**Problem 1: Frequency Count of Elements**

**Problem Statement:** Write a Java program to count the frequency of each element in an array using a HashMap.

**Solution:**

java

Copy code

import java.util.HashMap;

import java.util.Map;

public class FrequencyCount {

public static void main(String[] args) {

int[] numbers = {1, 2, 2, 3, 3, 3, 4, 4, 4, 4};

// Using HashMap to count frequencies

HashMap<Integer, Integer> frequencyMap = new HashMap<>();

for (int num : numbers) {

frequencyMap.put(num, frequencyMap.getOrDefault(num, 0) + 1);

}

// Printing the frequency of each element

for (Map.Entry<Integer, Integer> entry : frequencyMap.entrySet()) {

System.out.println("Element " + entry.getKey() + " occurs " + entry.getValue() + " times.");

}

}

}

**Problem 2: Find the First Non-Repeated Character in a String**

**Problem Statement:** Write a Java program to find the first non-repeated character in a string using a HashMap.

**Solution:**

java

Copy code

import java.util.HashMap;

public class FirstNonRepeatedCharacter {

public static void main(String[] args) {

String str = "swiss";

// Using HashMap to store character counts

HashMap<Character, Integer> charCountMap = new HashMap<>();

for (char c : str.toCharArray()) {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

// Finding the first non-repeated character

for (char c : str.toCharArray()) {

if (charCountMap.get(c) == 1) {

System.out.println("First non-repeated character is: " + c);

return;

}

}

System.out.println("No non-repeated character found.");

}

}

**Problem 3: Group Anagrams**

**Problem Statement:** Write a Java program to group anagrams from a list of strings using a HashMap.

**Solution:**

java

Copy code

import java.util.\*;

public class GroupAnagrams {

public static void main(String[] args) {

String[] words = {"eat", "tea", "tan", "ate", "nat", "bat"};

// Using HashMap to group anagrams

HashMap<String, List<String>> anagramMap = new HashMap<>();

for (String word : words) {

char[] charArray = word.toCharArray();

Arrays.sort(charArray);

String sortedWord = new String(charArray);

if (!anagramMap.containsKey(sortedWord)) {

anagramMap.put(sortedWord, new ArrayList<>());

}

anagramMap.get(sortedWord).add(word);

}

// Printing grouped anagrams

for (Map.Entry<String, List<String>> entry : anagramMap.entrySet()) {

System.out.println("Anagram group: " + entry.getValue());

}

}

}

**Problem 4: Find Pairs with Given Sum**

**Problem Statement:** Write a Java program to find all pairs of integers in an array whose sum is equal to a given value using a HashMap.

**Solution:**

java

Copy code

import java.util.HashMap;

import java.util.Map;

public class PairsWithGivenSum {

public static void main(String[] args) {

int[] numbers = {1, 5, 7, -1, 5};

int targetSum = 6;

// Using HashMap to find pairs

HashMap<Integer, Integer> numCountMap = new HashMap<>();

for (int num : numbers) {

numCountMap.put(num, numCountMap.getOrDefault(num, 0) + 1);

}

// Finding pairs

for (int num : numbers) {

int complement = targetSum - num;

if (numCountMap.containsKey(complement)) {

int count = numCountMap.get(complement);

for (int i = 0; i < count; i++) {

System.out.println("Pair found: (" + num + ", " + complement + ")");

}

// To avoid printing the same pair again

numCountMap.put(num, 0);

numCountMap.put(complement, 0);

}

}

}

}

**Problem 5: Employee Information System**

**Problem Statement:** Write a Java program to create an employee information system using a HashMap where employee IDs are mapped to employee names. Implement methods to add, remove, and fetch employee information.

