***Technical Concepts Handbook***

***- Siddhant Patni***

***Mobile: +91-7588811796***

***Email: siddhant4patni@gmail.com***

Contents

[***Java*** 8](#_Toc195281408)

[ **Explain Java Program execution?** 8](#_Toc195281409)

[ **Why is java more secure?** 9](#_Toc195281410)

[ **Explain public static void main (String[] args) method?** 9](#_Toc195281411)

[ **What is the marker interface?** 10](#_Toc195281412)

[ **What is Serialization??** 10](#_Toc195281413)

[ **What is the final class?** 10](#_Toc195281414)

[ **How to create custom exceptions?** 10](#_Toc195281415)

[ **Why is string immutable in java?** 10](#_Toc195281416)

[ **What is the final keyword in java?** 10](#_Toc195281417)

[ **Where can we use the final keyword?** 10](#_Toc195281418)

[ **What will happen if I create the final class?** 10](#_Toc195281419)

[ **What is the difference between comparable and comparator?** 10](#_Toc195281420)

[ **Explain OOPs concept?** 10](#_Toc195281421)

[ **What is Encapsulation? Explain with a real time example** 10](#_Toc195281422)

[ **How do we achieve encapsulation in Java?** 11](#_Toc195281423)

[ **What is abstract in java?** 12](#_Toc195281424)

[ **What is static in java?** 13](#_Toc195281425)

[ **What is the difference between HashMap and HashSet** 14](#_Toc195281426)

[ **Explain Java8 features?** 14](#_Toc195281427)

[ **What is Functional Interface?** 14](#_Toc195281428)

[ **What is the default method in functional interface?** 14](#_Toc195281429)

[ **What is the map function in stream?** 14](#_Toc195281430)

[ **What is the filter in the stream?** 14](#_Toc195281431)

[ **What are the methods of thread?** 14](#_Toc195281432)

[ **What is thread pool? What are the types of thread pool?** 15](#_Toc195281433)

[ **What are thread pool methods?** 15](#_Toc195281434)

[ **What is method overriding?** 15](#_Toc195281435)

[ **What is method overloading?** 15](#_Toc195281436)

[ **What is Polymorphism? Explain with an example?** 15](#_Toc195281437)

[ **What is Inheritance?** 15](#_Toc195281438)

[ **What is the interface in java?** 15](#_Toc195281439)

[ **Can we have a static method inside the interface?** 15](#_Toc195281440)

[ **Can we override the static method?** 15](#_Toc195281441)

[ **We have Class A and Class B which contains m1() method in both the class, and class B extends parent class A. Asked which method will get called on which object?** 15](#_Toc195281442)

[ **What is the difference between ArrayList and LinkedList?** 15](#_Toc195281443)

[ **Which is more efficient among the ArrayList and LinkedList?** 15](#_Toc195281444)

[ **Which among ArrayList and LinkedList will be more efficient for random access of data?** 15](#_Toc195281445)

[ **How to sort the ArrayList?** 15](#_Toc195281446)

[ **Sort based on multiple key and value pairs?** 15](#_Toc195281447)

[ **What is HashMap and explain its internal working?** 16](#_Toc195281448)

[ **Can we store null value in HashMap?** 16](#_Toc195281449)

[ **Can we store infinite data in a HashMap?** 16](#_Toc195281450)

[ **Can we access null value in HashMap?** 16](#_Toc195281451)

[ **What is exception handling?** 16](#_Toc195281452)

[ **Explain the hierarchy of Exception?** 16](#_Toc195281453)

[ **What is the difference between checked exception and unchecked exception?** 16](#_Toc195281454)

[ **How to handle unchecked exceptions?** 16](#_Toc195281455)

[ **Explain Generics? What is the use of Generics?** 16](#_Toc195281456)

[ **What is composition and aggregation?** 16](#_Toc195281457)

[ **Parent and child class methods and how to access those?** 16](#_Toc195281458)

[ **What is the use map method in java 8 stream?** 16](#_Toc195281459)

[ **Which collection you will use in order to remove duplicate elements and preserve the insertion order?** 16](#_Toc195281460)

[ **How to find the second largest element from the list?** 16](#_Toc195281461)

[ **What is the difference between Map and FlatMap in stream?** 16](#_Toc195281462)

[ **Where can you use FlatMap in a stream?** 16](#_Toc195281463)

[ **What is multithreading in java?** 17](#_Toc195281464)

[ **What are all methods present in Thread class?** 17](#_Toc195281465)

[ **How do notify methods work in multithreading?** 17](#_Toc195281466)

[ **What is synchronized in multithreading?** 17](#_Toc195281467)

[ **What is Singleton in java?** 17](#_Toc195281468)

[ **Explain Java 8 Functional interface?** 17](#_Toc195281469)

[ **What is the lambda expression?** 17](#_Toc195281470)

[ **What is the prerequisite for lambda expression?** 17](#_Toc195281471)

[ **Can we have more than one abstract method in a functional interface?** 17](#_Toc195281472)

[ **What are all functional interfaces present in java?** 17](#_Toc195281473)

[ **What is JVM?** 17](#_Toc195281474)

[ **What are the different loaders in JVM?** 17](#_Toc195281475)

[ **What is garbage collection in java?** 17](#_Toc195281476)

[ **What will happen if I assign null to some object then will that be garbage collected?** 17](#_Toc195281477)

[ **When does the finalize method get called?** 17](#_Toc195281478)

[ **How to handle IOException?** 17](#_Toc195281479)

[ **question?** 17](#_Toc195281480)

[ **question?** 18](#_Toc195281481)

[ **question?** 18](#_Toc195281482)

[ **question?** 18](#_Toc195281483)

[ **question?** 18](#_Toc195281484)

