QUSTION BANK-- ADVANCED JAVA

BASIC JAVA CONCEPTS

1. Q: What is the Java Virtual Machine (JVM)?

   A: The JVM is a virtual machine that enables a computer to run Java programs. It converts Java bytecode into machine language and executes it.

2. Q: What is the difference between JDK, JRE, and JVM?

   A: JDK (Java Development Kit) is a software development kit for developing Java applications. JRE (Java Runtime Environment) provides libraries, Java Virtual Machine (JVM), and other components to run applications written in Java. JVM (Java Virtual Machine) is the engine that runs Java applications.

3. Q: Explain the concept of Object-Oriented Programming (OOP).

   A: OOP is a programming paradigm based on the concept of objects, which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).

4. Q: What are the main principles of OOP?

   A: The main principles of OOP are Encapsulation, Inheritance, Polymorphism, and Abstraction.

5. Q: What is a class in Java?

   A: A class is a blueprint for creating objects, providing initial values for state (member variables or fields) and implementations of behavior (member functions or methods).

6. Q: What is an object in Java?

   A: An object is an instance of a class. It contains both data (attributes) and behavior (methods).

7. Q: What is inheritance in Java?

A: Inheritance is a mechanism where a new class inherits properties and behavior (fields and methods) from an existing class.

8. Q: What is polymorphism in Java?

A: Polymorphism allows methods to do different things based on the object it is acting upon, even though they share the same name.

JDBC ARCHITECTURE

9. Q: What is JDBC?

A:JDBC (Java Database Connectivity) is an API that enables Java applications to interact with databases. It provides a set of classes and interfaces for establishing a connection, sending SQL queries, and processing the results.

10. Q: What are the main steps in the JDBC process?

A:The main steps include:

Load and register the driver.

Establish a connection to the database.

Create a Statement or PreparedStatement object.

Execute SQL queries using executeQuery() or executeUpdate().

Process the result set.

Close the connection and resources.

11. Q: What is the difference between Statement and PreparedStatement?

A: *Statement*: Used for executing simple SQL queries without parameters. It is prone to SQL injection attacks.

PreparedStatement: Used for executing parameterized SQL queries. It precompiles the query, improving performance and security by preventing SQL injection.

12. Q: How do you load a JDBC driver?

A:You load a JDBC driver using the Class.forName() method. For example:

Class.forName("com.mysql.jdbc.Driver");

13.Q:What is a ResultSet in JDBC?

A: A ResultSet is an object that holds the data returned from a database after executing a SQL query. You can navigate through its rows and access the columns of each row.

14. Q: How do you handle SQL exceptions in JDBC?

A: SQL exceptions in JDBC are handled using a try-catch block. The SQLException class provides useful methods such as getMessage(), getErrorCode(), and getSQLState() to handle errors.

15. Q:What are JDBC transactions, and how do you manage them?

A: JDBC transactions allow you to group multiple SQL operations as a single unit of work. Transactions can be controlled using:

setAutoCommit(false) to disable auto-commit.

commit() to save changes.

rollback() to undo changes in case of failure.

16. Q: What is JDBC batch processing?

A: JDBC batch processing allows you to group multiple SQL statements into a batch and execute them together to improve performance. It reduces the number of database round trips. Example:

PreparedStatement pstmt = conn.prepareStatement("INSERT INTO users VALUES (?, ?)");

pstmt.setString(1, "Alice");

pstmt.setInt(2, 25);

pstmt.addBatch();

pstmt.executeBatch();

17. Q: What is the difference between executeQuery(), executeUpdate(), and execute()?

A:

executeQuery(): Used for executing SELECT queries that return a ResultSet.

executeUpdate(): Used for executing INSERT, UPDATE, DELETE statements. It returns the number of affected rows.

execute(): Used for executing any SQL statement (query or update). It returns a boolean: true if the result is a ResultSet, false otherwise.

18. Q: What are JDBC drivers, and how many types are there?

A: JDBC drivers are classes that implement the JDBC API to connect to a database. There are four types of JDBC drivers:

Type 1: JDBC-ODBC Bridge Driver.

Type 2: Native API Driver.

Type 3: Network Protocol Driver.

Type 4: Thin Driver (Pure Java Driver).

GENERICS & COLLECTION FRAMEWORK APIs

19. Q:What are Generics in Java and why are they used?

A: Generics enable types (classes and methods) to be parameterized. By using generics, you can write a single method or class that works with different types of data, ensuring type safety at compile-time. This prevents ClassCastException and eliminates the need for casting in many cases.

20. Q: What is the difference between List<?> and List<? extends T>?

A:

List<?> is a wildcard that can represent a list of any type.

