1. OOPS Concepts
    a. A.P.I.E. – Abstraction, Polymorphism, Inheritance, Encapsulation.
    b. SOLID
    c. Class design

2. Core Java

    a. Abstract Class
    b. Inner class / static inner class
    c. Immutable class
    d. Interface
    e. Marker Interface
    f. Exception handling
    g. Garbage Collection
    h. Class loader/ static and dynamic loading
    i. Comparable / Comparator Interface
    j. String pool / String Buffer/ String Builder
    k. Constructor chaining / in case of abstract class/ interface
    l. This and super keywords
    m. Serialization
    n. Iterator / List Iterator
    o. Rules of overloading and overriding
    p. Reflection
    q. Exception Hierarchy and finally Method, ConcurrentModificationException
    r. Pass by value / pass by reference
    s. Locking(class level vs instance level)
    t. Synchronization
    u. Locking mechanisms
    v. Serializable & Externalizable and cloanable Interfaces
    w. Finalize/clone method of object class

3. Collection Framework(internals)
    a. Array / Array List
    b. Linked List
    c. Vector (rarely asked)
    d. Hash Map
    e. hash Table
    f. Linked Hash Map
    g. Tree Map
    h. Sorted Map
    i. WeakHashMap
    j. All kinds of Sets/ Hash Set/Tree Set
    k. LinkedHashSet

l. Stack / Queue/ Priority Queue/ Blocking Queue
m. Condition Interface
n. Fail safe and fail fast iterator
o. CopyOnWriteArrayList
p. ConcurrentSkipListMap
q. ConcurrentHashMap
r. Collections.unmodifiableCollection()

4. Multithreading and Concurrency
    a. Thread lifecycle and basics
    b. Volatile
    c. Synchronize
    d. Race condition
    e. Deadlocks
    f. BlockingQueue / Producer Consumer problem
    g. Synchronizers like CyclicBarrier, CountdownLatch
    h. Phaser
    i. Atomic classes

5. Java 8 topics

6. Design Patterns / Sorting Algorithms
    a. Singletons
    b. Visitor
    c. Template
    d. Decorator
    e. Strategy
    f. Observer
    g. Façade /session Façade
    h. Factory /Abstract Factory
    i. DAO

7. Spring Core
    a. Bean Factory
    b. Application Context
    c. Bean Life Cycle
    d. Init / destroy methods
    e. Bean Listeners
    f. Processors
    g. Scopes
    h. Loading mechanisms
    i. IOC

8. Database (SQL/PLSQL)
    a. DDL
    b. DML
    c. Delete/truncate/Drop
    d. Union / Union All
    e. Index/ clustered- non clustered index (including implementations at DS level)
    f. Procedure
    g. Group by/ having
    h. Count(*) Max , Avg, etc
    i. Join (types of joins)
    j. Primary Kay / Unique Key
    k. Isolation levels
    l. ACID properties

9. Java Performance Tuning
    a. GC algorithm names only
    b. Heap memory settings
    c. strong, soft, weak and Phantom reference
    d. Stack and Heap Concept

10. Analytical/Logical /Scenario Based questions.
    a. LRU dictionary or Cache
    b. ATM/Library/HR dept design
    c. Parking allocation
    d. Find most frequently used word from text file
    e. Sorting 10 MB file using 1 MB memory
    f. 1 billion cellphone numbers to finds duplicates
    g. Find duplicate number in Integer Array
    h. Identify palindrome
    i. Fibonacci series printing using recursive
    j. Calculate factorial using recursive and iterative
    k. Implement single elevator , double elevator
    l. Simulate DVD renting system
    m. etc

Sample questions below:

Question Set 1

1. Design a stack that supports getMin() in O(1) time and O(1) extra space.

2. Program for n’th node from the end of a Linked List

3. Semaphore in java 8, print odd and even number using semaphore

4. How ArrayList works internally in Java 8

5. find second largest number in array without sorting in java

6. Sort an array of 0s, 1s and 2s

7. Reverse a linked list

8. Garbage collection algorithms

9. Implement two stacks in an array

10. Producer-Consumer solution using threads in Java

Question Set 2

1. Implement database connection pooling using semaphore

2. Countdown latch/cyclic barrier -explain, difference between cyclic barrier and countdown latch

3. How HashMap works internally in Java 8

4. Function to check if a singly linked list is palindrome

5. Atomic variable -How it works internally

6. Difference between Callable and Runnable

7. Detect and Remove Loop in a Linked List

8. CopyOnWriteArrayList implementation

9. Find first unique character in a String

10. Implement Multithreading application which demonstrates deadlocks and how to avoid deadlocks.

Question Set 3:

1. Find position of an element in a sorted array of infinite numbers

2. How ConcurrentHashMap works internally in Java 8

3. BlockingQueue-Expalin, implement own ArrayBlockingQueue

4. ReentrantLock implementation

5. Intersection point of two Linked Lists.

6. Creating custom exceptions

7. Design a vending machine

8. Java Reference- Soft, Weak, Strong and Phantom

9. Sort an array of 0s, and 1s

10. Different and best approach for Singleton Pattern

Queue Set 4:

1. Search an element in a sorted and rotated array

2. How TreeSet works internally in Java 8

3. UnModifiable collection own implementation

4. Java 8 new features

5. largest-sum-contiguous-subarray

6. Tree traversal with implementation [preorder, postorder, inorder and mirror]

7. Design multi-level parking system

8. Map sort by value

9. Design Principle

10. find the middle element in a linked list

    11. Implement StringPool -Flyweight Design Pattern

A typical interview:

1. Core Java

a. ArrayList vs LinkedList, vector - when to use which of these

b. Hashmap, equals and hashcode, Collison, comparator (deeper understanding)

c. Immutability and making a custom class mutable (inner workings)

d. Synchronize Hashmap and Concurrent Hashmap

e. Thread and Executer service (Practical knowledge/Deeper understanding)

i. Lock levels and question around that

ii. Benefits of using Executor Service

f. Garbage Collection

i. Memory Partitions and benefits

ii. Major vs minor GC

g. Data Structures - implementation of a LRU Cache

2. Logic and problem solving

a. Estimate value of 2^24

b. In an array of integers, find out odd occurring integer in O(n) time. Given all numbers appear even times except one integer.

c. Implement an org structure and give 2 APIs for getting all the managers and all the employees.

d. Asking questions around the problem proactively, for a deeper understanding of the same so as to develop an effective solution

3. Databases

a. Questions around Functions and Procedures

b. Queries using “group by”

c. Indexes

d. Joins

4. Communication

a. Ability to explain the thoughts clearly and be able to interact with team members across multiple global locations