HARIRAG G

230701104

Object Oriented Programming Using Java

Week 11

1)





2)





3)



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();

sc.nextLine(); // Consume the newline character

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'

Integer value = map.get("TWO"); // Using Integer to handle possible null values

System.out.println(value != null ? value : "Key not found");

// Checking whether key 'ONE' exists in map

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

// Checking whether value '3' exists in map

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

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

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

sc.close();

}

}

