**每日作业卷答案**

**JavaSE第09天**

# 关卡1

## 训练案例1

### 训练知识点

1.Map特点

### 训练描述

请简述Map 的特点

### 操作步骤答案

Map的特点:

1.Map每个元素由键与值两部分组成

2.Map键不能重复,每个键对应一个值

3.键和值可以为null

## 训练案例2

### 训练知识点

1.Map接口中的常用方法

### 训练描述

练习Map集合的常用方法: put, get, remove

### 操作步骤答案

**public** **class** Test02 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, String> hm = **new** HashMap<String, String>();

// 2.使用put添加元素

hm.put("黄晓明", "Baby");

hm.put("邓超", "孙俪");

hm.put("李晨", "范冰冰");

hm.put("大黑牛", "范冰冰");

// 3.使用put修改元素

String v1 = hm.put("李晨", "白百合");

// 4.使用get获取元素

String string = hm.get("大黑牛");

// 5.使用remove删除元素

String v2 = hm.remove("大黑牛");

System.***out***.println(v2);

// 6.打印集合中的元素

System.***out***.println(hm);

}

}

## 训练案例3

### 训练知识点

1.Map接口中的方法

### 训练描述

往一个Map集合中添加若干元素.获取Map中的所有key, 并使用增强for和迭代器遍历输出每个key

### 操作步骤答案

**public** **class** Test03 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, String> hm = **new** HashMap<String, String>();

// 2.使用put添加元素

hm.put("黄晓明", "Baby");

hm.put("邓超", "孙俪");

hm.put("李晨", "范冰冰");

hm.put("大黑牛", "范冰冰");

// 3.使用Map的keySet方法获取到所有的key

Set<String> keySet = hm.keySet();

// 4.使用增强for获取每个key

**for** (String key : keySet) {

System.***out***.println(key);

}

System.***out***.println("----------------");

// 5.获取keySet的迭代器

Iterator<String> itr = keySet.iterator();

// 6.循环判断迭代器中是否有下一个元素

**while** (itr.hasNext()) {

// 7.使用迭代器获取每个key

System.***out***.println(itr.next());

}

}

}

## 训练案例4

### 训练知识点

1.Map接口中的方法

### 训练描述

往一个Map集合中添加若干元素.获取Map中的所有value, 并使用增强for和迭代器遍历输出每个value

### 操作步骤答案

**public** **class** Test04 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, String> hm = **new** HashMap<String, String>();

// 2.使用put添加元素

hm.put("黄晓明", "Baby");

hm.put("邓超", "孙俪");

hm.put("李晨", "范冰冰");

hm.put("大黑牛", "范冰冰");

// 3.使用Map的values方法获取到所有的value

Collection<String> values = hm.values();

// 4.使用增强for获取每个value

**for** (String value : values) {

System.***out***.println(value);

}

System.***out***.println("----------------");

// 5.使用迭代器获取每个value

Iterator<String> itr = values.iterator();

**while** (itr.hasNext()) {

System.***out***.println(itr.next());

}

}

}

## 训练案例5

### 训练知识点

1.Map遍历

### 训练描述

Map集合中包含5对元素: "邓超"->"孙俪", "李晨"->"范冰冰", "刘德华"->"柳岩", “黄晓明”->” Baby”, “谢霆锋”->”张柏芝”.使用keySet方式迭代出Map集合中的元素

### 操作步骤答案

**public** **class** Test05 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, String> hm = **new** HashMap<String, String>();

// 2.使用put方法添加元素

hm.put("邓超", "孙俪");

hm.put("李晨", "范冰冰");

hm.put("刘德华", "柳岩");

hm.put("黄晓明", "Baby");

hm.put("谢霆锋", "张柏芝");

// 3.使用keySet方法获取所有的键

Set<String> keySet = hm.keySet();

// 4.获取到keySet的迭代器

Iterator<String> it = keySet.iterator();

// 5.循环判断迭代器是否有下一个元素

**while** (it.hasNext()) {

// 6.使用迭代器next方法获取到一个键

String key = it.next();

