**64. Write a Java program to insert a specified element at a given position in a linked list.**

Import java.util.LinkedList;

Public class InsertInLinkedList {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“0873CS231138”);

LinkedList<String> list = new LinkedList<>();

List.add(“Apple”);

List.add(“Banana”);

List.add(“Cherry”);

System.out.println(“Original LinkedList: “ + list);

String elementToInsert = “Orange”;

Int position = 1; // Index is zero-based, so 1 means insert after “Apple”

List.add(position, elementToInsert);

System.out.println(“Updated LinkedList after insertion: “ + list);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231138

Original LinkedList: [Apple, Banana, Cherry]

Updated LinkedList after insertion: [Apple, Orange, Banana, Cherry]

**65. Write a Java program to insert elements at the first and last positions of a linked list.**

Import java.util.LinkedList;

Public class InsertFirstLast {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“0873CS231138”);

LinkedList<String> list = new LinkedList<>();

List.add(“Middle”);

List.addFirst(“First”);

List.addLast(“Last”);

System.out.println(“Updated list: “ + list);

}

}

**OUTPUT**:

YESHIKA KOLTE

0873CS231121

Updated list: [First, Middle, Last]

**66. Write a Java program to add all elements from one TreeSet to another TreeSet.**

Import java.util.TreeSet;

Public class TreeSetAddAll {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“0873CS231138”);

Public static void main(String[] args) {

TreeSet<String> set1 = new TreeSet<>();

Set1.add(“Apple”);

Set1.add(“Banana”);

TreeSet<String> set2 = new TreeSet<>();

Set2.add(“Mango”);

Set2.addAll(set1);

System.out.println(“Merged TreeSet: “ + set2);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Merged TreeSet: [Apple, Banana, Mango]

**67. Write a Java program to display the elements of a TreeSet in reverse order.**

Import java.util.TreeSet;

Import java.util.Iterator;

System.out.println(“YESHIKA KOLTE”);

System.out.println(“0873CS231138”);

Public class TreeSetReverse {

Public static void main(String[] args) {

TreeSet<String> set = new TreeSet<>();

Set.add(“Apple”);

Set.add(“Banana”);

Set.add(“Mango”);

Iterator<String> itr = set.descendingIterator();

System.out.print(“Reverse order: “);

While (itr.hasNext()) {

System.out.print(itr.next() + “ “);

}

}

}

**OUTPUT**:

YESHIKA KOLTE

0873CS23138

Reverse order: Mango Banana Apple

**68. Write a Java program to retrieve the first and last elements from a TreeSet.**

Import java.util.TreeSet;

Public class TreeSetFirstLast {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

TreeSet<Integer> numbers = new TreeSet<>();

Numbers.add(10);

Numbers.add(50);

Numbers.add(30);

System.out.println(“First element: “ + numbers.first());

System.out.println(“Last element: “ + numbers.last());

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

First element: 10

Last element: 50

**69. Write a Java program to clone a TreeSet into another TreeSet**

Import java.util.TreeSet;

Public class TreeSetClone {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

TreeSet<String> original = new TreeSet<>();

Original.add(“Red”);

Original.add(“Green”);

TreeSet<String> clone = (TreeSet<String>) original.clone();

System.out.println(“Cloned TreeSet: “ + clone);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Cloned TreeSet: [Green, Red]

**70. Write a Java program to count the number of elements in a TreeSet.**

Import java.util.TreeSet;

Public class TreeSetCount {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

TreeSet<String> set = new TreeSet<>();

Set.add(“A”);

Set.add(“B”);

Set.add(“C”);

System.out.println(“Total elements: “ + set.size());

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Total elements: 3

**71. Write a Java program to compare two TreeSets.**

Import java.util.TreeSet;

Public class TreeSetCompare {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

TreeSet<String> set1 = new TreeSet<>();

Set1.add(“Red”);

Set1.add(“Green”);

TreeSet<String> set2 = new TreeSet<>();

Set2.add(“Green”);

Set2.add(“Red”);

System.out.println(“Sets are equal: “ + set1.equals(set2));

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Sets are equal: true

**72. Write a Java program to clone one HashSet into another.**

Import java.util.HashSet;

Public class HashSetClone {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashSet<String> original = new HashSet<>();

Original.add(“Dog”);

Original.add(“Cat”);

HashSet<String> clone = (HashSet<String>) original.clone();

System.out.println(“Cloned HashSet: “ + clone);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Cloned HashSet: [Cat, Dog]

**73. Write a Java program to convert a HashSet into an array.**

Import java.util.HashSet;

Public class HashSetToArray {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashSet<String> set = new HashSet<>();

Set.add(“One”);

Set.add(“Two”);

String[] array = new String[set.size()];

Set.toArray(array);

System.out.println(“Array elements:”);

For (String s : array) {

System.out.println(s);

}

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Array elements:

Two

One

**74. Write a Java program to convert a HashSet into a TreeSet.**

Import java.util.HashSet;