[ **question?** 18](#_Toc195281485)

[ **question?** 18](#_Toc195281486)

[ **question?** 18](#_Toc195281487)

[ **question?** 18](#_Toc195281488)

[ **question?** 18](#_Toc195281489)

[ **question?** 18](#_Toc195281490)

[ **question?** 18](#_Toc195281491)

[ **question?** 18](#_Toc195281492)

[ **question?** 18](#_Toc195281493)

[ **question?** 18](#_Toc195281494)

[ **question?** 18](#_Toc195281495)

[***Java Programs*** 19](#_Toc195281496)

[ **Program – 1** 19](#_Toc195281497)

[ **Program – 2** 19](#_Toc195281498)

[ **Program – 3** 19](#_Toc195281499)

[ **Program – 4** 19](#_Toc195281500)

[ **Program – 5** 19](#_Toc195281501)

[ **Program – 6** 19](#_Toc195281502)

[ **Program – 7** 19](#_Toc195281503)

[ **Program – 8** 20](#_Toc195281504)

[ **Program – 9** 20](#_Toc195281505)

[ **Program – 10** 20](#_Toc195281506)

[ **Program – 11** 20](#_Toc195281507)

[ **Program – 12** 20](#_Toc195281508)

[ **Program – 13** 20](#_Toc195281509)

[ **Program – 14** 20](#_Toc195281510)

[Advance Java / J2EE 21](#_Toc195281511)

[ **What is servlet?** 21](#_Toc195281512)

[ **What is JSP?** 21](#_Toc195281513)

[ **What is the difference between servlet and JSP?** 21](#_Toc195281514)

[ **What is the JSP lifecycle?** 21](#_Toc195281515)

[ **What is Spring MVC? Explain architecture of spring MVC?** 21](#_Toc195281516)

[ **What is the singleton design pattern?** 21](#_Toc195281517)

[ **question?** 21](#_Toc195281518)

[ **question?** 21](#_Toc195281519)

[ **question?** 21](#_Toc195281520)

[ **question?** 21](#_Toc195281521)

[ **question?** 21](#_Toc195281522)

[ **question?** 21](#_Toc195281523)

[ **question?** 21](#_Toc195281524)

[Design Patterns 22](#_Toc195281525)

[ **What is the singleton design pattern?** 22](#_Toc195281526)

[ **question?** 22](#_Toc195281527)

[ **question?** 22](#_Toc195281528)

[Spring 23](#_Toc195281529)

[ **What are the spring bean scopes?** 23](#_Toc195281530)

[ **Explain any scope of spring bean?** 23](#_Toc195281531)

[ **How to add scope in bean configuration?** 23](#_Toc195281532)

[ **What is the Spring batch? Explain where you have used the spring batch in your project?** 23](#_Toc195281533)

[ **How spring batch works?** 23](#_Toc195281534)

[ **How to implement the spring batch?** 23](#_Toc195281535)

[ **What is the difference between SOAP and REST? Which is more efficient?** 23](#_Toc195281536)

[ **How REST API works?** 23](#_Toc195281537)

[ **Difference between Spring and Spring Boot?** 23](#_Toc195281538)

[Hibernate 24](#_Toc195281539)

[ How data is persisted in database by using hibernate**?** 24](#_Toc195281540)

[ **How to retrieve select query using Hibernate?** 26](#_Toc195281541)

[ **How to retrieve data through hibernate?** 26](#_Toc195281542)

[Spring Boot 27](#_Toc195281543)

[ **Difference between Spring and Spring Boot?** 27](#_Toc195281544)

[ **What are the advantages of SprigBoot?** 27](#_Toc195281545)

[ **What is a yaml file in SpringBoot?** 27](#_Toc195281546)

[ **What is what is @transient in spring boot?** 27](#_Toc195281547)

[ **What are the types of propagation?** 27](#_Toc195281548)

[ **What @Qualifier in spring? What is the use of @Qualifier?** 27](#_Toc195281549)

[ **question?** 27](#_Toc195281550)

[ **question?** 27](#_Toc195281551)

[ **question?** 27](#_Toc195281552)

[ **question?** 27](#_Toc195281553)

[ **question?** 27](#_Toc195281554)

[ **question?** 27](#_Toc195281555)

[Microservices 28](#_Toc195281556)

[ **Monolith vs Microservices differences and when to choose which?** 28](#_Toc195281557)

[ **How to design a microservice from scratch?** 28](#_Toc195281558)

[ **API Gateway pattern and its advantages?** 28](#_Toc195281559)

[ **Inter-service communication: REST vs Messaging?** 28](#_Toc195281560)

[ **Circuit Breaker pattern and its implementation using Resilience4j?** 28](#_Toc195281561)

[ **Load balancing in microservices using Spring Cloud Load Balancer?** 28](#_Toc195281562)

[ **How Spring Cloud Config helps in centralized configuration management?** 28](#_Toc195281563)

[ **Service discovery using Eureka or Consul?** 28](#_Toc195281564)

[ **Feign Client vs WebClient: Which one to use and why?** 28](#_Toc195281565)

[ **Event-driven architecture and Kafka integration?** 28](#_Toc195281566)

[ **Database per service vs Shared Database: Pros and cons?** 28](#_Toc195281567)

[ **Saga Pattern for distributed transactions in microservices?** 28](#_Toc195281568)

[ **JWT-based authentication and OAuth2 in microservices?** 28](#_Toc195281569)

[ **How to handle security in an API Gateway?** 28](#_Toc195281570)

[ **Observability: Logging, tracing, and monitoring best practices?** 28](#_Toc195281571)

[ **Role of Prometheus and Grafana in microservices monitoring?** 29](#_Toc195281572)

