# 

# 

# 

# 

# 

# LAB 11

# SET, MAP

EXERCISE 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.

Answer:

import java.util.HashSet;

import java.util.Scanner;

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++)

numbers.add(sc.nextInt());

int skey=sc.nextInt();

// Show which numbers between 1 and 10 are in the set

if(numbers.contains(skey))

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

else {

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

}

}

}

|  | **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. |  |

EXERCISE 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

Answer:

//Intersection

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

Set<String> h1=new HashSet<>();

Set<String> h2=new HashSet<>();

int n1=obj.nextInt();

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

{

h1.add(obj.next());

}

int n2=obj.nextInt();

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

{

h2.add(obj.next());

}

h1.retainAll(h2);

for(String s:h1)

{

System.out.println(s);

}

obj.close();

}

}

|  | **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 |  |

EXERCISE 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

Answer:

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.get("TWO");

System.out.println(value);

//Checking whether key 'ONE' exist in map

System.out.println( map.containsKey("ONE") );

//Checking whether value '3' exist in map

System.out.println( map.containsValue(3) );

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

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

}

}

|  | **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 |  |