Core Java Concepts

Core Java Concepts to Learn

1. Java Data Types

- Primitive Data Types: int, float, double, char, boolean, etc.
- Reference Data Types: Objects, Arrays, etc.
- Wrapper Classes: Integer, Double, Character, Boolean, etc., which wrap primitive types in an object form.

2. Control Flow Statements

- Conditionals: if, else if, else, switch.
- Loops: for, while, do-while.
- Jump Statements: break, continue, return, throw, throws, and finally.

3. Arrays

- Understanding single-dimensional and multi-dimensional arrays.
- Operations like sorting, searching, and manipulating array elements.

4. Exception Handling

- Try, Catch, and Finally Blocks: Handling exceptions and ensuring cleanup of resources using finally.
- Checked vs Unchecked Exceptions: IOException, SQLException (checked), NullPointerException, ArithmeticException (unchecked).
- Custom Exceptions: How to create your own exceptions by extending Exception class.

5. String Handling

- String manipulation using methods like substring(), split(), toLowerCase(), toUpperCase(), equals(), etc.
- StringBuilder and StringBuffer for mutable string handling.
- String comparison (== vs .equals()), string concatenation.

6. Collections Framework

- List: ArrayList, LinkedList.
- Set: HashSet, LinkedHashSet, TreeSet.
- Map: HashMap, TreeMap, LinkedHashMap, HashTable.
- Queue: PriorityQueue, LinkedList.
- Iterator: Using the Iterator interface to traverse collections.
- Generics: Using type parameters to enforce type safety.

7. Java Threads and Concurrency

- Thread Class and Runnable Interface: Creating threads, start(), run().
- Synchronization: Using synchronized keyword to control access to shared resources.
- Thread Communication: wait(), notify(), notifyAll().
- ExecutorService: Thread pool management for handling concurrent tasks.
- Deadlock: Understanding and avoiding deadlocks in multi-threaded applications.

8. File I/O

- Reading from and writing to files using FileReader, BufferedReader, FileWriter, BufferedWriter.
- Serialization and Deserialization: Converting objects to streams and vice versa.
- Using NIO (New I/O) for file operations, Paths, Files, etc.

9. Java 8 Features (Lambda Expressions and Streams)

- Lambda Expressions: Syntax, functional interfaces, and use in method arguments.
- Functional Interfaces: Runnable, Callable, Comparator, etc.
- Streams API: Filtering, mapping, reducing, and collecting data using streams.

10. Inner Classes

- Nested Classes: Static and non-static inner classes.
- Local Inner Classes: Classes defined inside methods.
- Anonymous Classes: Useful when you need to instantiate a class for a one-time use.
- Lambda expressions are a more modern alternative to anonymous classes in many cases.

11. Java Memory Management and Garbage Collection

- Heap and Stack memory areas.
- Garbage Collection: Understanding how garbage collection works and different garbage collection algorithms.
- Memory Leaks: How to avoid memory leaks in your applications.

12. Java Class Loading Mechanism

- ClassLoader: How classes are loaded into memory.
- JVM Internals: Understanding how Java loads classes and handles memory management during execution.
- ClassPath: How Java finds classes at runtime.

13. JVM (Java Virtual Machine) and JRE (Java Runtime Environment)

- JVM Architecture: The internals of the JVM, how it manages memory, stack, heap, and garbage collection.

- JRE vs JDK: Understanding the difference between JDK (Java Development Kit) and JRE (Java Runtime Environment).

14. Annotations

- Using built-in annotations like @Override, @Deprecated, @SuppressWarnings.
- Custom Annotations: How to define and use your own annotations.

15. Access Modifiers and Scopes

- Access Modifiers: private, default, protected, and public access levels.
- Package and Import: Understanding how packages work and how classes are imported.
- Visibility and Inheritance: What members are accessible in subclass inheritance.

16. Java Reflection

- Using reflection to inspect classes, methods, fields, and annotations at runtime.
- Introspection: Creating and manipulating objects dynamically at runtime.

17. Design Patterns

- Creational Patterns: Singleton, Factory, Builder, Abstract Factory.
- Structural Patterns: Adapter, Composite, Proxy, Decorator.
- Behavioral Patterns: Strategy, Observer, Command, Template Method.

18. Java Networking

- Sockets: Using Socket and ServerSocket to build network-based applications.
- URL Connection: Making HTTP requests with HttpURLConnection.
- TCP/IP Communication: Understanding client-server communication models using Java.

19. Java Database Connectivity (JDBC)

- Connecting to databases using JDBC.
- PreparedStatement vs Statement for SQL query execution.
- Transaction Management in JDBC.
- ResultSet to process query results.

20. Java 9+ Features (Module System, JShell, etc.)

- Java Modules: Modular programming in Java introduced in Java 9.
- JShell: Interactive Java shell for quick prototyping and testing code.