Practice Program: 05-08-2024

1. Create and Implement the java program by using map.

Program:

import java.util.HashMap;

import java.util.Map;

public class MapExample {

public static void main(String[] args) {

Map<String, Integer> studentGrades = new HashMap<>();

studentGrades.put("John", 85);

studentGrades.put("Alice", 92);

studentGrades.put("Bob", 78);

studentGrades.put("Charlie", 95);

System.out.println("Student Grades:");

printMap(studentGrades);

System.out.println("\nGet John's Grade:");

System.out.println(studentGrades.get("John"));

studentGrades.put("John", 90);

System.out.println("\nUpdate John's Grade:");

System.out.println(studentGrades.get("John"));

studentGrades.remove("Bob");

System.out.println("\nRemove Bob's Grade:");

printMap(studentGrades);

System.out.println("\nContains Key 'Alice':");

System.out.println(studentGrades.containsKey("Alice"));

System.out.println("\nContains Value 95:");

System.out.println(studentGrades.containsValue(95));

System.out.println("\nMap Size:");

System.out.println(studentGrades.size());

studentGrades.clear();

System.out.println("\nClear Map:");

printMap(studentGrades);

}

public static void printMap(Map<String, Integer> map) {

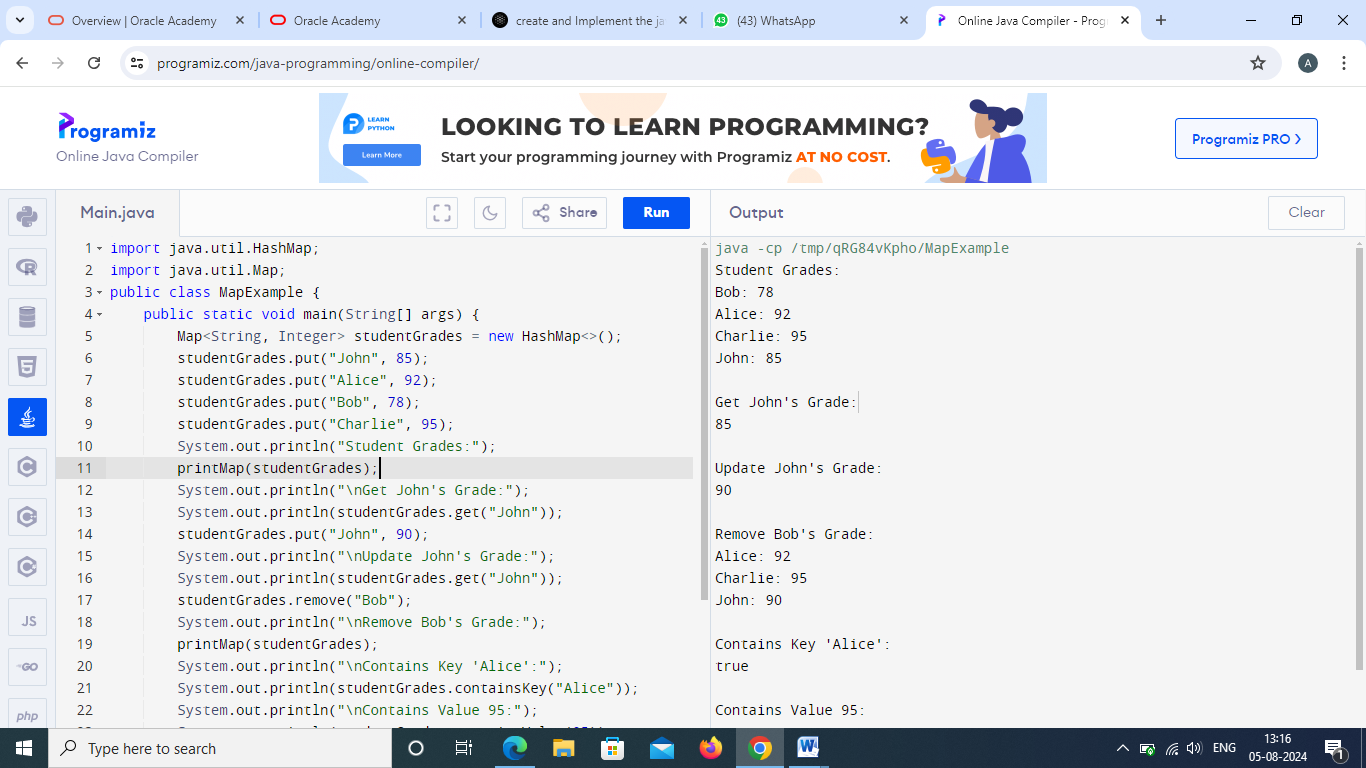
for (Map.Entry<String, Integer> entry : map.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}