[ **Kubernetes deployment strategies for microservices?** 29](#_Toc195281573)

[ **Blue-Green and Canary deployments in microservices?** 29](#_Toc195281574)

[ **When to use WebFlux for reactive microservices?** 29](#_Toc195281575)

[ **CQRS and Event Sourcing: When and why to use them?** 29](#_Toc195281576)

[SQL 30](#_Toc195281577)

[ **What is ordered by and range in the database?** 30](#_Toc195281578)

[ **Write a SQL query to get the second highest salary of an employee from the employee table?** 30](#_Toc195281579)

[ **Write a SQL query to print employee id, employee\_name, depatment\_name Employee - employee\_id, employee\_name, department\_id** 30](#_Toc195281580)

[**Department - department\_id, department\_name?** 30](#_Toc195281581)

[ **Question?** 30](#_Toc195281582)

[Docker 31](#_Toc195281583)

[ **question?** 31](#_Toc195281584)

[ **question?** 31](#_Toc195281585)

[ **question?** 31](#_Toc195281586)

[ **question?** 31](#_Toc195281587)

[ **question?** 31](#_Toc195281588)

[ **question?** 31](#_Toc195281589)

[Kubernetes 32](#_Toc195281590)

[ **question?** 32](#_Toc195281591)

[ **question?** 32](#_Toc195281592)

[ **question?** 32](#_Toc195281593)

[ **question?** 32](#_Toc195281594)

[ **question?** 32](#_Toc195281595)

[ **question?** 32](#_Toc195281596)

[AWS 33](#_Toc195281597)

[ **question?** 33](#_Toc195281598)

[ **question?** 33](#_Toc195281599)

[ **question?** 33](#_Toc195281600)

[ **question?** 33](#_Toc195281601)

[ **question?** 33](#_Toc195281602)

[ **question?** 33](#_Toc195281603)

[Azure 34](#_Toc195281604)

[ **question?** 34](#_Toc195281605)

[ **question?** 34](#_Toc195281606)

[ **question?** 34](#_Toc195281607)

[ **question?** 34](#_Toc195281608)

[ **question?** 34](#_Toc195281609)

[ **question?** 34](#_Toc195281610)

[Angular 35](#_Toc195281611)

[ **What are the components in the Angular?** 35](#_Toc195281612)

[ **How to pass data from components in Angular?** 35](#_Toc195281613)

[ **question?** 35](#_Toc195281614)

[ **question?** 35](#_Toc195281615)

[ **question?** 35](#_Toc195281616)

[ **question?** 35](#_Toc195281617)

[React 36](#_Toc195281618)

[ **question?** 36](#_Toc195281619)

[ **question?** 36](#_Toc195281620)

[ **question?** 36](#_Toc195281621)

[ **question?** 36](#_Toc195281622)

[ **question?** 36](#_Toc195281623)

[ **question?** 36](#_Toc195281624)

# ***Java***

### **Explain Java Program execution?**

* **1. Writing the Source Code**
  + You create a .java file containing your Java code (e.g., HelloWorld.java)
  + Example:

public class HelloWorld {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

**2. Compilation (javac)**

* + The Java compiler (javac) converts your source code into **bytecode** (platform-independent intermediate code)
  + Creates .class files (e.g., HelloWorld.class)
  + Bytecode is not machine code - it's instructions for the **Java Virtual Machine (JVM)**

**3. Class Loading**

When you run java HelloWorld:

**a) Bootstrap ClassLoader**

* Loads core Java classes (from rt.jar and other core libraries)
* Written in native code (not Java)

**b) Extension ClassLoader**

* Loads classes from Java extension directories

**c) Application ClassLoader**

* Loads your application classes (from the classpath)

**4. Bytecode Verification**

* JVM verifies the bytecode to ensure:
  + No illegal memory access
  + Proper stack manipulation
  + Correct method calls
  + No violation of access restrictions
* This provides security by preventing malicious code

**5. Just-In-Time (JIT) Compilation**

* The JVM's JIT compiler converts frequently executed bytecode into **native machine code**
* Happens at runtime
* Optimizes performance by:
  + Inlining methods
  + Removing dead code
  + Optimizing loops

**6. Execution**

* The JVM executes the program:
  + Creates the main thread
  + Allocates memory for objects in the Heap
  + Manages method calls using the Stack
  + Handles garbage collection automatically

**7. Runtime Memory Areas**

The JVM manages these memory areas during execution:

a) Method Area

* Stores class structures, method code, and static variables

b) Heap

* Stores all objects and their instance variables
* Garbage collection works here

c) JVM Stacks

* Each thread has its own stack
* Stores frames for each method call (local variables, operands, return values)

d) PC Registers

* Tracks the execution position for each thread

e) Native Method Stacks

* For native code (non-Java code)

**8. Garbage Collection**

* Automatically reclaims memory from objects no longer in use
* Runs in the background
* Different algorithms available (Serial, Parallel, G1, ZGC, etc.)



### **What is Classloader? Different types of it?**

* Answer

### **Why is java more secure?**

* Answer

### **Explain public static void main (String[] args) method?**

* Answer

### **What is the marker interface?**

* Answer

### **What is serialization and deserialization?**

* Answer

### **What is the use of the transient keyword in serialization?**

* Answer

### **What is the final class?**

* Answer

### **How to create custom exceptions?**

* Answer

### **Why is string immutable in java?**

* Answer

### **What is the final keyword in java?**

* Answer

### **Where can we use the final keyword?**

* Answer

### **What will happen if I create the final class?**

* Answer

### **What is the difference between comparable and comparator?**

* Answer

### **Explain OOPs concept?**

* Object-Oriented Programming (OOPs) is a programming paradigm based on the concept of "objects", which can contain data and code. Java is a fully object-oriented language (except for primitive types) and supports the following core OOPs concepts:
* **Abstraction**
  + Hides internal implementation and shows only functionality.
  + Achieved using **abstract classes** and **interfaces**.
* **Encapsulation**
  + Bundling of data (variables) and methods into a single unit (class).
  + Achieved using **access modifiers** (private, public, protected).
* **Inheritance**
  + Allows a class (child) to inherit fields and methods from another class (parent).
  + Promotes code reusability and method overriding.
* **Polymorphism**
  + One entity behaves differently based on context.
  + **Compile-time (method overloading)** and **Run-time (method overriding)** are the two types.