**Solution:**

java

Copy code

import java.util.HashMap;

public class EmployeeInfoSystem {

private HashMap<Integer, String> employeeMap;

public EmployeeInfoSystem() {

employeeMap = new HashMap<>();

}

// Method to add employee

public void addEmployee(int id, String name) {

employeeMap.put(id, name);

}

// Method to remove employee

public void removeEmployee(int id) {

employeeMap.remove(id);

}

// Method to get employee name by ID

public String getEmployee(int id) {

return employeeMap.getOrDefault(id, "Employee not found");

}

public static void main(String[] args) {

EmployeeInfoSystem eis = new EmployeeInfoSystem();

eis.addEmployee(101, "Alice");

eis.addEmployee(102, "Bob");

eis.addEmployee(103, "Charlie");

// Fetching employee information

System.out.println("Employee with ID 101: " + eis.getEmployee(101));

System.out.println("Employee with ID 102: " + eis.getEmployee(102));

// Removing an employee

eis.removeEmployee(102);

System.out.println("Employee with ID 102 after removal: " + eis.getEmployee(102));

}

}

These problems cover a range of operations and use cases for HashMap in Java. Practice them to strengthen your understanding and problem-solving skills with HashMap.

4o

give me 10 different problems on hashmaps

ChatGPT

Sure, here are 10 different problems involving HashMap in Java, each followed by a brief description of the task to be accomplished:

**Problem 1: Word Frequency in a Paragraph**

**Problem Statement:** Given a paragraph of text, write a Java program to count the frequency of each word using a HashMap.

**Problem 2: Find Duplicates in an Array**

**Problem Statement:** Write a Java program to find all duplicate elements in an array using a HashMap.

**Problem 3: Character Count in a String**

**Problem Statement:** Write a Java program to count the occurrences of each character in a string using a HashMap.

**Problem 4: Two Sum Problem**

**Problem Statement:** Given an array of integers, return indices of the two numbers such that they add up to a specific target using a HashMap.

**Problem 5: Group People by Age**

**Problem Statement:** Given a list of people with their ages, write a Java program to group people by their age using a HashMap.

**Problem 6: Find All Anagrams in a String**

**Problem Statement:** Write a Java program to find all anagram substrings of a given string using a HashMap.

**Problem 7: Check for Isomorphic Strings**

**Problem Statement:** Write a Java program to check if two strings are isomorphic using a HashMap.

**Problem 8: Employee Management System**

**Problem Statement:** Write a Java program to implement an employee management system using a HashMap, with functionalities to add, remove, update, and fetch employee details.

**Problem 9: LRU Cache Implementation**

**Problem Statement:** Implement a Least Recently Used (LRU) cache in Java using a HashMap and a doubly linked list.

**Problem 10: Sort Elements by Frequency**

**Problem Statement:** Given an array of integers, write a Java program to sort the elements by their frequency using a HashMap.

**Solutions**

**Problem 1: Word Frequency in a Paragraph**

**Solution:**

java

Copy code

import java.util.HashMap;

import java.util.Map;

public class WordFrequency {

public static void main(String[] args) {

String paragraph = "This is a test. This test is only a test.";

// Removing punctuation and converting to lowercase

String[] words = paragraph.replaceAll("[^a-zA-Z ]", "").toLowerCase().split("\\s+");

// Using HashMap to count word frequencies

HashMap<String, Integer> wordCountMap = new HashMap<>();

for (String word : words) {

wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);

}

// Printing the word frequencies

for (Map.Entry<String, Integer> entry : wordCountMap.entrySet()) {

System.out.println("Word: " + entry.getKey() + ", Frequency: " + entry.getValue());

}

}

}

**Problem 2: Find Duplicates in an Array**

**Solution:**

java

Copy code

import java.util.HashMap;

import java.util.Map;

public class FindDuplicates {

public static void main(String[] args) {

int[] numbers = {1, 2, 2, 3, 4, 4, 5};

// Using HashMap to find duplicates

HashMap<Integer, Integer> countMap = new HashMap<>();

for (int num : numbers) {

countMap.put(num, countMap.getOrDefault(num, 0) + 1);

}

// Printing duplicates

System.out.println("Duplicate elements:");

for (Map.Entry<Integer, Integer> entry : countMap.entrySet()) {

if (entry.getValue() > 1) {

System.out.println(entry.getKey() + " occurs " + entry.getValue() + " times");

}

}

}

}

**Problem 3: Character Count in a String**

**Solution:**

java

Copy code

import java.util.HashMap;

public class CharacterCount {

public static void main(String[] args) {

String str = "programming";