1. Create a program using linked list for stack.

Program:

import java.util.LinkedList;

public class Stack{

public static void main(String args[]){

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

letter.push("apple");

letter.push("ball");

letter.push("cat");

letter.push("dog");

System.out.println("Linked list : "+letter);

System.out.println("Stack Size: " +letter.size());

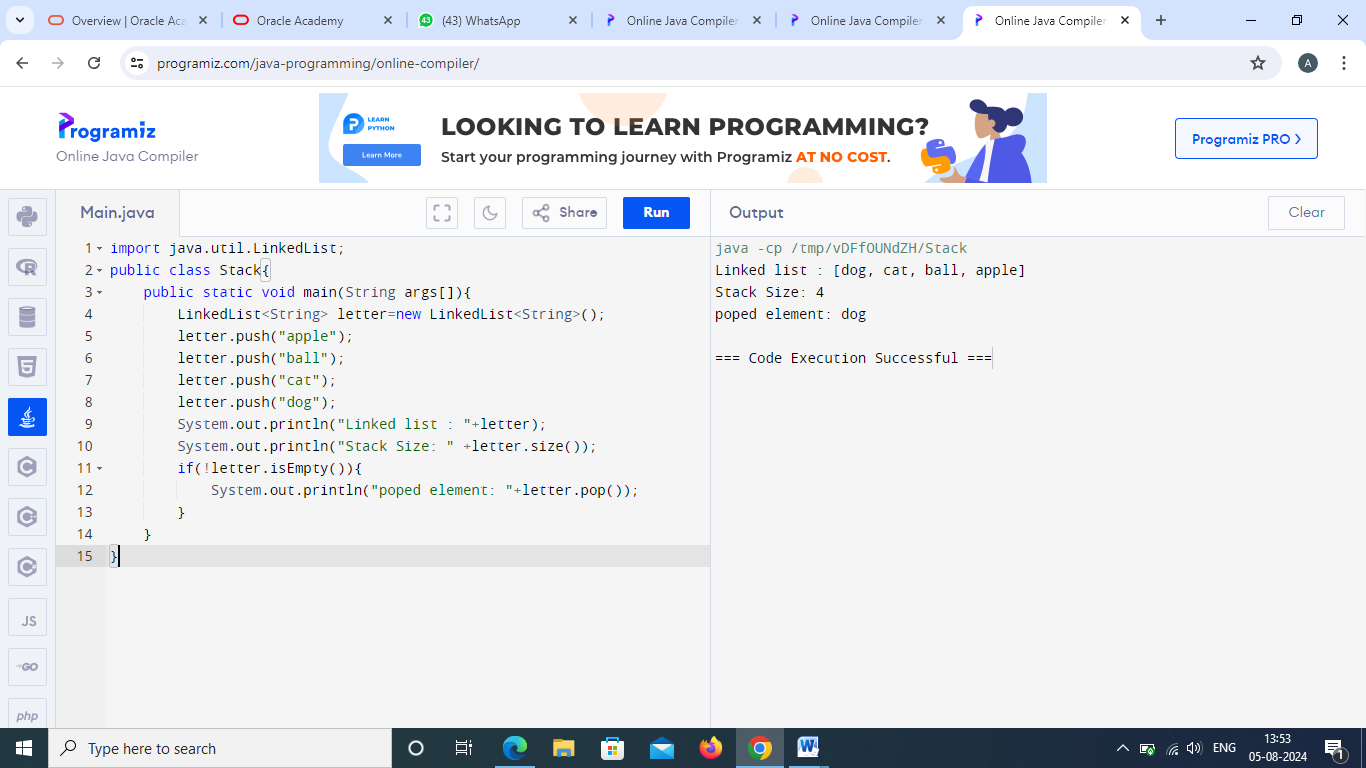
if(!letter.isEmpty()){

System.out.println("poped element: "+letter.pop());

}

}

}



1. Create a program using linked list for queue.

Program:

import java.util.LinkedList;

public class Queue{

public static void main(String args[]){

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

letter.add("a");

letter.add("b");

letter.add("c");

letter.add("d");

System.out.println("Linked list : "+letter);