### **What is an object?**

* answer

### **Explain method in Object class? Explain 9 methods in Object class?**

* answer

### **What is Encapsulation? Explain with a real time example**

* Encapsulation is one of the fundamental OOPs principles in Java. It refers to wrapping data (variables) and the code (methods) that operates on the data into a single unit — typically a class. It helps protect the internal state of an object from unwanted external modifications.
* Encapsulation is achieved using:
  + Private fields to restrict direct access.
  + Public getters and setters to allow controlled access.
* Example: Bank Account

public class BankAccount {

private double balance; // private field — can't be accessed directly

public BankAccount(double initialBalance) {

this.balance = initialBalance;

}

// public method to get balance

public double getBalance() {

return balance;

}

// public method to deposit money (with validation)

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

}

}

// public method to withdraw money (with validation)

public void withdraw(double amount) {

if (amount > 0 && balance >= amount) {

balance -= amount;

}

}

}

**Explanation:**

* The balance variable is **encapsulated** — it's private.
* Access is provided **only through public methods** (getBalance, deposit, withdraw), ensuring that:
  + The balance can't be set to a negative value directly.
  + Deposits and withdrawals go through validation.

This prevents incorrect or unauthorized changes to the data, which is the essence of **encapsulation** in real-world applications.

### **How do we achieve encapsulation in Java?**

* Encapsulation is one of the four fundamental OOP concepts in Java. It refers to bundling data (variables) and methods that operate on that data into a single unit (class) while restricting direct access to some of the object's components.

**Key Ways to Achieve Encapsulation:**

**1. Using Private Access Modifier**

Make class fields private to prevent direct access from outside the class.

public class Person {

private String name; // private field

private int age;

}

**2. Providing Public Getter and Setter Methods**

Create public methods to access and modify private fields.

public class Person {

private String name;

private int age;

// Getter for name

public String getName() {

return name;

}

// Setter for name

public void setName(String name) {

this.name = name;

}

// Getter for age

public int getAge() {

return age;

}

// Setter for age with validation

public void setAge(int age) {

if(age > 0) { // validation logic

this.age = age;

}

}

}

**3. Implementing Constructors with Validation**

Initialize fields through constructors with validation.

public Person(String name, int age) {

this.name = name;

if(age > 0) {

this.age = age;

}

}

**Benefits of Encapsulation:**

* **Data Hiding**: Internal representation is hidden from outside
* **Increased Flexibility**: Can change internal implementation without affecting other code
* **Reusability**: Encapsulated code is easier to reuse
* **Control**: Can add validation logic in setters
* **Maintainability**: Easier to maintain and modify code

**Example Usage:**

public class Main {

public static void main(String[] args) {

Person person = new Person("Alice", 30);

// Access through getters

System.out.println(person.getName()); // Alice

System.out.println(person.getAge()); // 30

// Modify through setters

person.setAge(31);

person.setName("Alice Smith");

// Invalid age won't be set

person.setAge(-5); // age remains 31

}

}

### **What is abstract in java?**

* **Abstract** is a non-access modifier that can be applied to classes and methods.

**Abstract Class:** Abstract is a non-access modifier that can be applied to classes and methods.

* Cannot be instantiated (cannot create objects)
* Can contain both abstract and concrete methods
* Used as a base class for inheritance

abstract class Animal {

// Abstract method (no implementation)

abstract void makeSound();

// Concrete method

void eat() {

System.out.println("Animal is eating");

}

}

**Abstract Method:**

* Has no body (no implementation)
* Must be overridden by the first concrete subclass
* Can only exist in abstract classes

abstract class Shape {

abstract double calculateArea(); // abstract method

}

**Key Points:**

* If a class has even one abstract method, the class must be declared abstract
* Abstract classes can have constructors (called when subclass is instantiated)
* Used to define common interface for subclasses

### **What is static in java?**

* **Static** is a modifier that can be applied to variables, methods, blocks, and nested classes.

**Static Variable (Class Variable):**

* Belongs to the class rather than any object
* Shared by all instances of the class
* Initialized when class is loaded

class Counter {

static int count = 0; // static variable

Counter() {

count++;

}

}

**Static Method:**

* Belongs to the class rather than instances
* Can be called without creating an object
* Can only access static members directly

class MathUtils {

static int add(int a, int b) { // static method

return a + b;

}

}

// Usage: MathUtils.add(5, 3);

**Static Block:**

* Used for static initialization of a class
* Executed when the class is loaded

class MyClass {

static {

System.out.println("Static block executed");

}

}

**Example Combining Both:**

abstract class Database {

static final String DEFAULT\_URL = "jdbc:default"; // static constant

abstract void connect(); // abstract method

static void printDefaultUrl() { // static method

System.out.println(DEFAULT\_URL);

}

}



**Note:**

Static members are resolved at compile-time (early binding), while abstract methods enable runtime polymorphism (late binding).