// 7.通过一个键找到一个值

String value = hm.get(key);

// 8.输出键和值

System.***out***.println(key + " - " + value);

}

}

}

## 训练案例6

### 训练知识点

1.Map遍历

### 训练描述

Map集合中包含5对元素: "邓超"->"孙俪", "李晨"->"范冰冰", "刘德华"->"柳岩", “黄晓明”->” Baby”, “谢霆锋”->”张柏芝”.使用Entry方式迭代出Map集合中的元素

### 操作步骤答案

**public** **class** Test06 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, String> hm = **new** HashMap<String, String>();

// 2.使用put方法添加元素

hm.put("邓超", "孙俪");

hm.put("李晨", "范冰冰");

hm.put("刘德华", "柳岩");

hm.put("黄晓明", "Baby");

hm.put("谢霆锋", "张柏芝");

// 3.使用entrySet方法获取到所有的键值对关系

Set<Entry<String, String>> entrySet = hm.entrySet();

// 4.获取entrySet的迭代器

Iterator<Entry<String, String>> it1 = entrySet.iterator();

// 5.使用迭代器查看是否有下一个元素

**while** (it1.hasNext()) {

// 6.使用迭代器拿到下一个元素(Entry类型)

Map.Entry<String, String> entry = it1.next();

// 7.通过一个Entry<K,V>获取到key和value

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

}

}

}

## 训练案例7

### 训练知识点

1.HashMap存储键是String值是自定义对象

### 训练描述

每位学生（姓名，年龄）都有自己的家庭住址。那么，既然有对应关系，则将学生对象和家庭住址存储到Map集合中。**家庭住址**作为**键**, **学生**作为**值**。并使用keySet和entrySet方式遍历Map集合

### 操作步骤答案

学生类:

// 1.定义学生类.包含姓名和年龄属性

**public** **class** Student {

**private** String name;

**private** **int** age;

**public** Student(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

}

测试类:

**public** **class** Test07 {

**public** **static** **void** main(String[] args) {

// 2.创建HashMapkey保存学生家庭住址,value是学生对象

HashMap<String, Student> hm = **new** HashMap<String, Student>();

// 3.添加一批学生到HashMap中

Student p1 = **new** Student("马云", 46);

Student p2 = **new** Student("马化腾", 42);

Student p3 = **new** Student("马冬梅", 22);

Student p4 = **new** Student("马赛克", 18);

hm.put("杭州", p1);

hm.put("深圳", p2);

hm.put("北京", p3);

hm.put("东京", p4);

// 4.使用keySet方式遍历Map

**for** (String key : hm.keySet()) {

// 5.通过key获取value

Student p = hm.get(key);

System.***out***.println(key + " - " + p.getName() + "," + p.getAge());

}

System.***out***.println("----------------------");

// 6.使用entrySet方式遍历Map

Set<Entry<String, Student>> entrySet = hm.entrySet();

**for** (Entry<String, Student> entry : entrySet) {

Student p = entry.getValue();

System.***out***.println(entry.getKey() + " - " + p.getName() + "," + p.getAge());

}

}

}

## 训练案例8

### 训练知识点

1.HashMap存储键是自定义对象值是String

### 训练描述

每位学生（姓名，年龄）都有自己的家庭住址。那么，既然有对应关系，则将学生对象和家庭住址存储到map集合中。**学生**作为**键**, **家庭住址**作为**值,** 并使用keySet和entrySet方式遍历Map集合.注意，学生**姓名**相同并且**年龄**相同视为同一名学生

### 操作步骤答案

学生类:

// 1.定义学生类.包含姓名和年龄属性,重写hashCode和equals方法

**public** **class** Student {

**private** String name;

**private** **int** age;

**public** Student(String name, **int** age) {

**this**.name = name;

**this**.age = age;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

@Override

**public** **int** hashCode() {

**final** **int** prime = 31;

**int** result = 1;

result = prime \* result + age;

result = prime \* result + ((name == **null**) ? 0 : name.hashCode());

