**Java Coding questions:**

1. **Given a list of custom objects, sort the list based on various parameter of the object. Suppose you have list of employee object then sort the list based on name, salary etc.**

**Ans:**

**Class Employee {**

**String name;**

**double salary;**

**public Employee(String name, double salary){**

**this.name = name;**

**this.salary = salary;**

**}**

**}**

**List<Employee> empList = new ArrayList<Employee>();**

**empList.add(new Employee(“Rakesh”, 200.0));**

**empList.add(new Employee(“Sharma”, 300.0));**

**empList.add(new Employee(“Talari”, 800.0));**

**// Method 1: Using streams and lamda expration**

**// sorting by name**

**List<Employee> sortedByNameList = empList.stream().sorted(Comparator.comparing(Employee::getName)).collect(Collectors.toList());**

**// sorting by salary**

**List<Employee> sortedBySalaryList = empList.stream().sorted(Comparator.comparingInt(Employee::getSalary)).collect(Collectors.toList());**

**// Method 2: Using comparator interface**

**Class SalaryComparator implements Comparator<Employee>{**

**// implementing compare method of Comparator interface**

**public int compare(Employee emp, Employee emp1)**

**{**

**return emp.salary – emp1.salary;**

**}**

**}**

**Class NameComparator implements Comparator<Employee>{**

**// implementing compare method of Comparator interface**

**public int compare(Employee emp, Employee emp1)**

**{**

**return emp.name.compareTo(emp1.name);**

**}**

**}**

**// sorting by name**

**Collections.sort(empList, new NameComparator());**

**// sorting by salary**

**Collections.sort(empList, new NameComparator());**

1. **Create a Deadlock situation programmatically.**

**Ans:**

**Let us assume we have two threads or processes (T1, T2) and two resources (R1, R2).**

* **Thread (T1) requires Resources R1 and R2 in respective order to complete the task.**
* **Thread (T2) requires Resources R2 and R1in respective order to complete the task.**

**Here T1 will acquire a lock on resource R1 and waiting for R2 to get unlocked by T2. At the same time T2 will acquire a lock on resource R2 and waiting for R1 to get unlocked by T1.**

**Both the threads are unable to complete the task by waiting for resources to get unlocked.**

**This is called Deadlock situation.**

**Program:**

**public class DeadLock {**

**// Two resources shared between threads**

**private static final Object r1 = new Object();**

**private static final Object r2 = new Object();**

**public static void main(String[] args) {**

**// t1 locks r1 and tries to lock r2**

**Thread t1 = new Thread(() -> {**

**synchronized (r1) {**

**System.out.println("T1: Locked r1");**

**try {**

**Thread.sleep(100);**

**} catch (InterruptedException e) {**

**Thread.currentThread().interrupt();**

**}**

**System.out.println("T1: Trying to lock r2");**

**synchronized (r2) {**

**System.out.println("T1: Locked r2");**

**}**

**}**

**});**

**// T2 locks r2 and tries to lock r1**

**Thread t2 = new Thread(() -> {**

**synchronized (r2) {**

**System.out.println("T2: Locked r2");**

**try {**

**Thread.sleep(100);**

**} catch (InterruptedException e) {**

**Thread.currentThread().interrupt();**

**}**

**System.out.println("T2: Trying to lock r1");**

**synchronized (r1) {**

**System.out.println("T2: Locked r1");**

**}**

**}**

**});**

**// Start both threads**

**t1.start();**

**t2.start();**

**}**

**}**

1. **Check for Balanced Brackets in an expression**

**Example:**

**Input: exp = “[()]{}{[()()]()}”**

**Output: Balanced**

**Explanation: all the brackets are well-formed**

**Input: exp = “[(])”**

**Output: Not Balanced**

**Explanation: 1 and 4 brackets are not balanced because**

**there is a closing ‘]’ before the closing ‘(‘**

**Ans:**

**package BalancedBrackets;**

**import java.util.Stack;**

**public class BalancedBrackets {**

**public static void main(String[] args) {**

**String exp = "[()]{}{[()()]()}";**

**if( isBalanced(exp)){**

**System.out.println(exp + "has Balnced Brackets");**

**}else{**

**System.out.println(exp + "has Balnced Brackets");**

**}**

**}**

**public static boolean isBalanced(String exp){**

**Stack<Character> stack = new Stack<>();**

**for(char ch: exp.toCharArray()){**

**if (ch == '(' || ch == '{' || ch == '[' )**

**{**

**stack.push(ch);**

**}**

**else {**

**if ( ! stack.empty() &&  (( stack.peek() == '(' && ch == ')') || ( stack.peek() == '{' && ch == '}') || (stack.peek() == '[' && ch== ']')))**