### **What is the difference between HashMap and HashSet**

* Answer

### **Explain Java8 features?**

* answer

### **What is Functional Interface?**

* answer

### **What is the default method in functional interface?**

* answer

### **What is the map function in stream?**

* answer

### **What is the filter in the stream?**

* answer

### **What are the methods of thread?**

* Answer

### **What is thread pool? What are the types of thread pool?**

* answer

### **What are thread pool methods?**

* Answer

### **What is method overriding?**

* Answer

### **How to achieve method overriding?**

* Answer

### **What is method overloading?**

* answer

### **What is Polymorphism? Explain with an example?**

* answer

### **What is Inheritance?**

* Answer

### **What is the interface in java?**

* Answer

### **Can we have a static method inside the interface?**

* Answer

### **Can we override the static method?**

* answer

### **We have Class A and Class B which contains m1() method in both the class, and class B extends parent class A. Asked which method will get called on which object?**

* answer

### **What is the difference between ArrayList and LinkedList?**

* Answer

### **Which is more efficient among the ArrayList and LinkedList?**

* answer

### **Which among ArrayList and LinkedList will be more efficient for random access of data?**

* answer

### **How to sort the ArrayList?**

* answer

### **Sort based on multiple key and value pairs?**

* answer

### **What is HashMap and explain its internal working?**

* Answer

### **What are the types of HashMaps?**

* answer

### **Can we store null value in HashMap?**

* answer

### **Can we store infinite data in a HashMap?**

* answer

### **Can we access null value in HashMap?**

* answer

### **What is exception handling?**

* answer

### **Explain the hierarchy of Exception?**

* answer

### **What is the difference between checked exception and unchecked exception?**

* answer

### **How to handle unchecked exceptions?**

* answer

### **Explain Generics? What is the use of Generics?**

* answer

### **What is composition and aggregation?**

* answer

### **Parent and child class methods and how to access those?**

* answer

### **What is the use map method in java 8 stream?**

* answer

### **Which collection you will use in order to remove duplicate elements and preserve the insertion order?**

* answer

### **How to find the second largest element from the list?**

* answer

### **What is the difference between Map and FlatMap in stream?**

* answer

### **Where can you use FlatMap in a stream?**

* answer

### **What is multithreading in java?**

* answer

### **What are all methods present in Thread class?**

* answer

### **How do notify methods work in multithreading?**

* answer

### **What is synchronized in multithreading?**

* answer

### **What is Singleton in java?**

* answer

### **Explain Java 8 Functional interface?**

* answer

### **What is the lambda expression?**

* answer

### **What is the prerequisite for lambda expression?**

* answer

### **Can we have more than one abstract method in a functional interface?**

* answer

### **What are all functional interfaces present in java?**

* answer

### **What is JVM?**

* answer

### **What are the different loaders in JVM?**

* answer

### **What is garbage collection in java?**

* answer

### **What will happen if I assign null to some object then will that be garbage collected?**

* answer

### **When does the finalize method get called?**

* answer

### **How to handle IOException?**

* answer

### **What is JDBC?**

* answer

### **Explain Hashset?**

* answer

### **What is the difference between list and set?**

* answer

### **Consider we have Strings as String s1 = “Welcome”, String s2 = “Welcome” and String s3 = “WelcomeOne”. What does s1==s2 represent?**

* answer

### **Explain String constant pool?**

* answer

### **What is the difference between equals() and hashCode() method?**

* answer

### **What is the default size of ArrayList?**

* answer

### **How to increase the size of ArrayList?**

* answer

### **Can we add infinite elements to ArrayList?**

* answer

### **Can we store infinite elements in a list?**

* answer

### **Does Java support Pointers?**

* answer

### **Why doesn't java support pointers?**

* answer

### **Explain which language you prefer in terms of security among C, C++, Java and Python?**

* answer

### **Is Java 100% object oriented?**

* answer

### **How can we make java 100% object oriented?**

* answer

### **Can we create objects of static class?**

* Answer

### **What is the difference between ArrayList and LinkedList? Which is better?**

* answer

### **What is Enumeration in java?**

* answer

### **Explain Thread life cycle?**

* answer

### **What is Garbage Collection? Explain?**

* answer

### **Explain JVM, JDK and JRE?**

* answer

### **Which compiler is used by Java?**

* answer

### **What is Functional Interface in java?**

* answer

### **Where are the hashcode and equals methods defined in java?**

* answer

### **What is a deadlock? Explain with an example?**

* answer

### **How many design patterns are present in java?**

* answer

### **Can we override the static method?**

* answer

### **Explain access modifiers in java?**

* answer

### **What is volatile in java? Where can we use it?**

* answer

### **What is an idempotent method in java?**

* answer

### **How to create threads in java?**

* answer

### **How to start a thread in java?**

* answer

### **Explain heap memory in java?**

* answer

### **When does the garbage collection get called?**

* answer

### **What is final, static and non-static (instance) in java?**

* answer

### **How to create custom exceptions?**

* answer

### **What will happen if I use Exception and custom exceptions together? Priority?**

* answer

### **How will I use two threads using a singleton design pattern?**

* answer

### **How to implement thread safety in java using singleton?**

* answer

### **Write an immutable class in Java?**

* answer

### **Can we override the protected method?**

* answer

### **How to create threads in two different ways?**

* answer

### **Thread safety with Singleton class?**

* answer

### **What is synchronized in java?**

* answer

### **Compare ArrayList objects from the employee and return the highest employee age?**

* answer

### **How does ArrayList internally work?**

* answer

### **What is the default size of ArrayList?**

* answer

### **How to increase the size of ArrayList? By how much?**

* answer

### **Which data structure is used by ArrayList?**

* answer

### **What is predicate and consumer in java 8?**

* answer

### **What is marker interface in java? What is the purpose of the marker interface?**

* answer

### **Explain Thread executors?**

* answer

### **What is the use of finally block in exception handling?**

* answer

### **How can we ignore the finally block?**

* answer

### **What is a try with resources?**

* answer

### **Best time complexity of Arraylist and Linkedlist? Which is better?**

* answer

### **What is the use of a concurrent hashmap?**

* answer

### **What is synchronous and asynchronous call?**

* answer

### **Difference between HashMap and LinkedHashMap?**

* answer

### **What is the difference between Fail Fast and Fail Safe iterator?**

* answer

### **Can we write the main method as private?**

* answer

### **What is the use of static keywords in java?**

* answer

### **What is inner class? Have you used inner class in your project?**

* answer

### **What is the purpose of inner class?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# ***Java Programs***

### **Program – 1**

Write a program in java to return max sum of contiguous subarray of size 3?

