**PG-DAC AUGUST 24 BATCH**

1)Write a Java program that takes a list of integers as input and returns a list of duplicate integers.

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

public class DuplicateFinder {

public static List<Integer> findDuplicates(List<Integer> numbers) {

Set<Integer> seen = new HashSet<>();

List<Integer> duplicates = new ArrayList<>();

for (int number : numbers) {

if (!seen.add(number)) {

duplicates.add(number);

}

}

return duplicates;

}

public static void main(String[] args) {

List<Integer> input = new ArrayList<>();

input.add(1);

input.add(2);

input.add(3);

input.add(2);

input.add(1);

input.add(4);

List<Integer> duplicates = findDuplicates(input);

System.out.println("Duplicate integers: " + duplicates);

}

}

2)Create a Person class with attributes name and age. Write a Java program that sorts a list of Person objects first by age and then by name if the ages are equal.

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

class Person {

private String name;

private int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

@Override

public String toString() {

return "Person{" +

"name='" + name + '\'' +

", age=" + age +

'}';

}

}

public class PersonSorter {

public static void main(String[] args) {

List<Person> people = new ArrayList<>();

people.add(new Person("Alice", 25));

people.add(new Person("Bob", 30));

people.add(new Person("Charlie", 25));

people.add(new Person("David", 30));

// Sort by age and then by name

Collections.sort(people, new Comparator<Person>() {

@Override

public int compare(Person p1, Person p2) {

int ageComparison = p1.getAge() - p2.getAge();

if (ageComparison == 0) {

return p1.getName().compareTo(p2.getName());

} else {

return ageComparison;

}

}

});

// Print the sorted list

for (Person person : people) {

System.out.println(person);

}

}

}

3)Write a Java program to find the first non-repeated character in a string using a HashMap.

String input = "aabbccddeffg";

Expected output = 'e';

import java.util.HashMap;

import java.util.Map;

public class FirstNonRepeatingCharacter {

public static char findFirstNonRepeatingChar(String input) {

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

for (char c : input.toCharArray()) {

charCounts.put(c, charCounts.getOrDefault(c, 0) + 1);

}

for (char c : input.toCharArray()) {

if (charCounts.get(c) == 1) {

return c;

}

}

return '\0'; // No non-repeating character found

}

public static void main(String[] args) {

String input = "aabbccddeffg";

char result = findFirstNonRepeatingChar(input);

System.out.println("First non-repeating character: " + result);

}

}

4) Write a Java program that merges two sorted lists of integers into a single sorted list.

import java.util.ArrayList;

import java.util.List;

public class MergeSortedLists {

public static List<Integer> mergeSortedLists(List<Integer> list1, List<Integer> list2) {

List<Integer> mergedList = new ArrayList<>();

int i = 0, j = 0;

while (i < list1.size() && j < list2.size()) {

if (list1.get(i) <= list2.get(j)) {

mergedList.add(list1.get(i));

i++;

} else {

mergedList.add(list2.get(j));

j++;

}

}

// Add remaining elements from list1 or list2

while (i < list1.size()) {

mergedList.add(list1.get(i));

i++;

}

while (j < list2.size()) {

mergedList.add(list2.get(j));

j++;

}

return mergedList;

}

public static void main(String[] args) {

List<Integer> list1 = new ArrayList<>();

list1.add(1);

list1.add(3);

list1.add(5);

List<Integer> list2 = new ArrayList<>();

list2.add(2);

list2.add(4);

list2.add(6);

List<Integer> mergedList = mergeSortedLists(list1, list2);

System.out.println("Merged list: " + mergedList);

}

}