**WEEK-11**

**Question 1**

**Java HashSet class implements the Set interface, backed by a hash table which is actually a**[**HashMap**](https://www.geeksforgeeks.org/java-util-hashmap-in-java/)**instance.**

**No guarantee is made as to the iteration order of the hash sets which means that the class does not guarantee the constant order of elements over time.**

**This class permits the null element.**

**The class also offers constant time performance for the basic operations like add, remove, contains, and size assuming the hash function disperses the elements properly among the buckets.**

**Java HashSet Features**

**A few important features of HashSet are mentioned below:**

* **Implements**[**Set Interface**](https://www.geeksforgeeks.org/set-in-java/)**.**
* **The underlying data structure for HashSet is [Hashtable](https://www.geeksforgeeks.org/hashtable-in-java/).**
* **As it implements the Set Interface, duplicate values are not allowed.**
* **Objects that you insert in HashSet are not guaranteed to be inserted in the same order. Objects are inserted based on their hash code.**
* **NULL elements are allowed in HashSet.**
* **HashSet also implements Serializable and Cloneable interfaces.**
* **public class HashSet<E> extends AbstractSet<E> implements Set<E>, Cloneable, Serializable  
  Sample Input and Output:  
  5  
  90  
  56  
  45  
  78  
  25  
  78  
  Sample Output:  
  78 was found in the set.  
  Sample Input and output:  
  3  
  2  
  7  
  9  
  5  
  Sample Input and output:  
  5 was not found in the set.**

**Program:**

**import java.util.HashSet;**

**import java.util.Scanner;**

**public class HashSetExample {**

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

**Scanner scanner = new Scanner(System.in);**

**// Read the number of elements**

**int n = scanner.nextInt();**

**// Initialize HashSet**

**HashSet<Integer> set = new HashSet<>();**

**// Read elements into the HashSet**

**for (int i = 0; i < n; i++) {**

**set.add(scanner.nextInt());**

**}**

**// Read the target element to search**

**int target = scanner.nextInt();**

**// Check if the element is in the set**

**if (set.contains(target)) {**

**System.out.println(target + " was found in the set.");**

**} else {**

**System.out.println(target + " was not found in the set.");**

**}**

**scanner.close();**

**}**

**}**

****

**Question 2**

**Write a Java program to compare two sets and retain elements that are the same.**

**Sample Input and Output:**

**5**

**Football**

**Hockey**

**Cricket**

**Volleyball**

**Basketball**

**7      // HashSet 2:**

**Golf**

**Cricket**

**Badminton**

**Football**

**Hockey**

**Volleyball**

**Handball**

**SAMPLE OUTPUT:**

**Football**

**Hockey**

**Cricket**

**Volleyball**

**Basketball**

**Program:**

**import java.util.HashSet;**

**import java.util.Scanner;**

**public class CompareSets {**

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

**Scanner scanner = new Scanner(System.in);**

**// Read size and elements of the first HashSet**

**int n1 = scanner.nextInt();**

**scanner.nextLine(); // Consume newline**

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

**for (int i = 0; i < n1; i++) {**

**set1.add(scanner.nextLine());**

**}**

**// Read size and elements of the second HashSet**

**int n2 = scanner.nextInt();**

**scanner.nextLine(); // Consume newline**

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

**for (int i = 0; i < n2; i++) {**

**set2.add(scanner.nextLine());**

**}**

**// Retain only common elements in set1**

**set1.retainAll(set2);**

**// Print the common elements**

**for (String sport : set1) {**

**System.out.println(sport);**

**}**

**scanner.close();**

**}**

**}**

****

**Question 3**

**Java HashMap Methods**

[**containsKey()**](https://www.w3schools.com/java/ref_hashmap_containskey.asp)**Indicate if an entry with the specified key exists in the map**

[**containsValue()**](https://www.w3schools.com/java/ref_hashmap_containsvalue.asp)**Indicate if an entry with the specified value exists in the map**

[**putIfAbsent()**](https://www.w3schools.com/java/ref_hashmap_putifabsent.asp)**Write an entry into the map but only if an entry with the same key does not already exist**

[**remove()**](https://www.w3schools.com/java/ref_hashmap_remove.asp)**Remove an entry from the map**

[**replace()   Write to an entry in the map only if it exists**](https://www.w3schools.com/java/ref_hashmap_replace.asp)

[**size()**](https://www.w3schools.com/java/ref_hashmap_size.asp)**Return the number of entries in the map**

**Your task is to fill the incomplete code to get desired output**

**Program:**

**import java.util.HashMap;**

**import java.util.Map.Entry;**

**import java.util.Set;**

**import java.util.Scanner;**

**class prog {**

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

**// Creating HashMap with default initial capacity and load factor**

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

**String name;**

**int num;**

**Scanner sc = new Scanner(System.in);**

**int n = sc.nextInt();**

**for (int i = 0; i < n; i++) {**

**name = sc.next();**

**num = sc.nextInt();**

**map.put(name, num);**

**}**

**// Printing key-value pairs**

**Set<Entry<String, Integer>> entrySet = map.entrySet();**

**for (Entry<String, Integer> entry : entrySet) {**

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

**}**

**System.out.println("----------");**

**// Creating another HashMap**

**HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();**

**// Inserting key-value pairs to anotherMap using put() method**

**anotherMap.put("SIX", 6);**

**anotherMap.put("SEVEN", 7);**

**// Inserting key-value pairs of map to anotherMap using putAll() method**

**anotherMap.putAll(map); // code here**

**// Printing key-value pairs of anotherMap**

**entrySet = anotherMap.entrySet();**

**for (Entry<String, Integer> entry : entrySet) {**

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

**}**

**// Adds key-value pair 'FIVE-5' only if it is not present in map**

**map.putIfAbsent("FIVE", 5);**

**// Retrieving a value associated with key 'TWO'**

**int value = map.getOrDefault("TWO", -1); // Using getOrDefault for safety**

**System.out.println(value);**

**// Checking whether key 'ONE' exist in map**

**System.out.println(map.containsKey("ONE")); // Filled code here**

**// Checking whether value '3' exist in map**

**System.out.println(map.containsValue(3)); // Filled code here**

**// Retrieving the number of key-value pairs present in map**

**System.out.println(map.size()); // Filled code here**

**sc.close();**

**}**

**}**

****