(Write algorithm steps as well)

Example. [2,1,5,1,3,2], where k=3 (max size of sub array)?

* answer

### **Program – 2**

Print even and odd numbers using thread

* Answer

### **Program – 3**

Write a method to return the maximum value from an integer array passed as an input parameter

public class FindLargestNumber

{

public static int returnLargetNumber(int arr[])

{

int i;

int largestNumber = arr[0];

for (i = 1; i < arr.length; i++)

if (arr[i] > largestNumber) {

largestNumber = arr[i];

return largestNumber;

}

public static void main(String[] args)

{

int arr[] = {55, 12, 0, 786, 98};

System.out.println("Largest number in given array : " + returnLargetNumber(arr));

}

}

### **Program – 4**

Problem Definition: Write a program to implement Singleton class?

* Answer

### **Program – 5**

Problem Definition: Write a program - method to return missing number from array of length n=3

Example: n=3, [2,0,3] -> Output -> 1

* Answer

### **Program – 6**

Problem Definition: Write a program in java to check if two strings are Anagram or not. Return boolean method

LISTEN - SILENT

Implement using collections.

import java.util.Arrays;

import java.util.HashMap;

public class Anagram {

public static boolean checkAnagramUsingArray(String str1, String str2) {

str1 = str1.replaceAll("\\s", "");

str2 = str2.replaceAll("\\s", "");

if (str1.length() != str2.length()) {

return false;

}

char[] str1Array = str1.toLowerCase().toCharArray();

char[] str2Array = str2.toLowerCase().toCharArray();

Arrays.sort(str1Array);

Arrays.sort(str2Array);

return Arrays.equals(str1Array, str2Array);

}

public static boolean checkAnagramUsingCollection(String str1, String str2) {

str1 = str1.replaceAll("\\s", "");

str2 = str2.replaceAll("\\s", "");

if (str1.length() != str2.length()) {

return false;

}

HashMap<Character, Integer> hm1 = new HashMap<Character, Integer>();

HashMap<Character, Integer> hm2 = new HashMap<Character, Integer>();

char[] str1Array = str1.toCharArray();

char[] str2Array = str2.toCharArray();

for (char value : str1Array) {

if (hm1.get(value) == null) {

hm1.put(value, 1);

} else {

int c = hm1.get(value);

hm1.put(value, ++c);

}

}

for (char c : str2Array) {

if (hm2.get(c) == null) {

hm2.put(c, 1);

} else {

int d = hm2.get(c);

hm2.put(c, ++d);

}

}

return hm1.equals(hm2);

}

public static void main(String[] args) {

String str1 = "LISTEN";

String str2 = "SILENT";

System.out.println(checkAnagramUsingArray(str1, str2));

System.out.println(checkAnagramUsingCollection(str1, str2));

}

}

### **Program – 7**

Problem Definition: Write a program to Swap two numbers without using a third variable

* Answer

### **Program – 8**

Problem Definition: Write a program to print even and odd numbers without using the modulus operator?

* Answer

### **Program – 9**

Problem Definition: Write a program to return the third largest element from an array using java 8 features?

* Answer

### **Program – 10**

Problem Definition: Write a program to print even and odd numbers using java 8 stream api?

* Answer

### **Program – 11**

Problem Definition: Write a program to print fibonacci series?

* Answer

### **Program – 12**

Problem Definition: Write a program to perform Reverse an array in groups of given size?

**Example:** [1,2,3,4,5,6,7] where k=3

**Output:** [5,6,7,1,2,3,4]

* Answer

### **Program – 13**

Problem Definition: Write a program to print factorials of natural numbers from 1 to 10

* Answer

### **Program – 14**

Problem Definition: Write a program to reverse the integer array without using loops? Using recursion?

* Answer

### **Program – 15**

Problem Definition: Write a program to perform Overriding methods in parent child

* Answer

### **Program – 16**

Problem Definition: Write a Java 8 code to return the student object if gender is male? (filter)

* Answer

### **Program – 17**

Problem Definition: Write a program using string, use only single for loop?

Input - String str = "Siddhant Patni";

Output - SiindtdahPata

* Answer

### **Program – 18**

Problem Definition: Write a program to print all the subsets of the given set with sum equal to given sum

set of numbers = {3, 35, 56, 2, 95, 10, 65, 150, 165, 23, 65, 18, 57}

sum = 28

* Answer

### **Program – 19**

Problem Definition: Write a program to count set bits in an integer

Example : 13 -> 1101 Print number of 1’s in the given binary number

* Answer

### **Program – 20**

Problem Definition: Write a program to balancing of the brackets

Example : Input - {[(a+b)+c]+x+y]}

* Answer

### **Program – 21**

Problem Definition:

* Answer

### **Program – 22**

Problem Definition:

* Answer

### **Program – 23**

Problem Definition:

* Answer

### **Program – 24**

Problem Definition:

* Answer

### **Program – 25**

Problem Definition:

