# CS23333-Object Oriented Programming Using Java-2023

Dashboard / My courses / CS23333-OOPUJ-2023 / Lab-11-Set, Map / Lab-11-Logic Building

Quiz navigation

1 2 3

Show one page at a time

Finish review

| | |
|---|---|
| **Status** | Finished |
| **Started** | Sunday, 17 November 2024, 10:44 PM |
| **Completed** | Sunday, 17 November 2024, 11:38 PM |
| **Duration** | 53 mins 56 secs |

Question **1**

Incorrect

Marked out of 1.00

Flag question

**Java HashSet** class implements the Set interface, backed by a hash table which is actually a HashMap 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.
- The underlying data structure for HashSet is Hashtable.
- 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.
```

**Answer:** (penalty regime: 0 %)

Reset answer

```
    // Method to check if a number is present in the HashSet
    public void checkIfNumberExists(HashSet<Integer> set, int num) {
        if (set.contains(num)) {
            System.out.println(num + " was found in the set.");
        } else {
            System.out.println(num + " was not found in the set.");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of test cases: ");

        int testCases = 0;
        if (sc.hasNextInt()) {
            testCases = sc.nextInt();
        } else {
            System.out.println("Invalid input. Exiting program.");
            sc.close();
            return;
        }

        for (int t = 0; t < testCases; t++) {
            HashSet<Integer> set = new HashSet<>();
            System.out.print("Enter the number of elements to add to the set: ");

            int n = 0;
            if (sc.hasNextInt()) {
                n = sc.nextInt();
            } else {
                System.out.println("Invalid input for number of elements. Skipping test ca
                sc.next(); // Skip invalid input
                continue;
            }
```

```java

            System.out.println("Enter " + n + " integers:");
            new HashSetExample().addElementsToSet(set, n, sc);

            System.out.print("Enter the number to search in the set: ");
            if (sc.hasNextInt()) {
                int num = sc.nextInt();
                new HashSetExample().checkIfNumberExists(set, num);
            } else {
                System.out.println("Invalid input for number to search. Skipping test case
                sc.next(); // Skip invalid input
            }
        }

        sc.close();
    }
}
```

| | Test | Input | Expected | Got |
|---|---|---|---|---|
| | 1 | 5<br>90<br>56<br>45<br>78<br>25<br>78 | 78 was found in the set. | Enter number of test cases: Enter the number of elements to add to th<br>Invalid input. Please enter integers only.<br><br>***Run error***<br>Exception in thread "main" java.util.NoSuchElementException<br>at java.base/java.util.Scanner.throwFor(Scanner.java:937)<br>at java.base/java.util.Scanner.next(Scanner.java:1478)<br>at HashSetExample.addElementsToSet(HashSetExample.java:15)<br>at HashSetExample.main(HashSetExample.java:56) |

Testing was aborted due to error.

Your code must pass all tests to earn any marks. Try again.

Show differences

Question **2**

Correct

Marked out of 1.00

Flag question

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

**Answer:** (penalty regime: 0 %)

```java
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;

public class SetIntersection {

    // Method to input a set of strings
    public static Set<String> inputSet(Scanner sc, int n) {
        Set<String> set = new HashSet<>();
        for (int i = 0; i < n; i++) {
            set.add(sc.nextLine()); // Adding elements to the set
        }
        return set;
    }

    // Method to find the intersection of two sets
    public static Set<String> findIntersection(Set<String> set1, Set<String> set2) {
        Set<String> intersection = new HashSet<>(set1); // Create a copy of set1
        intersection.retainAll(set2); // Retains only the elements that are in both sets
        return intersection;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Input the first set of strings
        int n1 = sc.nextInt();
        sc.nextLine(); // Consume the newline character after the integer input
        Set<String> set1 = inputSet(sc, n1);

        // Input the second set of strings
        int n2 = sc.nextInt();
        sc.nextLine(); // Consume the newline character after the integer input
        Set<String> set2 = inputSet(sc, n2);

        // Find the intersection of the two sets
        Set<String> intersection = findIntersection(set1, set2);

        // Display the intersection
        for (String sport : intersection) {
            System.out.println(sport);
        }

        // Close the scanner object
        sc.close();
    }
}
```

| | Test | Input | Expected | Got | |
|---|---|---|---|---|---|
| | 1 | 5<br>Football<br>Hockey<br>Cricket<br>Volleyball<br>Basketball<br>7<br>Golf<br>Cricket<br>Badminton<br>Football<br>Hockey<br>Volleyball<br>Throwball | Cricket<br>Hockey<br>Volleyball<br>Football | Cricket<br>Hockey<br>Volleyball<br>Football | |
| | 2 | 4<br>Toy<br>Bus<br>Car<br>Auto<br>3<br>Car<br>Bus<br>Lorry | Bus<br>Car | Bus<br>Car | |

Passed all tests!

Question **3**

Correct

Marked out of 1.00

Flag question

Java HashMap Methods

containsKey() Indicate if an entry with the specified key exists in the map

containsValue() Indicate if an entry with the specified value exists in the map

putIfAbsent() Write an entry into the map but only if an entry with the same key does not already exist

remove() Remove an entry from the map

replace() Write to an entry in the map only if it exists

size() Return the number of entries in the map

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

**Answer:** (penalty regime: 0 %)

Reset answer

```
import java.util.HashMap;
import java.util.Map.Entry;
import java.util.Set;
import java.util.Scanner;

public class Prog {
    public static void main(String[] args) {
        HashMap<String, Integer> map = new HashMap<String, Integer>();
        String name;
        int num;
        Scanner sc = new Scanner(System.in);

        // Reading number of entries
```

```java
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            name = sc.next();  // Read name (key)
            num = sc.nextInt(); // Read number (value)
            map.put(name, num);
        }

        // Printing the entries of the first map
        Set<Entry<String, Integer>> entrySet = map.entrySet();
        for (Entry<String, Integer> entry : entrySet) {
            System.out.println(entry.getKey() + " : " + entry.getValue());
        }

        // Divider line
        System.out.println("----------");

        // Creating another map and copying contents of the first map
        HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();
        anotherMap.put("SIX", 6);
        anotherMap.put("SEVEN", 7);
        anotherMap.putAll(map);

        // Printing the entries of the second map
        entrySet = anotherMap.entrySet();
        for (Entry<String, Integer> entry : entrySet) {
            System.out.println(entry.getKey() + " : " + entry.getValue());
        }

        // Using putIfAbsent to add a new key if not present
        map.putIfAbsent("FIVE", 5);

        // Retrieving value associated with the key "TWO"
        Integer value = map.get("TWO");
        if (value != null) {
            System.out.println(value);
        } else {
            System.out.println("Key 'TWO' not found in map.");
        }

```

|  | Test | Input | Expected | Got |  |
|---|---|---|---|---|---|
|  | 1 | 3<br>ONE<br>1<br>TWO<br>2<br>THREE<br>3 | ONE : 1<br>TWO : 2<br>THREE : 3<br>----------<br>SIX : 6<br>ONE : 1<br>TWO : 2<br>SEVEN : 7<br>THREE : 3<br>2<br>true<br>true<br>4 | ONE : 1<br>TWO : 2<br>THREE : 3<br>----------<br>SIX : 6<br>ONE : 1<br>TWO : 2<br>SEVEN : 7<br>THREE : 3<br>2<br>true<br>true<br>4 |  |

Passed all tests!

Finish review

◄ Lab-11-MCQ

Jump to...

TreeSet example ►