

[Dashboard](#) / [My courses](#) / [CS23333-OOPJ-2023](#) / [Lab-11-Set\\_Map](#) / [Lab-11-Logic Building](#)

<b>Status</b>	Finished
<b>Started</b>	Tuesday, 19 November 2024, 7:49 AM
<b>Completed</b>	Tuesday, 19 November 2024, 8:25 AM
<b>Duration</b>	35 mins 48 secs

Question 1

Correct

Marked out of 1.00

**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

```
1 import java.util.HashSet;
2 import java.util.Scanner;
3 class prog {
4     public static void main(String[] args) {
5         Scanner sc= new Scanner(System.in);
6         int n = sc.nextInt();
7         // Create a HashSet object called numbers
8         HashSet<Integer> numbers = new HashSet<>();
9
10        // Add values to the set
11        for(int i=0;i<n;i++)
12            numbers.add(sc.nextInt());
13
14        int skey=sc.nextInt();
15
16        // Show which numbers between 1 and 10 are in the set
17        if(numbers.contains(skey))
18        {
19            System.out.println( skey + " was found in the set.");
20        } else {
21            System.out.println(skey + " was not found in the set.");
22        }
23    }
24 }
25
```

	Test	Input	Expected	Got	
✓	1	5 90 56 45 78 25 78	78 was found in the set.	78 was found in the set.	✓
✓	2	3 -1 2 4 5	5 was not found in the set.	5 was not found in the set.	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

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 %)

```
1 import java.util.*;
2 class prog{
3     public static void main(String[] args){
4         Scanner sc = new Scanner(System.in);
5         int n = sc.nextInt();
6         HashSet<String> st1 = new HashSet<>();
7         for(int i = 0; i < n; i++){
8             st1.add(sc.next().trim());
9         }
10        HashSet<String> st2 = new HashSet<>();
11        int m = sc.nextInt();
12        for(int i = 0; i < m; i++){
13            st2.add(sc.next().trim());
14        }
15        for(String str : st1){
16            if(st2.contains(str)){
17                System.out.println(str);
18            }
19        }
20    }
21 }
```

	Test	Input	Expected	Got	
✓	1	5 Football Hockey Cricket Volleyball Basketball 7 Golf Cricket Badminton Football Hockey Volleyball Throwball	Cricket Hockey Volleyball Football	Cricket Hockey Volleyball Football	✓
✓	2	4 Toy Bus Car Auto 3 Car Bus Lorry	Bus Car	Bus Car	✓

Passed all tests! ✓

Question 3

Correct

Marked out of 1.00

## 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

```

1 import java.util.HashMap;
2 import java.util.Map.Entry;
3 import java.util.Set;
4 import java.util.Scanner;
5 class prog
6 {
7     public static void main(String[] args)
8     {
9         //Creating HashMap with default initial capacity and load factor
10        HashMap<String, Integer> map = new HashMap<String, Integer>();
11
12        String name;
13        int num;
14        Scanner sc= new Scanner(System.in);
15        int n=sc.nextInt();
16        for(int i =0;i<n;i++)
17        {
18            name=sc.next();
19            num= sc.nextInt();
20            map.put(name,num);
21        }
22
23        //Printing key-value pairs
24
25        Set<Entry<String, Integer>> entrySet = map.entrySet();
26
27        for (Entry<String, Integer> entry : entrySet)
28        {
29            System.out.println(entry.getKey()+" : "+entry.getValue());
30        }
31        System.out.println("-----");
32        //Creating another HashMap
33
34        HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();
35
36        //Inserting key-value pairs to anotherMap using put() method
37
38        anotherMap.put("SIX", 6);
39
40        anotherMap.put("ONE", 1);
41
42        //Inserting key-value pairs of map to anotherMap using putAll() method
43
44        anotherMap.put("TWO",2);
45        anotherMap.put("SEVEN",7);
46        anotherMap.put("THREE",3);
47        // code here
48
49        //Printing key-value pairs of anotherMap
50
51        entrySet = anotherMap.entrySet();
52

```

	Test	Input	Expected	Got	
✓	1	3 ONE 1 TWO 2 THREE 3	ONE : 1 TWO : 2 THREE : 3 ----- SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	ONE : 1 TWO : 2 THREE : 3 ----- SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	✓

Passed all tests! ✓

[◀ Lab-11-MCQ](#)

Jump to...

⌵

[TreeSet example ▶](#)