**{**

**stack.pop();**

**}**

**else {**

**return  false;**

**}**

**}**

**}**

**return stack.empty();**

**}**

**}**

1. **Implement the linked list in java.**

**Ans:**

**Node.java**

**public class Node {**

**int data;**

**Node next;**

**Node(int data){**

**this.data = data;**

**this.next = null;**

**}**

**}**

**LinkedList.java**

**package LinkedList;**

**public class LinkedList {**

**private Node head;**

**public LinkedList() {**

**this.head = null;**

**}**

**// get first node of list**

**public Node getFirst(){**

**return this.head;**

**}**

**// get last node of list**

**public Node getLast(){**

**Node current = head;**

**while (current.next != null) {**

**current = current.next;**

**}**

**return current;**

**}**

**// add node at first**

**public void addFirst(int data){**

**Node newNode = new Node(data);**

**newNode.next = head;**

**this.head = newNode;**

**}**

**// add node at last**

**public void addLast(int data){**

**Node newNode = new Node(data);**

**Node lastNode = getLast();**

**lastNode.next = newNode;**

**}**

**// remove first node**

**public void removeFirst(){**

**this.head = this.head.next;**

**}**

**// remove last node**

**public void removeLast(){**

**Node current = head;**

**Node prv = null;**

**while (current.next != null) {**

**prv = current;**

**current = current.next;**

**}**

**prv.next = null;**

**}**

**}**

1. How do you find the second largest number in an array in Java?

**Ans:**

**package SecondLargest;**

**public class SecondLargest {**

**int findSecoundLargest(int[] arr) {**

**int largest = arr[0];**

**int secondLargest = arr[0];**

**for (int num : arr) {**

**if (num > largest) {**

**secondLargest = largest;**

**largest = num;**

**} else if (num > secondLargest && num < largest) {**

**secondLargest = num;**

**}**

**}**

**return secondLargest;**

**}**

**}**

1. **Write a Java program that will print the number of occurrences of each character in a string.**

**Ans:**

**package CharacterFrequency;**

**import java.util.HashMap;**

**import java.util.Map;**

**public class CharacterFrequency {**

**Map<Character, Integer> frequency = new HashMap<Character, Integer>();**

**String str = new String("Rakesh");**

**for(char ch: str.toCharArray()){**

**if(frequency.containsKey(ch)){**

**frequency.put(ch, frequency.get(ch)+1);**

**}else{**

**frequency.put(ch, 1);**

**}**

**}**

**}**

**7. Write java program to shift the element to the right by one index.**

**Ans:**

**package RightShift;**

**public class RightShift {**

**public static void main(String[] args) {**

**int[] arr = new int[]{1, 2, 3};**

**if(arr.length > 0){**

**int last = arr[arr.length-1];**

**for(int j = arr.length-1; j > 0; j--){**

**//Shift element to right**

**arr[j] = arr[j-1];**

**}**

**arr[0] = last;**

**}**

**}**

**}**

**React Coding Questions:**

1. **Write a JavaScript function to return Fibonacci numbers till n.**

**Ans:**

**function fibonacci(n){**

**let n1 = 0;**

**let n2 = 1;**

**let nextTerm;**

**for(let i=1; i <=n; i++){**

**console.log(n1);**

**nextTerm = n1 + num2;**

**n1 = n2;**

**n2 = nextTerm;**

**}**

**}**

1. **In the array, [1,1,2,2,2,3,3,4,5,5,5]; write a Javascript function to show the elements with no of repetitions.**

**Ans:**

**function frequency(str){**

**const freq = new Map();**

**// Count frequency**

**for (let ch of str) {**

**freq.set(ch, (freq.get(ch) || 0) + 1);**

**}**

**// print**

**for (let x of [...freq.keys()]) {**

**console.log(x + ": " + freq.get(x));**

**}**

**}**

1. **Write a javascript function to check if a string is palindrome or not?**

**Ans:**

**function isPalindrome(str) {**

**let j = str.length - 1**

**for (let i = 0; i < str.length / 2; i++) {**

**if (str[i] != str[j]) {**

**return false;**

**}**

**j--;**

**}**

**return true;**

**}**