* Answer

# Advance Java / J2EE

### **What is servlet?**

* Answer

### **Explain servlet lifecycle?**

* Answer

### **What is a dispatcher servlet?**

* answer

### **What is JSP?**

* answer

### **What is the difference between servlet and JSP?**

* answer

### **What is the JSP lifecycle?**

* answer

### **What is Spring MVC? Explain architecture of spring MVC?**

* Answer

### **What is the singleton design pattern?**

* answer

### **What is the REST API?**

* answer

### **Difference between API and REST?**

* answer

### **Difference between SOAP and REST?**

* answer

### **Which among SOAP and REST is easy?**

* answer

### **Explain REST architecture?**

* answer

### **question?**

* answer

### **question?**

* answer

# Data Structure

### **Find the middle element from the singly linked list without finding the length?**

* Answer

### **Complexity to find length of singly linked list?**

* Answer

# Design Patterns

### **What is the singleton design pattern? Explain it?**

* answer

### **Explain Factory design pattern?**

* answer

### **What is the Observer design pattern?**

* answer

# Spring

### **What are the spring bean scopes?**

* answer

### **Explain any scope of spring bean?**

* Answer

### **How to add scope in bean configuration?**

* Answer

### **What is the Spring batch? Explain where you have used the spring batch in your project?**

* answer

### **How spring batch works?**

* answer

### **How to implement the spring batch?**

* Answer

### **What is the difference between SOAP and REST? Which is more efficient?**

* Answer

### **How REST API works?**

* Answer

### **Difference between Spring and Spring Boot?**

* Answer

### **Difference between application context and beanfactory in spring?**

* Answer

### **Explain Spring MVC architecture?**

* Answer

### **Difference between Spring and Spring Boot?**

* Answer

### **What is the use of @Transactional annotation in spring?**

* Answer

# Hibernate

### How data is persisted in database by using hibernate**?**

* Hibernate is an ORM (Object-Relational Mapping) framework that simplifies database operations in Java applications. Here's how data gets persisted to a database using Hibernate:

1. Configuration Setup

First, configure Hibernate to connect to your database:

<!-- hibernate.cfg.xml -->

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

<property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/mydb</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">password</property>

<!-- SQL dialect -->

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<!-- Mapping files -->

<mapping class="com.example.Employee"/>

</session-factory>

</hibernate-configuration>

2. Entity Class Definition

Create a Java class annotated with @Entity:

@Entity

@Table(name = "employees")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "emp\_name")

private String name;

private double salary;

// Getters and setters

}

3. Persistence Process

Here's the step-by-step persistence flow:

a) Create SessionFactory

Configuration config = new Configuration().configure("hibernate.cfg.xml");

SessionFactory sessionFactory = config.buildSessionFactory();

b) Open Session

Session session = sessionFactory.openSession();

c) Begin Transaction

Transaction tx = session.beginTransaction();

d) Create Entity Object

Employee emp = new Employee();

emp.setName("John Doe");

emp.setSalary(50000);

e) Save/Persist the Object

session.save(emp); // or session.persist(emp);

f) Commit Transaction

tx.commit();

g) Close Session

session.close();

4. What Happens Internally

When you call **session.save():**

Hibernate checks the entity state - determines if it's transient (new), persistent (managed), or detached

Generates SQL - Hibernate creates the appropriate INSERT statement based on your entity mappings

Executes SQL - The statement is executed against the database

Updates entity state - The transient object becomes persistent

Handles ID generation - If using auto-increment, retrieves the generated ID

5. Transaction Management

Hibernate operations should be performed within transactions:

Session session = sessionFactory.openSession();

try {

Transaction tx = session.beginTransaction();

// Perform operations

Employee emp = new Employee("Jane Smith", 60000);

session.save(emp);

tx.commit(); // Changes are flushed to database here

} catch (Exception e) {

if (tx != null) tx.rollback();

throw e;

} finally {

session.close();

}

6. Different Persistence Operations

**Operation Description**

save() Persists the object, returns generated ID

persist() Similar to save() but doesn't guarantee immediate ID assignment

update() Updates a detached object

merge() Copies state of detached object to persistent object

saveOrUpdate() Either saves or updates based on object state

7. Hibernate Cache Flow

**First Level Cache (Session cache):**

* Exists per session
* All persistent objects are stored here
* Cleared when session closes

**Second Level Cache (Optional):**

* Shared across sessions
* Needs explicit configuration
* Reduces database hits for frequently accessed data

Example Complete Flow

public class HibernateExample {

public static void main(String[] args) {

// 1. Configure and build SessionFactory

SessionFactory sessionFactory = new Configuration()

.configure("hibernate.cfg.xml")

.buildSessionFactory();

// 2. Open session

Session session = sessionFactory.openSession();

Transaction tx = null;

try {

// 3. Begin transaction

tx = session.beginTransaction();

// 4. Create and persist object

Employee newEmployee = new Employee();

newEmployee.setName("Michael Johnson");

newEmployee.setSalary(75000);

// 5. Save to database

Long employeeId = (Long) session.save(newEmployee);

System.out.println("Employee saved with ID: " + employeeId);

// 6. Commit transaction

tx.commit();

} catch (Exception e) {

if (tx != null) tx.rollback();

e.printStackTrace();

} finally {

// 7. Close session

session.close();

sessionFactory.close();

}

}

}

**Key Benefits of Hibernate Persistence:**

* Object-Oriented Approach - Work with objects rather than SQL
* Automatic SQL Generation - No need to write CRUD queries
* Transaction Management - Built-in support for ACID properties
* Caching - Improves performance
* Database Independence - Switch databases with minimal code changes

This is the fundamental process of how Hibernate persists Java objects to relational databases while handling all the underlying JDBC complexity.