List<? extends T> limits the wildcard to classes that are subclasses (or same as) of T. This is useful for methods that can work with any subclass of a given type but don't modify the list.

21. Q: Explain the PECS principle in Generics.

A:

PECS stands for Producer Extends, Consumer Super:

Use <? extends T> when you need to read from a collection (because the collection is producing items of type T).

Use <? super T> when you need to write to a collection (because the collection is consuming items of type T).

22. Q: What is a Type Erasure in Generics?

A: Type Erasure is the process by which generic types are replaced with their raw types during compilation. For example, List<Integer> becomes List at runtime. Type information is erased to ensure compatibility with legacy code (pre-Java 5).

23. Q: How does the HashMap work in the Collection Framework?

A: HashMap stores key-value pairs, where the key is used to determine the bucket (via hashing). The key's hashCode() determines the bucket index, and collisions are resolved using a linked list or a balanced tree if too many keys have the same hash value.

24. Q: What is the difference between ArrayList and LinkedList?

A:

ArrayList: Implements a dynamic array; good for random access (O(1) for get()), but slow for insertions/deletions in the middle (O(n)).

LinkedList: Implements a doubly linked list; good for insertions/deletions (O(1)), but slower for random access (O(n)).

25. Q: What is the purpose of Comparator and Comparable interfaces?

A:

Comparable: Objects that implement this interface can be compared using their natural order (e.g., Integer sorts numbers in ascending order). It uses the compareTo() method.

Comparator: Allows you to define custom ordering of objects. It is implemented in a separate class and uses the compare() method.

26. Q: What is the difference between Set and List?

A:

Set: A collection that does not allow duplicate elements. Common implementations include HashSet (no order), LinkedHashSet (maintains insertion order), and TreeSet (sorted).

List: An ordered collection that allows duplicates and maintains insertion order. Common implementations include ArrayList and LinkedList.

27. Q: How does ConcurrentHashMap differ from HashMap?

A:

HashMap is not thread-safe. If multiple threads access it simultaneously without proper synchronization, it may result in inconsistent data.

ConcurrentHashMap is thread-safe and uses a mechanism of lock-striping, where only parts of the map are locked during updates, thus providing better concurrency than synchronized HashMap.

28. Q: What is the role of Wildcards in Generics?

A: Wildcards allow you to pass different types into a generic class or method while still ensuring type safety. There are three types of wildcards:

? (unbounded wildcard): Accepts any type.

? extends T (bounded wildcard): Accepts T or any of its subtypes.

? super T (lower-bounded wildcard): Accepts T or any of its supertypes.

SOCKET PROGRAMMING

29.Q: What is socket programming in Java?

A: Socket programming in Java is a way of connecting two nodes on a network to communicate with each other. One socket listens on a particular port at an IP, while another socket reaches out to the other to form a connection.

30.Q: What is the difference between TCP and UDP?

A: TCP (Transmission Control Protocol) is connection-oriented and provides reliable communication. UDP (User Datagram Protocol) is connectionless and provides a faster, but less reliable communication.

31. Q: How do you create a server socket in Java?

A:

```
ServerSocket serverSocket = new ServerSocket(portNumber);
Socket clientSocket = serverSocket.accept();
```

32. Q: How do you create a client socket in Java?

A:

```
Socket socket = new Socket("hostname", portNumber);
```

33. Q: How do you send data through a socket?

A:

```
OutputStream outputStream = socket.getOutputStream();
PrintWriter out = new PrintWriter(outputStream, true);
out.println("Hello, World!");
```

34. Q: How do you receive data through a socket?

A:

```
InputStream inputStream = socket.getInputStream();
BufferedReader in = new BufferedReader(new
InputStreamReader(inputStream));
String response = in.readLine();
```

35. Q: What is the ServerSocket class used for?

A: ServerSocket is used to create a server that listens for incoming client connections.

36. Q: How do you close a socket in Java?

A: socket.close();

37. Q: What is a DatagramSocket in Java?

A: DatagramSocket is used to create a socket for sending and receiving datagram packets in UDP communication.

38. Q: How do you send a datagram packet in Java?

A:

```
DatagramSocket socket = new DatagramSocket();
```

```
byte[] buffer = "Hello".getBytes();
InetAddress address = InetAddress.getByName("hostname");
DatagramPacket packet = new DatagramPacket(buffer, buffer.length,
address, port);
socket.send(packet);
```

JSP (JAVA SERVER PAGES)

39. Q: What is JSP?
A: JSP (JavaServer Pages) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types, using Java.

40. Q: Describe the lifecycle of a JSP page.
A: The lifecycle includes Translation, Compilation, Initialization, Execution, and Cleanup.