**return** result;

}

@Override

**public** **boolean** equals(Object obj) {

**if** (**this** == obj)

**return** **true**;

**if** (obj == **null**)

**return** **false**;

**if** (getClass() != obj.getClass())

**return** **false**;

Student other = (Student) obj;

**if** (age != other.age)

**return** **false**;

**if** (name == **null**) {

**if** (other.name != **null**)

**return** **false**;

} **else** **if** (!name.equals(other.name))

**return** **false**;

**return** **true**;

}

}

测试类:

**public** **class** Test08 {

**public** **static** **void** main(String[] args) {

// 2.创建HashMapkey保存学生对象,value是学生家庭住址

HashMap<Student, String> hm = **new** HashMap<>();

// 3.添加一批学生到HashMap中

Student p1 = **new** Student("马云", 46);

Student p2 = **new** Student("马化腾", 42);

Student p3 = **new** Student("马冬梅", 22);

Student p4 = **new** Student("马赛克", 18);

Student p5 = **new** Student("马赛克", 18);

hm.put(p1, "杭州");

hm.put(p2, "深圳");

hm.put(p3, "北京");

hm.put(p4, "东京");

hm.put(p5, "西安");

// 4.使用keySet方式遍历Map

Set<Student> keySet = hm.keySet();

**for** (Student p : keySet) {

// 根据key获取value

String value = hm.get(p);

System.***out***.println(p.getName() + "," + p.getAge() + " - " + value);

}

System.***out***.println("-------------");

// 5.使用entrySet方式遍历Map

Set<Entry<Student, String>> entrySet = hm.entrySet();

**for** (Entry<Student, String> entry : entrySet) {

Student key = entry.getKey();

String value = entry.getValue();

System.***out***.println(key.getName() + "," + key.getAge() + " - " + value);

}

}

}

## 训练案例9

### 训练知识点

1.可变参数

### 训练描述

定义一个方法add接收0个及以上的int类型数据,这个方法的作用是将传入的int类型的值进行相加,并返回相加后的结果

### 操作步骤答案

**public** **class** Test09 {

**public** **static** **void** main(String[] args) {

// 6.调用add方法传入任意个int类型的参数

*add*(1);

*add*(1, 2);

*add*(1, 3, 5);

}

// 1.定义add方法.参数是可变参数(int... arr)

**public** **static** **int** add(**int**... arr) {

// 2.在add方法定义求和变量sum

**int** sum = 0;

// 3.使用增强for获取每个参数

**for** (**int** i : arr) {

// 4.将每个参数都加到求和变量sum中

sum += i;

}

// 5.返回sum

**return** sum;

}

}

## 训练案例10

### 训练知识点

1.Collections工具类

### 训练描述

ArrayList集合中有如下内容: {33,11,77,55},使用Collections.sort()对ArrayList集合中的数据进行排序,并打印出排序后的结果

### 操作步骤答案

**public** **class** Test10 {

**public** **static** **void** main(String[] args) {

// 1.创建ArrayList

ArrayList<Integer> arr = **new** ArrayList<Integer>();

// 2.使用add方法添加{33,11,77,55}四个元素

arr.add(33);

arr.add(11);

arr.add(77);

arr.add(55);

// 3.调用Collections的sort方法,对集合排序

Collections.*sort*(arr);

// 4.使用增强for遍历ArrayList集合

**for** (Integer integer : arr) {

System.***out***.println(integer);

}

}

}

## 训练案例11

### 训练知识点

1.Collections工具类

### 训练描述

ArrayList集合中有如下内容: {11,33,55,77},使用Collections.shuffle ()对ArrayList集合中的数据进行随机打乱顺序,并打印出排序后的结果

### 操作步骤答案

**public** **class** Test11 {

**public** **static** **void** main(String[] args) {

// 1.创建ArrayList

ArrayList<Integer> arr = **new** ArrayList<Integer>();

// 2.使用add方法添加{11,33,55,77}四个元素

arr.add(11);

arr.add(33);

arr.add(55);

arr.add(77);

// 3.调用Collections的shuffle方法,对集合中的元素打乱顺序

Collections.*sort*(arr);