Import java.util.TreeSet;

Public class HashSetToTreeSet {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

Public static void main(String[] args) {

HashSet<String> hashSet = new HashSet<>(); hashSet.add(“Banana”); hashSet.add(“Apple”);

TreeSet<String> treeSet = new TreeSet<>(hashSet);

System.out.println(“TreeSet: “ + treeSet);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

TreeSet: [Apple, Banana]

**75. Write a Java program to find numbers less than 7 in a TreeSet**.

Import java.util.TreeSet;

Public class TreeSetLessThan {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

TreeSet<Integer> numbers = new TreeSet<>();

Numbers.add(2);

Numbers.add(5);

Numbers.add(8);

Numbers.add(10);

System.out.println(“Numbers less than 7: “ + numbers.headSet(7));

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Numbers less than 7: [2, 5]

**76. Write a Java program to compare two HashSets.**

Import java.util.HashSet;

Public class HashSetCompare {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashSet<String> set1 = new HashSet<>();

Set1.add(“X”);

Set1.add(“Y”);

HashSet<String> set2 = new HashSet<>();

Set2.add(“Y”);

Set2.add(“X”);

System.out.println(“Are sets equal? “ + set1.equals(set2));

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Are sets equal? True

**77. Write a Java program to retain common elements from two sets.**

Import java.util.HashSet;

Public class HashSetRetain {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashSet<String> set1 = new HashSet<>();

Set1.add(“A”);

Set1.add(“B”);

HashSet<String> set2 = new HashSet<>();

Set2.add(“B”);

Set2.add(“C”);

Set1.retainAll(set2);

System.out.println(“Common elements: “ + set1);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Common elements: [B]

**78. Write a Java program to remove all elements from a HashSet.**

Import java.util.HashSet;

Public class HashSetRemoveAll {

Public static void main(String[] args) {

HashSet<String> set = new HashSet<>();

Set.add(“Java”);

Set.add(“Python”);

Set.clear();

System.out.println(“HashSet after clear(): “ + set);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

HashSet after clear(): []

**79. Write a Java program to copy all mappings from one map to another.**

Import java.util.HashMap;

Public class HashMapCopy {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashMap<Integer, String> map1 = new HashMap<>();

Map1.put(1, “One”);

Map1.put(2, “Two”);

HashMap<Integer, String> map2 = new HashMap<>();

Map2.putAll(map1);

System.out.println(“Copied HashMap: “ + map2);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Copied HashMap: {1=One, 2=Two}

**80. Write a Java program to remove all key-value pairs from a map.**

Import java.util.HashMap;

Public class HashMapClear {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashMap<String, String> map = new HashMap<>();

Map.put(“A”, “Apple”);

Map.put(“B”, “Banana”);

Map.clear();

System.out.println(“Map after clear(): “ + map);

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Map after clear(): {}

**81. Write a Java program to check if a map is empty or contains key-value mappings.**

Import java.util.HashMap;

Public class HashMapIsEmpty {

Public static void main(String[] args) {

HashMap<Integer, String> map = new HashMap<>();

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

System.out.println(“Is map empty? “ + map.isEmpty());

Map.put(10, “Ten”);

System.out.println(“Is map empty after adding element? “ + map.isEmpty());

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Is map empty? True

Is map empty after adding element? False

**82. Write a Java program to create a shallow copy of a HashMap instance.**

Import java.util.HashMap;

Public class HashMapShallowCopy {

Public static void main(String[] args) {

HashMap<String, String> original = new HashMap<>();

Original.put(“Name”, “YESHIKA KOLTE”);

HashMap<String, String> copy = (HashMap<String, String>) original.clone();

System.out.println(“Shallow copy of map: “ + copy);

}

}

**OUTPUT:**

Shallow copy of map: {Name=YESHIKA KOLTE}

**83. Write a Java program to test whether a specified key exists in the map.**

Import java.util.HashMap;

Public class HashMapContainsKey {

Public static void main(String[] args) {

System.out.println(“YESHIKA KOLTE”);

System.out.println(“O873CS231138”);

HashMap<Integer, String> map = new HashMap<>();

Map.put(1, “A”);

Map.put(2, “B”);

Int keyToCheck = 2;

If (map.containsKey(keyToCheck)) {

System.out.println(“Key “ + keyToCheck + “ exists with value: “ + map.get(keyToCheck));

} else {

System.out.println(“Key “ + keyToCheck + “ does not exist.”);

}

}

}

**OUTPUT:**

YESHIKA KOLTE

0873CS231121

Key 2 exists with value: B

**84. Create a table Item\_dtls (Electronics)**

Try to insert at least 10 records in the above table

Try to insert at least 2 records with null value

CREATE TABLE Item\_dtls (

Item\_id INT PRIMARY KEY,

Item\_name VARCHAR(50),

Category VARCHAR(30),

Price DECIMAL(10, 2)