41. Q: What are JSP directives?
A: JSP directives provide global information about an entire JSP page and are used to set page-level instructions. The main directives are page, include, and taglib.

42. Q: What is a JSP scriptlet?
A: A JSP scriptlet is a piece of Java code embedded within the HTML code in a JSP page, denoted by <% %> tags.

43. Q: How do you include a static file in a JSP page?
A: <%@ include file="header.html" %>

44. Q: What is a JSP expression?
A: A JSP expression is used to output Java values directly into the HTML, denoted by <%= %> tags.

45. Q: How do you handle exceptions in JSP?
A: You can use the errorPage attribute of the page directive to specify an error page:
<%@ page errorPage="error.jsp" %>

46. Q: What are JSP implicit objects?

A: JSP implicit objects are pre-defined variables that provide access to various objects related to the web environment. Examples include request, response, session, application, out, config, pageContext, page, and exception.

47.Q: What is the role of the jsp:include action tag?
A: The jsp:include action tag includes the content of another resource (like a JSP page or HTML file) at runtime.

48.Q: What is the difference between jsp:include and the include directive?
A: The include directive (<%@ include %>) includes content at page translation time, whereas the jsp:include action tag includes content at request time.

SERVLETS

49.Q: What is a Servlet in Java?
A: A Servlet is a Java class that is used to extend the capabilities of servers hosting applications accessed by means of a request-response programming model.

50.Q: Describe the lifecycle of a Servlet.
A: The lifecycle includes Initialization (init method), Service (service method), and Destruction (destroy method).

51.Q: How do you configure a Servlet in a web application?
A: You can configure a Servlet in the web.xml file:

```
<servlet>
    <servlet-name>exampleServlet</servlet-name>
    <servlet-class>com.example.ExampleServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>exampleServlet</servlet-name>
    <url-pattern>/example</url-pattern>
</servlet-mapping>
```

52.Q: What is the HttpServlet class?

A: HttpServlet is a class that extends GenericServlet and provides methods, such as doGet, doPost, doPut, and doDelete, to handle HTTP-specific services.

53. Q: How do you handle GET requests in a Servlet?
A: protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
// Handle GET request
}

54. Q: How do you handle POST requests in a Servlet?
A: protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
// Handle POST request
}

55. Q: What is the ServletConfig interface?
A: ServletConfig is an interface that provides servlet configuration information to the servlet at runtime.

56. Q: What is the ServletContext interface?
A: ServletContext is an interface that provides a servlet with information about its environment.

57. Q: How do you forward a request from one Servlet to another?
A: RequestDispatcher dispatcher = request.getRequestDispatcher("anotherServlet");
dispatcher.forward(request, response);

58. Q: How do you redirect a request in a Servlet?
A: response.sendRedirect("anotherServlet");

JAVA BEANS

59. Q: What is a JavaBean?
A: A JavaBean is a reusable software component that follows certain conventions, including having a no-argument constructor, being serializable, and providing getter and setter methods.

60. Q: What are the key features of a JavaBean?
A: Key features include properties, events, methods, and persistence.

61. Q: How do you create a simple JavaBean?
A: public class MyBean implements Serializable {

```
    private String property;

    public MyBean() {}

    public String getProperty() {
        return property;
    }

    public void setProperty(String property) {
        this.property = property;
    }
}
```

62. Q: What is the significance of the no-argument constructor in a JavaBean?
A: The no-argument constructor allows the bean to be instantiated easily and is required for the bean to be managed by various frameworks and tools.

63. Q: What are getter and setter methods in a JavaBean?
A: Getter methods retrieve the value of a property, and setter methods set or update the value of a property.

64. Q: Explain the concept of bound properties in JavaBeans.
A: Bound properties are properties that notify listeners when their values change.

65. Q: What are constrained properties in JavaBeans?
A: Constrained properties are properties that notify listeners when their values change and allow the change to be vetoed.

66. Q: How do you implement event handling in JavaBeans?
A: Event handling is implemented using listener interfaces and methods for adding and removing listeners.

67.Q: What is the PropertyChangeListener interface?

A: PropertyChangeListener is an interface for receiving property change events.

68.Q: How do you use JavaBeans in JSP?

A: You can use JavaBeans in JSP using the (JavaServer Pages)

```
<jsp:useBean id="myBean" class="com.example.MyBean"
scope="session" />
<jsp:setProperty name="myBean" property="property" value="value" />
<jsp:getProperty name="myBean" property="property" />
```