// 4.使用增强for遍历ArrayList集合

**for** (Integer integer : arr) {

System.***out***.println(integer);

}

}

}

## 训练案例12

### 训练知识点

1.Collections工具类

### 训练描述

ArrayList集合中有如下内容: {11,33,55,77},使用Collections.reverse()对ArrayList集合中的数据进行反转,使用增强for遍历ArrayList集合

### 操作步骤答案

**public** **class** Test12 {

**public** **static** **void** main(String[] args) {

// 1.创建ArrayList

ArrayList<Integer> arr = **new** ArrayList<Integer>();

// 2.使用add方法添加{11,33,55,77}四个元素

arr.add(11);

arr.add(33);

arr.add(55);

arr.add(77);

// 3.调用Collections的reverse方法,对ArrayList集合中的数据进行反转

Collections.*reverse*(arr);

// 4.使用增强for遍历ArrayList集合

**for** (Integer integer : arr) {

System.***out***.println(integer);

}

}

}

## 训练案例13

### 训练知识点

1.Collections工具类

### 训练描述

ArrayList集合中有如下内容: {11,33,55,77},使用Collections.binarySearch ()查询33元素在ArrayList集合中的索引是多少

### 操作步骤答案

**public** **class** Test13 {

**public** **static** **void** main(String[] args) {

// 1.创建ArrayList

ArrayList<Integer> arr = **new** ArrayList<Integer>();

// 2.使用add方法添加{11,33,55,77}四个元素

arr.add(11);

arr.add(33);

arr.add(55);

arr.add(77);

// 3.调用Collections的binarySearch方法,对ArrayList集合中的数据进行反转

**int** index = Collections.*binarySearch*(arr, 33);

// 4.输出查找到的索引

System.***out***.println(index);

}

}

# 关卡2

## 训练案例1

### 训练知识点

1.Map集合

### 训练描述

使用键盘录入5个学生信息,录入格式为:”姓名,年龄”.将这5个学生信息存入Map集合中,key是姓名,value是年龄

### 操作步骤答案

**public** **class** Test01 {

**public** **static** **void** main(String[] args) {

// 1.创建HashMap

HashMap<String, Integer> hm = **new** HashMap<>();

// 2.创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// 3.循环判断只要HashMap数量是否小于5

**while** (hm.size() < 5) {

// 4.HashMap数量小于5个就让用户接着输入

System.***out***.println("请输入姓名和年龄,格式为:姓名,年龄");

String line = sc.nextLine();

// 5.使用,分割用户输入的字符串

String[] strings = line.split(",");

// 6.将分割后的索引为0的字符串作为key,索引为1的字符串作为value,放入HashMap中

hm.put(strings[0], Integer.*valueOf*(strings[1]));

}

// 7.输出HashMap中的数据

System.***out***.println(hm);

}

}

## 训练案例2

### 训练知识点

1.可变参数

### 训练描述

定义一个方法add接收0个及以上的String类型数据,这个方法的作用是将传入的String类型的参数放入一个ArrayList中,并返回这个ArrayList集合

### 操作步骤答案

**public** **class** Test02 {

**public** **static** **void** main(String[] args) {

// 6.调用add方法传入任意个字符串

ArrayList<String> list = *add*("a", "b", "d");

// 7.使用增强for遍历返回的ArrayList,打印每个元素

**for** (String str : list) {

System.***out***.println(str);

}

}

// 1.定义add方法,参数是(String ... str),返回值是ArrayList<String>

**public** **static** ArrayList<String> add(String ... str) {

// 2.在add方法中创建ArrayList用于存放传入的字符串

ArrayList<String> array = **new** ArrayList<String>();

// 3.使用增强for循环获取每个参数

**for** (String s : str) {

// 4.将每个参数都添加到ArrayList中

array.add(s);

}

// 5.返回array

**return** array;

}

}

## 训练案例3

### 训练知识点

1.集合转数组

### 训练描述

ArrayList集合中有如下内容: {11,33,55,77}.使用2种方式将集合中的数据转成数组

### 操作步骤答案

**public** **class** Test03 {

**public** **static** **void** main(String[] args) {

// 1.定义ArrayList

ArrayList<Integer> arr = **new** ArrayList<Integer>();