);

INSERT INTO Item\_dtls VALUES (1, ‘Smartphone’, ‘Mobile’, 25000);

INSERT INTO Item\_dtls VALUES (2, ‘Laptop’, ‘Computers’, 60000);

INSERT INTO Item\_dtls VALUES (3, ‘TV’, ‘Home Appliances’, 30000);

INSERT INTO Item\_dtls VALUES (4, ‘Headphones’, ‘Accessories’, 2000);

INSERT INTO Item\_dtls VALUES (5, ‘Camera’, ‘Photography’, 35000);

INSERT INTO Item\_dtls VALUES (6, ‘Smartwatch’, ‘Wearable’, 10000);

INSERT INTO Item\_dtls VALUES (7, ‘Microwave’, ‘Kitchen’, 8000);

INSERT INTO Item\_dtls VALUES (8, ‘Printer’, ‘Office’, 5000);

INSERT INTO Item\_dtls VALUES (9, NULL, ‘Gaming’, 40000);

INSERT INTO Item\_dtls VALUES (10, ‘Speaker’, NULL, 3000);

**85. Create a table Sales\_dtls**

Try to insert at least 10 records in the above table

Try to insert at least 2 records with null value

CREATE TABLE Sales\_dtls (

Sale\_id INT PRIMARY KEY,

Item\_id INT,

Quantity INT,

Sale\_date DATE

);

INSERT INTO Sales\_dtls VALUES (1, 1, 2, ‘2025-01-10’);

INSERT INTO Sales\_dtls VALUES (2, 2, 1, ‘2025-01-11’);

INSERT INTO Sales\_dtls VALUES (3, 3, 1, ‘2025-01-12’);

INSERT INTO Sales\_dtls VALUES (4, 4, 5, ‘2025-01-13’);

INSERT INTO Sales\_dtls VALUES (5, 5, 3, ‘2025-01-14’);

INSERT INTO Sales\_dtls VALUES (6, 6, 2, ‘2025-01-15’);

INSERT INTO Sales\_dtls VALUES (7, 7, 1, ‘2025-01-16’);

INSERT INTO Sales\_dtls VALUES (8, 8, 4, ‘2025-01-17’);

INSERT INTO Sales\_dtls VALUES (9, 9, NULL, ‘2025-01-18’);

INSERT INTO Sales\_dtls VALUES (10, 10, 2, NULL);

**86.create a table manufacturers**

Try to insert at least 10 records in the above table

Try to insert at least 2 records with null value

Consider the below tables with estimated columns and then practise below questions.

CUST DTLS

CUST Act DTLS

ACT\_TYPES\_INFO

PROD\_DTLS

EMP

DEPT

CREATE TABLE manufacturers (

Mfg\_id INT PRIMARY KEY,

Mfg\_name VARCHAR(50),

Country VARCHAR(50)

);

INSERT INTO manufacturers VALUES (1, ‘Samsung’, ‘South Korea’);

INSERT INTO manufacturers VALUES (2, ‘Apple’, ‘USA’);

INSERT INTO manufacturers VALUES (3, ‘Sony’, ‘Japan’);

INSERT INTO manufacturers VALUES (4, ‘LG’, ‘South Korea’);

INSERT INTO manufacturers VALUES (5, ‘Dell’, ‘USA’);

INSERT INTO manufacturers VALUES (6, ‘HP’, ‘USA’);

INSERT INTO manufacturers VALUES (7, ‘Panasonic’, ‘Japan’);

INSERT INTO manufacturers VALUES (8, ‘Xiaomi’, ‘China’);

INSERT INTO manufacturers VALUES (9, NULL, ‘China’);

INSERT INTO manufacturers VALUES (10, ‘Realme’, NULL);

**87)Fetch all clerks information.**

SELECT \* FROM EMP WHERE job = ‘CLERK’;

**88)Display all departments information located at CHICAGO?**

SELECT \* FROM DEPT WHERE location = ‘CHICAGO’;

**89)Display product details manufactured in the current year only?**

SELECT \* FROM PROD\_DTLS WHERE EXTRACT(YEAR FROM manufacture\_date) = EXTRACT(YEAR FROM SYSDATE);

**90)Get the details of cutomers accounts who opened the accounts before this year**?

SELECT \* FROM CUST\_ACT\_DTLS WHERE account\_open\_date < TRUNC(SYSDATE, ‘YYYY’); 90) Get all SALARY account details?

SELECT \* FROM CUST\_ACT\_DTLS WHERE account\_type = ‘SALARY’;

**91) Display customer names and mobile numbers from the city ‘Texas’?**

Select cname, mobile from cust\_dtls where city=’Texas’; 92) Get the information of Trading account?

SELECT \* FROM CUST\_ACT\_DTLS WHERE account\_type = ‘TRADING’; 93) Display only Expired product details?

SELECT \* FROM prod\_dtls WHERE exp < SYSDATE;