// Using HashMap to count character occurrences

HashMap<Character, Integer> charCountMap = new HashMap<>();

for (char c : str.toCharArray()) {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

// Printing character counts

System.out.println("Character counts:");

for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {

System.out.println(entry.getKey() + ": " + entry.getValue());

}

}

}

**Problem 4: Two Sum Problem**

**Solution:**

java

Copy code

import java.util.HashMap;

public class TwoSum {

public static int[] findTwoSum(int[] nums, int target) {

HashMap<Integer, Integer> numMap = new HashMap<>();

for (int i = 0; i < nums.length; i++) {

int complement = target - nums[i];

if (numMap.containsKey(complement)) {

return new int[] { numMap.get(complement), i };

}

numMap.put(nums[i], i);

}

throw new IllegalArgumentException("No two sum solution");

}

public static void main(String[] args) {

int[] nums = {2, 7, 11, 15};

int target = 9;

int[] result = findTwoSum(nums, target);

System.out.println("Indices of the two numbers: " + result[0] + " and " + result[1]);

}

}

**Problem 5: Group People by Age**

**Solution:**

java

Copy code

import java.util.HashMap;

import java.util.List;

import java.util.ArrayList;

import java.util.Map;

class Person {

String name;

int age;

Person(String name, int age) {

this.name = name;

this.age = age;

}

}

public class GroupPeopleByAge {

public static void main(String[] args) {

List<Person> people = new ArrayList<>();

people.add(new Person("Alice", 30));

people.add(new Person("Bob", 25));

people.add(new Person("Charlie", 30));

people.add(new Person("David", 25));

people.add(new Person("Eve", 35));

// Using HashMap to group people by age

HashMap<Integer, List<String>> ageGroupMap = new HashMap<>();

for (Person person : people) {

ageGroupMap.computeIfAbsent(person.age, k -> new ArrayList<>()).add(person.name);

}

// Printing groups of people by age

for (Map.Entry<Integer, List<String>> entry : ageGroupMap.entrySet()) {

System.out.println("Age " + entry.getKey() + ": " + entry.getValue());

}

}

}

**Problem 6: Find All Anagrams in a String**

**Solution:**

java

Copy code

import java.util.\*;

public class FindAllAnagrams {

public static List<Integer> findAnagrams(String s, String p) {

List<Integer> result = new ArrayList<>();

if (s == null || p == null || s.length() < p.length()) return result;

HashMap<Character, Integer> pMap = new HashMap<>();

for (char c : p.toCharArray()) {

pMap.put(c, pMap.getOrDefault(c, 0) + 1);

}

int left = 0, right = 0, count = pMap.size();

while (right < s.length()) {

char rightChar = s.charAt(right);

if (pMap.containsKey(rightChar)) {

pMap.put(rightChar, pMap.get(rightChar) - 1);

if (pMap.get(rightChar) == 0) count--;

}

right++;

while (count == 0) {

if (right - left == p.length()) result.add(left);

char leftChar = s.charAt(left);

if (pMap.containsKey(leftChar)) {

pMap.put(leftChar, pMap.get(leftChar) + 1);

if (pMap.get(leftChar) > 0) count++;

}

left++;

}

}

return result;

}

public static void main(String[] args) {

String s = "cbaebabacd";

String p = "abc";

List<Integer> result = findAnagrams(s, p);

System.out.println("Anagram indices: " + result);

}

}

**Problem 7: Check for Isomorphic Strings**

**Solution:**

java

Copy code

import java.util.HashMap;

public class IsomorphicStrings {

public static boolean isIsomorphic(String s, String t) {

if (s.length() != t.length()) return false;

HashMap<Character, Character> mapST = new HashMap<>();

HashMap<Character, Character> mapTS = new HashMap<>();

for (int i = 0; i < s.length(); i++) {

char c1 = s.charAt(i);

char c2 = t.charAt(i);

if (mapST.containsKey(c1)) {

if (mapST.get(c1) != c2) return false;

} else {

mapST.put(c1, c2);

}

if (mapTS.containsKey(c2)) {

if (mapTS.get(c2) != c1) return false;

} else {

mapTS.put(c2, c1);

}

}

return true;

4o