System.out.println("Queue Size: "+letter.size());

if(!letter.isEmpty()){

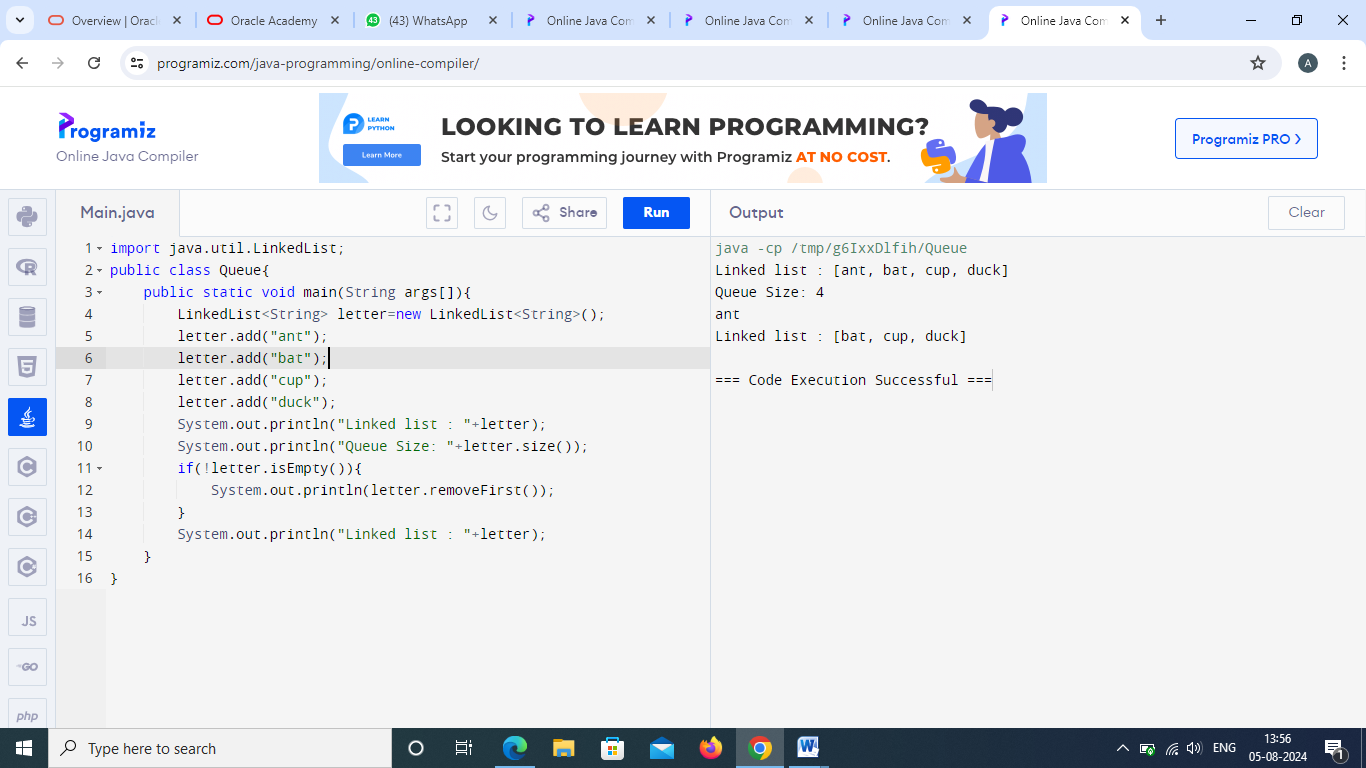
System.out.println(letter.removeFirst());

}

System.out.println("Linked list : "+letter);

}

}



4. Create mobile class with field price brand and so on take any one field and sort the class

Program:

import java.util.Arrays;

import java.util.Comparator;

class Mobile {

private double price;

private String brand;

private String model;

private int ram;

public Mobile(double price, String brand, String model, int ram) {

this.price = price;

this.brand = brand;

this.model = model;

this.ram = ram;

}

public double getPrice() {

return price;

}

public String getBrand() {

return brand;

}

public String getModel() {

return model;

}

public int getRam() {

return ram;

}

public String toString() {

return "Mobile{" +

"price=" + price +

", brand='" + brand + '\'' +

", model='" + model + '\'' +

", ram=" + ram +

'}';

}

}

public class MobileSorter {

public static void main(String[] args) {

Mobile[] mobiles = new Mobile[] {

new Mobile(15000, "Samsung", "Galaxy M31", 6),

new Mobile(20000, "Apple", "iPhone 12", 4),

new Mobile(10000, "Xiaomi", "Redmi Note 9", 4),

new Mobile(25000, "OnePlus", "Nord", 8),

new Mobile(12000, "Realme", "6 Pro", 6)

};

System.out.println("Before sorting:");

Arrays.stream(mobiles).forEach(System.out::println);

Arrays.sort(mobiles, Comparator.comparingDouble(Mobile::getPrice));

System.out.println("\nAfter sorting by price:");

Arrays.stream(mobiles).forEach(System.out::println);

}

}

