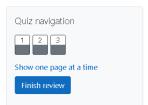
# CS23333-Object Oriented Programming Using Java-2023





Correct Marked out of

▼ 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

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:
 56
 Sample Output:
  78 was found in the set.
  Sample Input and output:
  Sample Input and output:
  5 was not found in the set.
```

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

```
Reset answer
  1 - import java.util.HashSet;
      pupport Java.util.Redsises;
import java.util.Scanner;
public class prog {
  public static void main(String[] args) {
    Scanner sc= new Scanner(System.in);
    int n = sc.nextInt();
    // Create a HashSet object called numbers
            HashSet<Integer> numbers = new HashSet<>();
            // Add values to the set
            for(int i=0:i<n:i++)
 11
            numbers.add(sc.nextInt());
 13
 14
         int skey=sc.nextInt();
 15
 16
            // Show which numbers between 1 and 10 are in the set
 17
                 18
 19
 20
                 System.out.println(skey + " was not found in the set.");
 22
 24 }
25 }
```

T	st Inpu	t Expected	Got	
<b>v</b> 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.	~

Question 2 Correct Marked out of

▼ Flag question

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

#### Sample Input and Output:

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 %)

```
12
13
                     }
int n2 = scn.nextInt();
                     Int n2 = scin.nexcin();
scn.nextLine();
Set<string> set2 = new HashSet<>();
for(int i = 0;i<n2;i++){
    set2.add(scn.nextLine());</pre>
    14
15
    16
17
   18
19
                      set1.retainAll(set2);
                     for(String item : set1){
    System.out.println(item);
   20
   21
   23
24 }
               }
```

	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	~			
<b>~</b>	2	4 Toy Bus Car Auto 3 Car Bus Lorry	Bus Car	Bus Car	~			
Passed all tests! ✓								

# Question ${\bf 3}$

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

DuttfAbsent\() 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;
       import java.util.Map.Entry;
import java.util.Set;
       import java.util.Scanner:
    5
6 v
           public static void main(String[] args)
    8
                //Creating HashMap with default initial capacity and load factor
   10
                HashMap<String, Integer> map = new HashMap<String, Integer>();
   11
   12
13
                String name;
                int num;
                Scanner sc= new Scanner(System.in);
int n=sc.nextInt();
   14
15
   16
                 for(int i =0;i<n;i++)</pre>
   17
                {
   18
                     name=sc.next();
                     num= sc.nextInt();
   19
                     map.put(name,num);
                }
  21
                //Printing key-value pairs
  23
  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("----");
                //Creating another HashMap
   32
33
                HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();
   34
                //Inserting key-value pairs to anotherMap using put() method
   36
   37
   38
                anotherMap.put("SIX", 6);
   39
  40
41
                anotherMap.put("SEVEN", 7);
  42
                //Inserting key-value pairs of map to anotherMap using putAll() method
  43
  44
                anotherMap.putAll(map); // code here
  45
                //Printing key-value pairs of anotherMap
  47
                entrySet = anotherMap.entrySet();
  49
   50
                for (Entry<String, Integer> entry : entrySet)
   51
   52
                    System.out.println(entry.getKey()+" : "+entry.getValue());
```

```
Test Input Expected Got
     1
                ONE : 1
                          ONE : 1
          ONE
                TWO : 2
                           TWO : 2
                 THREE : 3 THREE : 3
          TWO
          2 SIX: 6
THREE ONE: 1
                          SIX: 6
                          ONE : 1
                TWO : 2
                SEVEN : 7 SEVEN : 7
                THREE: 3 THREE: 3
                true
                          true
                true
                          true
Passed all tests! 🗸
```

inish review