// 2.使用add方法添加元素{11,33,55,77}

arr.add(10);

arr.add(30);

arr.add(20);

arr.add(50);

// 3.使用方式1转数组.toArray()

Object[] array = arr.toArray();

// 4.使用方式2转数组.准备空数组

Integer[] array2 = **new** Integer[arr.size()];

arr.toArray(array2);

// 5.打印数组

System.***out***.println(Arrays.*toString*(array));

System.***out***.println(Arrays.*toString*(array2));

}

}

## 训练案例4

### 训练知识点

1.Map集合

### 训练描述

研发部门有5个人，信息如下：（姓名-工资）【柳岩=2100, 张亮=1700, 诸葛亮=1800, 灭绝师太=2600, 东方不败=3800】,将以上员工的相关信息存放在适合的集合中,给柳岩涨工资300,迭代出每个元素的内容输出到控制台

### 操作步骤答案

**public** **class** Test04 {

**public** **static** **void** main(String[] args) {

// 1.定义HashMap,姓名作为key,工资作为value

HashMap<String, Integer> map = **new** HashMap<String,Integer>();

// 2.使用put方法添加需要的元素

map.put("柳岩", 2100);

map.put("张亮", 1700);

map.put("诸葛亮", 1800);

map.put("灭绝师太", 2600);

map.put("东方不败", 3800);

// 3.获取到柳岩的工资

**int** salary = map.get("柳岩");

// 4.修改柳岩的工资为当前工资加上300

map.put("柳岩", salary + 300);

// 5.使用增强for+keySet迭代出每个员工的工资

Set<String> keySet = map.keySet();

**for** (String key : keySet) {

System.***out***.println(key + " -- " + map.get(key));

}

}

}

## 训练案例5

### 训练知识点

1.Map集合

### 训练描述

现在有一个map集合如下:{1="张三丰",2="周芷若",3="汪峰",4="灭绝师太"}

要求:

1.将map集合中的所有信息显示到控制台上

2.向该map集合中插入一个编码为5姓名为李晓红的信息

3.移除该map中的编号为1的信息

4.将map集合中编号为2的姓名信息修改为"周林"

### 操作步骤答案

**public** **class** Test05 {

**public** **static** **void** main(String[] args) {

// 1.定义HashMap,编号作为key,姓名作为value

Map<Integer,String> map = **new** HashMap<Integer, String>();

// 2.使用put方法添加元素

map.put(1, "张三丰");

map.put(2, "周芷若");

map.put(3, "汪峰");

map.put(4, "灭绝师太");

// 3.使用keySet+增强for迭代map中的元素,并打印

Set<Integer> keySet = map.keySet();

**for** (Integer key : keySet) {

String value = map.get(key);

System.***out***.println(key + " -- " + value);

}

// 4.使用put向该map集合中插入一个编码为5姓名为李晓红的信息

map.put(5, "李晓红");

// 5.使用remove移除该map中的编号为1的信息

map.remove(1);

// 6.使用put将map集合中编号为2的姓名信息修改为"周林"

map.put(2, "周林");

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

}

}

## 训练案例6

### 训练知识点

1.Map集合

### 训练描述

利用键盘录入，输入一个字符串,统计该字符串中各个字符的数量

如用户输入字符串:"hello world java",程序输出结果：h(1)e(1)l(3)o(2) (2)w(1)r(1)d(1)j(1)a(2)v(1)

注：输出结果不要求顺序一致

### 操作步骤答案

**public** **class** Test06 {

**public** **static** **void** main(String[] args) {

// 1.创建键盘输入对象

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一行字符串:");

// 2.使用nextLine从键盘中读取一行内容

String line = sc.nextLine();

// 3.将字符串转换成字符数组

**char**[] charArray = line.toCharArray();

// 4.创建map集合，用来存放字符：key是字符 value是字符出现的个数

Map<Character,Integer> map = **new** HashMap<>();

