FULL STACK DEVELOPMENT-WORKSHEET-B ANSWERS

Q.1 What is the output for the below code ?

A. 6

Q3. Write a java program to Remove Duplicates elements from Array List0 ,?

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;
class GFG
{
   public static void main(String[] args)
   {
      List<Integer> list = new ArrayList<>(
         Arrays.asList(1, 10, 1, 2, 2, 3, 10, 3, 3, 4, 5, 5));

      System.out.println("ArrayList with duplicates: "
                  + list);

      List<Integer> newList = list.stream()
                           .distinct()
                           .collect(Collectors.toList());
   System.out.println("ArrayList with duplicates removed: "
                  + newList);
   }
}
```

Q4. Write a java Program to Union and Intersection of two Linked List ?

A. import java.util.*;

```
public class Doublyll {
   static class Node
   {
      int data;
      Node next;
      Node prev;
      Node(int data)
      {
         this.data=data;
         this.next=null;
         this.prev=null;
      }
   }
   Node head=null;
   Node tail=null;
   public void creation()
   {
      int data, n, m,p;
      Scanner s = new Scanner(System.in);
      do{
         System.out.println("enter data");
         data=s.nextInt();
         Node new_node=new Node(data);
         if(head==null)
         {
            head=new_node;
            tail=new_node;
         }
         else
         {
            System.out.println("Enter 1 to insert at beginning, 2 to insert at end and 3 to insert at
specific position");
            m=s.nextInt();
            switch(m)
            {
               case 1:
            new_node.next=head;
            head.prev=new_node;
            head=new_node;
            break;
            case 2:
            tail.next=new_node;
            new_node.prev=tail;
            tail=new_node;
            break;
            case 3:
            System.out.println("Enter position of node to be inserted");
            p=s.nextInt();
```

```
            Node temp1=head;
            Node ptr=temp1.next;
            for(int i=1;i<(p-1); i++)
            {
               temp1=ptr;
               ptr=ptr.next;
            }
            new_node.prev=temp1;
            new_node.next=ptr;
            temp1.next=new_node;
            ptr.prev=new_node;
            break;
            }
         }
         System.out.println(" Do you want to add more data? if yes press 1, if not press any key");
         n=s.nextInt();
      }
      while(n==1);
   }
   public void traverse()
   {
    Node temp=head;
    if(head==null)
    {
      System.out.println("LL doesn't exist");
    }
    else{
      while(temp!=null)
      {
         System.out.println(temp.data);
         temp=temp.next;
      }
    }
   }
   public static void main(String[] args) {
     Doublyll ll = new Doublyll();
      ll.creation();
      ll.traverse();
   }
}
```

Q5 Write a java Program to Sum of middle row and column in Matrix?

A.
```
import java.util.*;
public class MatrixSumMiddle {
   public static void main(String[] args) {
      Scanner s = new Scanner(System.in);
```

```
        System.out.print("Enter the number of rows: ");
        int rows = s.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols = s.nextInt();
        int[][] matrix = new int[rows][cols];
        System.out.println("Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = s.nextInt();
            }
        }
        int middleRow = rows / 2;
        int middleCol = cols / 2;
        int sumMiddleRow = 0;
        for (int j = 0; j < cols; j++) {
            sumMiddleRow += matrix[middleRow][j];
        }
        int sumMiddleCol = 0;
        for (int i = 0; i < rows; i++) {
            sumMiddleCol += matrix[i][middleCol];
        }
        System.out.println("The matrix is:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(matrix[i][j] + "\t");
            }
            System.out.println();
        }
        System.out.println("Sum of middle row: " + sumMiddleRow);
        System.out.println("Sum of middle column: " + sumMiddleCol);
        s.close();
    }
}
```

Q.6 Write a java Program Merge two sorted linked lists,

A.

```
import java.util.*;
public class Mergesort {
    public static void conquer(int arr[], int si, int mid, int ei) {
        int merged[] = new int[ei - si + 1];
        int idx1 = si;
        int idx2 = mid + 1;
        int x = 0;
        while (idx1 <= mid && idx2 <= ei) {
            if (arr[idx1] <= arr[idx2]) {
                merged[x++] = arr[idx1++];
            } else {
                merged[x++] = arr[idx2++];
```

```
        }
      }
      while (idx1 <= mid) {
        merged[x++] = arr[idx1++];
      }
      while (idx2 <= ei) {
        merged[x++] = arr[idx2++];
      }
       for (int i = 0, j = si; i < merged.length; i++, j++) {
        arr[j] = merged[i];
      }
   }
   public static void divide(int arr[], int si, int ei) {
      if (si < ei) {
        int mid = si + (ei - si) / 2;
        divide(arr, si, mid);
        divide(arr, mid + 1, ei);
        conquer(arr, si, mid, ei);
      }
   }
   public static void main(String[] args) {
      int arr[] = { 6, 3, 9, 5, 2, 8 };
      int n = arr.length;
      divide(arr, 0, n - 1);
      for (int i = 0; i < n; i++) {
        System.out.print(arr[i] + " ");
      }
      System.out.println();
   }
}
```

Q7. Write a java Program to Print Bottom View of Binary Tree ?

A.
```
import java.util.*;
import java.util.Map.Entry;
class Node
{
        int data;
        int hd;
        Node left, right;
        public Node(int key)
        {
                data = key;
                hd = Integer.MAX_VALUE;
                left = right = null;
        }
}
class Tree
```

