:** It serves as a single entry point for routing client requests to the appropriate microservices.
* **What is Hystrix in microservices?**
  + **Answer:** Hystrix is a fault-tolerant library used to handle failures in microservices.
* **What is Test-Driven Development (TDD)?**
  + **Answer:** TDD is a methodology where test cases are written before the actual functionality is implemented.
* **How does JUnit facilitate testing?**
  + **Answer:** JUnit provides annotations (@Test) and assertion methods (assertEquals) for defining and verifying test cases.
* **What are the main components of JUnit 5?**
  + **Answer:**
    1. JUnit Jupiter: For writing tests.
    2. JUnit Vintage: For backward compatibility.
    3. JUnit Platform: For launching test frameworks.
* **What is Project Lombok, and what are its benefits?**
  + **Answer:** Lombok eliminates boilerplate code for getters, setters, constructors, etc., using annotations like @Getter, @Setter, @Data.
* **Explain the significance of the** @Data **annotation in Lombok.**
  + **Answer:** Combines @Getter, @Setter, @ToString, @EqualsAndHashCode, and @RequiredArgsConstructor.
* **What is SLF4J, and why is it used?**
  + **Answer:** SLF4J (Simple Logging Facade for Java) provides a generic logging interface and allows switching between logging frameworks like Log4j.
* **What are the levels of logging in SLF4J?**
  + **Answer:**
    1. info
    2. warn
    3. debug
    4. error
    5. fatal.
* **What are repositories in Spring Data JPA?**
  + **Answer:** Repositories provide CRUD operations on entities and include interfaces like JpaRepository, CrudRepository.
* **What is the use of the** @Query **annotation in Spring Data JPA?**
  + **Answer:** It allows writing custom queries in the Repository interface.
* **What is REST, and how does Spring Boot implement it?**
  + **Answer:** REST (Representational State Transfer) is a web service technology using HTTP methods. Spring Boot implements REST using annotations like @RestController and @RequestMapping.
* **What is the difference between SOAP and REST?**
  + **Answer:**
    1. **SOAP:** Uses XML for communication; protocol-based.
    2. **REST:** Lightweight, uses JSON/XML; architectural style.
* **What are common formats used in RESTful services?**
  + **Answer:** JSON, XML, HTML.
* **What is Git, and how does it differ from SVN?**
  + **Answer:**
    1. **Git:** Distributed Version Control System (DVCS).
    2. **SVN:** Centralized Version Control System.
* **What are the common Git commands?**
  + **Answer:**
    1. git init
    2. git clone
    3. git add
    4. git commit
    5. git push
    6. git pull.
* **What types of validations are used in Spring Boot?**
  + **Answer:**
    1. Field-level validations using annotations like @NotNull, @Size.
    2. Controller-level validations using exception handlers.
* **What is JSON Web Token (JWT)?**
  + **Answer:** JWT is a compact, secure token used for sharing information between parties in a stateless manner.
* **What are the components of a JWT?**
  + **Answer:**
    1. Header: Metadata like the type of token and algorithm.
    2. Payload: User claims.
    3. Signature: Ensures integrity.
* **How is JWT used for authentication?**
  + **Answer:** A JWT is issued to a user upon login and is sent with each request for validation.