// 5.遍历字符数组

**for** (**char** c : charArray) {

// 6.拿到每个字符,去map中查看是否有保存对应的次数

Integer count = map.get(c);

**if** (count == **null**) {

// 7.如果这个字符没有保存次数,就设置为1次

map.put(c, 1);

} **else** {

// 8.如果这个字符有保存次数,就设置为原来的次数+1次

map.put(c, count + 1);

}

}

// 9.创建字符串缓冲区对象

StringBuilder sb = **new** StringBuilder();

// 10.按题目要求拼接字符串

**for**(Entry<Character,Integer> entry : map.entrySet()) {

sb.append(entry.getKey() + "(" + entry.getValue() + ")");

}

// 11.输出结果

System.***out***.println(sb);

}

}

## 训练案例7

### 训练知识点

1.Map集合

### 训练描述

键盘录入一个字符串,分别统计出其中英文字母、空格、数字和其它字符的数量,输出结果:”其他=1, 空格=2, 字母=12, 数字=6”

### 操作步骤答案

**public** **class** Test07 {

**public** **static** **void** main(String[] args) {

// 6.创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

// 7.使用nextLine获取用户录入的字符串

System.***out***.println("请录入一串字符串");

String line = sc.nextLine();

// 8.定义一个Map用于统计,字母,空格,数字和其他字符的数量,key是字符value是字符出现的次数

HashMap<String, Integer> map = **new** HashMap<String, Integer>();

// 9.将输入的字符串转成字符

**char**[] charArray = line.toCharArray();

// 10.遍历字符串数组中的每一个字符

**for** (**char** ch : charArray) {

**if** ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z')) {

// 11.如果ch是一个字符,调用countingKey方法传入(map, "字母")作为参数

*countingKey*(map, "字母");

} **else** **if** (ch == ' ') {

// 12.如果字符是一个空格,调用countingKey方法传入(map, "空格")作为参数

*countingKey*(map, "空格");

} **else** **if** (ch >= '0' && ch <= '9') {

// 13.如果字符是一个数字,调用countingKey方法传入(map, "数字")作为参数

*countingKey*(map, "数字");

} **else** {

// 14.如果字符是一个其他,调用countingKey方法传入(map, "其他")作为参数

*countingKey*(map, "其他");

}

}

// 15.打印统计结果

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

}

// 1.定义countingKey(HashMap<String, Integer> map, String key)方法.用于统计key出现的次数.调用一次,让key所对应的次数+1.

**private** **static** **void** countingKey(HashMap<String, Integer> map, String key) {

// 2.在countingKey方法中.使用get方法从map中取出key的值

Integer num = map.get(key);

// 3.如果为空,说明key第一次出现

**if** (num == **null**) {

// 4.放入key,把次数设置为1

map.put(key, 1);

} **else** {

// 5.如果之前,已经有key了,就把key的次数+1

map.put(key, num + 1);

}

}

}

## 训练案例8

### 训练知识点

1.Map集合

### 训练描述

定义一个List集合,泛型为String类型。统计整个集合中所有字符（注意，不是字符串）的个数。例如:集合中有”abc”、”bcd”两个元素,程序最终输出结果为:“a = 1,b = 2,c = 2,d = 1”

### 操作步骤答案

**public** **class** Test08 {

**public** **static** **void** main(String[] args) {

// 1.定义ArrayList存放元素

ArrayList<String> arr = **new** ArrayList<String>();

// 2.使用add方法添加需要的元素

arr.add("abc");

arr.add("bcd");

// 3.定义HashMap,key是字符,value是字符对应的次数

HashMap<Character, Integer> hm = **new** HashMap<>();

// 4.使用增强for获取ArrayList中的每个字符串

**for** (String str : arr) {

// 5.将每个字符串转成字符数组

**char**[] charArray = str.toCharArray();

// 6.使用增强for遍历字符数组

**for** (**char** ch : charArray) {

// 7.获取拿到每个字符,使用字符去HashMap中查找次数

Integer num = hm.get(ch);