```
{
	Node root;
	public Tree() {}
	public Tree(Node node)
	{
		root = node;
	}
	public void bottomView()
	{
		if (root == null)
			return;
      int hd = 0;
		Map<Integer, Integer> map = new TreeMap<>();
		Queue<Node> queue = new LinkedList<Node>();
		root.hd = hd;
		queue.add(root);
		while (!queue.isEmpty())
		{
			Node temp = queue.remove();
			hd = temp.hd;
			map.put(hd, temp.data);
			if (temp.left != null)
			{
				temp.left.hd = hd-1;
				queue.add(temp.left);
			}
			if (temp.right != null)
			{
				temp.right.hd = hd+1;
				queue.add(temp.right);
			}
		}
		Set<Entry<Integer, Integer>> set = map.entrySet();
		Iterator<Entry<Integer, Integer>> iterator = set.iterator();

		while (iterator.hasNext())
		{
			Map.Entry<Integer, Integer> me = iterator.next();
			System.out.print(me.getValue()+" ");
		}
	}
}


public class BottomView
{
	public static void main(String[] args)
	{
```

```
                Node root = new Node(20);
                root.left = new Node(8);
                root.right = new Node(22);
                root.left.left = new Node(5);
                root.left.right = new Node(3);
                root.right.left = new Node(4);
                root.right.right = new Node(25);
                root.left.right.left = new Node(10);
                root.left.right.right = new Node(14);
                Tree tree = new Tree(root);
                System.out.println("Bottom view of the given binary tree:");
                tree.bottomView();
        }
}
```

Q.8 Write a java Program to Convert a Binary Tree into its Mirror Tree

```
class Node {
   int data;
   Node left, right;

   public Node(int item)
   {
      data = item;
      left = right = null;
   }
}

class BinaryTree {
   Node root;

   void mirror() { root = mirror(root); }

   Node mirror(Node node)
   {
      if (node == null)
         return node;
      Node left = mirror(node.left);
      Node right = mirror(node.right);
      node.left = right;
      node.right = left;
      return node;
   }

   void inOrder() { inOrder(root); }
   void inOrder(Node node)
   {
      if (node == null)
```

```
            return;

        inOrder(node.left);
        System.out.print(node.data + " ");

        inOrder(node.right);
    }
    public static void main(String args[])
    {
        BinaryTree tree = new BinaryTree();
        tree.root = new Node(1);
        tree.root.left = new Node(2);
        tree.root.right = new Node(3);
        tree.root.left.left = new Node(4);
        tree.root.left.right = new Node(5);
        System.out.println(
            "Inorder traversal of the constructed tree is");
        tree.inOrder();
        System.out.println("");
        tree.mirror();
        System.out.println(
            "Inorder traversal of the mirror tree is");
        tree.inOrder();
    }
}
```

Q9. Write a java Program to Determine if given Two Trees are Identical or not,

A.

```
import java.util.*;
class Node {
    int data;
    Node left, right;
    Node(int item)
    {
        data = item;
        left = right = null;
    }
}
class BinaryTree {
    Node root1, root2;
   boolean identicalTrees(Node a, Node b)
    {
        if (a == null && b == null)
            return true;
        if (a != null && b != null)
            return (a.data == b.data
                    && identicalTrees(a.left, b.left)
```

```
                && identicalTrees(a.right, b.right));
        return false;
    }
    public static void main(String[] args)
    {
        BinaryTree tree = new BinaryTree();

        tree.root1 = new Node(1);
        tree.root1.left = new Node(2);
        tree.root1.right = new Node(3);
        tree.root1.left.left = new Node(4);
        tree.root1.left.right = new Node(5);
        tree.root2 = new Node(1);
        tree.root2.left = new Node(2);
        tree.root2.right = new Node(3);
        tree.root2.left.left = new Node(4);
        tree.root2.left.right = new Node(5);
        if (tree.identicalTrees(tree.root1, tree.root2))
            System.out.println("Both trees are identical");
        else
            System.out.println("Trees are not identical");
    }
}
```

Q10. Write a java Program to find whether a no is power of two or not

A.
```
import java.util.*;
public class PowerOfTwo {
    public static boolean isPowerOfTwo(int n) {
        if (n <= 0) {
            return false;
        }
        return (n & (n - 1)) == 0;
    }

    public static void main(String[] args) {
        int num = 16;
        if (isPowerOfTwo(num)) {
            System.out.println(num + " is a power of two.");
        } else {
            System.out.println(num + " is not a power of two.");
        }
    }
}
```