### **How to retrieve select query using Hibernate?**

* Answer

### **How to retrieve data through hibernate?**

* Answer

### **Hibernate caching?**

* Answer

### **Explain the hibernate sessions?**

* Answer

### **Difference between JDBC and Hibernate?**

* Answer

### **What is the difference between get and load method in hibernate?**

* Answer

### **Explain bean scopes?**

* Answer

### **Can we have beans without bean id?**

* Answer

# Spring Boot

### **Difference between Spring and Spring Boot?**

* Answer

### **What are the advantages of SprigBoot?**

* Answer

### **What is a yaml file in SpringBoot?**

* Answer

### **What is what is @transient in spring boot?**

* answer

### **What are the types of propagation?**

* Answer

### **What @Qualifier in spring? What is the use of @Qualifier?**

* answer

### **What are the advantages of Spring boot?**

* answer

### **How to create multiple modules using spring boot?**

* answer

### **What are all methods present in the controller?**

* answer

### **Explain annotations in Spring boot?**

* answer

### **What is the query parameter?**

* answer

### **How to change the server in Spring boot?**

* answer

### **What is the difference between @Controller and @RestController?**

* answer

### **What is the use of @Responsebody annotation?**

* answer

### **Explain @RestController annotation in Spring boot?**

* answer

### **Which two annotations are used in built by @RestController?**

* answer

### **How to access application.properties properties in java code?**

* answer

### **Explain Spring boot application? How does it work?**

* answer

# Microservices

### **Monolith vs Microservices differences and when to choose which?**

* answer

### **How to design a microservice from scratch?**

* answer

### **API Gateway pattern and its advantages?**

* answer

### **Inter-service communication: REST vs Messaging?**

* answer

### **Circuit Breaker pattern and its implementation using Resilience4j?**

* answer

### **Load balancing in microservices using Spring Cloud Load Balancer?**

* Answer

### **How Spring Cloud Config helps in centralized configuration management?**

* answer

### **Service discovery using Eureka or Consul?**

* answer

### **Feign Client vs WebClient: Which one to use and why?**

* answer

### **Event-driven architecture and Kafka integration?**

* answer

### **Database per service vs Shared Database: Pros and cons?**

* answer

### **Saga Pattern for distributed transactions in microservices?**

* answer

### **JWT-based authentication and OAuth2 in microservices?**

* answer

### **How to handle security in an API Gateway?**

* answer

### **Observability: Logging, tracing, and monitoring best practices?**

* answer

### **Role of Prometheus and Grafana in microservices monitoring?**

* answer

### **Kubernetes deployment strategies for microservices?**

* answer

### **Blue-Green and Canary deployments in microservices?**

* answer

### **When to use WebFlux for reactive microservices?**

* answer

### **CQRS and Event Sourcing: When and why to use them?**

* answer

# SQL

### **What is ordered by and range in the database?**

* Answer

### **Write a SQL query to get the second highest salary of an employee from the employee table?**

* Answer

### **Write a SQL query to print employee id, employee\_name, depatment\_name Employee - employee\_id, employee\_name, department\_id**

### **Department - department\_id, department\_name?**

* answer

### **What is a collision in the oracle database?**

* Answer

### **If you create multiple folders one inside another and store one file into the last folder then how can you implement using database tables in oracle? How many tables are required?**

* answer

### **What are DDL and DML commands?**

* Answer

### **Write a query to update data in the table?**

* Answer

### **What is the use of joins in the database?**

* Answer

### **Explain ACID properties in the database?**

* Answer

### **Explain Normalization?**

* Answer

### **What are the types of Normalization?**

* Answer

### **What is Denormalization?**

* Answer

### **What is the use of join?**

* Answer

### **Write a SQL query to return the 7th highest salary of an employee?**

* Answer

### **What is Union in SQL?**

* Answer

### **Can we perform union operations on one table?**

* Answer

### **What is the difference between where and let?**

* Answer

### **Explain ACID properties in the database?**

* Answer

# Docker

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# Kubernetes

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# AWS

### **What is cloud computing?**

* answer

### **Explain services provided by cloud computing?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# Azure

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# Angular

### **What are the components in the Angular?**

* answer

### **How to pass data from components in Angular?**

* answer

### **What does the component.ts file contain in angular?**

* answer

### **Explain two way data binding in angular?**

* answer

### **How to make backend calls in angular?**

* answer

### **What is authguard in angular?**

* answer

### **What is interpolation in angular?**

* answer

### **How to create components in angular?**

* answer

### **Explain angular directives?**

* answer

### **What are observables in angular?**

* answer

### **How to implement dependency injection in angular?**

* answer

### **question?**

* answer

# React

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

### **question?**

* answer

# Others

### **Explain Jira?**

* answer

### **What is Version control i.e. Git?**

* answer

### **What is the output of below program?**

#include<stdio.h>

int main()

{

int a =0;

1 + 1 - 1 + 1

a=a++ + ++a - a++ + ++a;

printf(“%d\n”,a);

return 0;

}

* answer

### **What is https protocol?**

* answer

### **How does https internally work?**

* answer

### **What is a certificate?**

* Answer

### **Difference between http and https?**

* answer

### **What is a Sprint retrospective?**

* answer

### **question?**

* answer

### **question?**

* answer

# Linux

### **Explain Linux commands?**

* answer

### **Command - Read a file and print first two lines from file Ans. head -2 filename?**

* answer

### **Search a word from a file?**

* grep 'word' filename ack 'pattern' /path/to/file.txt

### **Why do you want to change your current organization?**

* answer

# Puzzles

### **Weighing the 9 balls puzzle, find the heavier ball among 9 balls. How many max iterations will it be required to find the heavier ball?**

* answer