1. OOPS Concepts
   1. A.P.I.E.
2. Core Java
   1. Abstract Class
   2. Inner class / static inner class
   3. Immutable class
   4. Interface
   5. Marker Interface
   6. Exception handling
   7. Garbage Collection
   8. Class loader/ static and dynamic loading
   9. Comparable / Comparator Interface
   10. String pool / String Buffer/ String Builder
   11. Constructor chaining / in case of abstract class/ interface
   12. This and super keywords
   13. Serialization
   14. Iterator / List Iterator
   15. Rules of overloading and overriding
   16. Reflection
   17. Exception Hierarchy and finally Method, ConcurrentModificationException
   18. Pass by value / pass by reference
   19. Locking(class level vs instance level)
   20. Synchronization
   21. Locking mechanisms
   22. Serializable & Externalizable and cloanable Interfaces
   23. Finalize/clone method of object class
3. Collection Framework(internals)
   1. Array / Array List
   2. Linked List
   3. Vector (rarely)
   4. Hash Map
   5. hash Table
   6. Linked Hash Map
   7. Tree Map
   8. Sorted Map
   9. WeakHashMap
   10. All kinds of Sets/ Hash Set/Tree Set
   11. LinkedHashSet
   12. Stack / Queue/ Priority Queue/ Blocking Queue
   13. Condition Interface
   14. Producer – consumer , race condition
   15. Fail safe and fail fast iterator
   16. CopyOnWriteArrayList
   17. ConcurrentSkipListMap
   18. ConcurrentHashMap
   19. Collections.unmodifiableCollection()
4. Design Patterns / Sorting Algorithms
   1. Singletons
   2. Visitor
   3. Template
   4. Decorator
   5. Strategy
   6. Observer
   7. Façade /session Façade
   8. Factory /Abstract Factory
   9. DAO
5. Spring Core
   1. Bean Factory
   2. Application Context
   3. Bean Life Cycle
   4. Init / destroy methods
   5. Bean Listeners
   6. Processors
   7. Scopes
   8. Loading mechanisms
   9. IOC
6. Database (SQL/PLSQL)
   1. DDL
   2. DML
   3. Delete/truncate/Drop
   4. Union / Union All
   5. Index/ clustered- non clustered index (including implementations at DS level)
   6. Procedure
   7. Group by/ having
   8. Count(\*) Max , Avg, etc
   9. Join (types of joins)
   10. Primary Kay / Unique Key
   11. Isolation levels
   12. ACID properties
7. Java Performance Tuning
   1. GC algorithm names only
   2. Heap memory settings
   3. strong, soft, weak and Phantom reference
   4. Stack and Heap Concept
8. Analytical/Logical /Scenario Based questions.
   1. LRU dictionary or Cache
   2. ATM/Library/HR dept design
   3. Parking allocation
   4. Find most frequently used word from text file
   5. Sorting 10 MB file using 1 MB memory
   6. 1 billion cellphone numbers to finds duplicates
   7. Find duplicate number in Integer Array
   8. Identify palindrome
   9. Fibonacci series printing using recursive
   10. Calculate factorial using recursive and iterative
   11. Implement single elevator , double elevator
   12. Simulate DVD renting system
   13. 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