// 8.如果为空,说明该字符第一次出现

**if** (num == **null**) {

// 9.放入字符作为key,把次数设置为1

hm.put(ch, 1);

} **else** {

// 10.如果之前,已经有字符了,就把字符的次数+1

hm.put(ch, num + 1);

}

}

}

// 11.输出map中的内容

System.***out***.println(hm);

}

}

## 训练案例9

### 训练知识点

1.Map集合

### 训练描述

键盘录入语句英语,统计每个单词出现的次数,有如下字符串"If you want to change your fate I think you must come to the dark horse to learn java"(用空格间隔)

打印格式：

to=3

think=1

you=2

…

### 操作步骤答案

**public** **class** Test09 {

**public** **static** **void** main(String[] args) {

// 1.创建键盘输入对象

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一行字符串:");

// 2.使用nextLine从键盘中读取一行内容

String line = sc.nextLine();

// 3.定义HashMap,key是单词,value是字符对应的次数

HashMap<String, Integer> hm = **new** HashMap<>();

// 4.使用空格分割字符串

String[] strings = line.split(" ");

// 5.遍历分割好的每个字符串

**for** (String str : strings) {

// 6.获取拿到每个字符串,使用字符去HashMap中查找次数

Integer num = hm.get(str);

// 7.如果为空,说明该字符串第一次出现

**if** (num == **null**) {

// 8.放入字符串作为key,把次数设置为1

hm.put(str, 1);

} **else** {

// 9.如果之前,已经有字符串了,就把字符的次数+1

hm.put(str, num + 1);

}

}

System.***out***.println(hm);

}

}

## 训练案例10

### 训练知识点

1.Map集合

### 训练描述

通过键盘录入五个人的姓名,每当录入一个人的姓名,随机给出一个成绩[0, 100](包含0分,也包含100分).将姓名和成绩存储到Map集合中,姓名作为key,成绩作为value.统计五个学生的总分,最高分,最低分,平均分,并将总分,最高分,最低分,平均分,写入到score.txt文件中,格式为: 总分=xxx,最高分=xxx,最低分=xxx,平均分=xxx

### 操作步骤答案

**public** **class** Test10 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建键盘输入对象

Scanner sc = **new** Scanner(System.***in***);

// 2.创建随机数对象

Random ran = **new** Random();

// 3.创建HashMap集合key存放姓名,value存放成绩

HashMap<String, Integer> hm = **new** HashMap<>();

// 4.循环判断HashMap中数量是否小于5

**while** (hm.size() < 5) {

System.***out***.println("请输入一个学生的姓名:");

// 5.如果HashMap中数量小于5,让用户再输入一个姓名

String name = sc.next();

// 6.使用随机数生成一个成绩

**int** score = ran.nextInt(101);

// 7.将姓名和分数添加到HashMap中

hm.put(name, score);

}

// 8.使用迭代器获取一个键

String k = hm.keySet().iterator().next();

// 9.通过键找到对应的分数,作为默认分数

Integer defaultScore = hm.get(k);

// 10.定义总分分变量

**int** totalScore = 0;

// 11.定义最高分变量=默认分数

**int** maxScore = defaultScore;

// 12.定义最低分变量=默认分数

**int** minScore = defaultScore;

// 13.使用keySet获取到所有的key

Set<String> keySet = hm.keySet();

// 14.使用增强for遍历所有的key

**for** (String key : keySet) {

// 15.根据key找到对应的value

Integer value = hm.get(key);

// 16.将这个分数添加到总分

totalScore += value;

// 17.如果最高分小于这个分数,让最高分=这个分数

**if** (maxScore < value) {

maxScore = value;

} **else** **if** (minScore > value) {

// 18.如果最低分大于这个分数,让最低分=这个分数

minScore = value;

}

}

// 19.计算平均分

**int** avg = totalScore / hm.size();

// 20.定义StringBuilder,用于拼接,总分,最高分,最低分,平均分字符串

StringBuilder sb = **new** StringBuilder();

sb.append("总分=").append(totalScore).append(",最高分=").append(maxScore).append(",最低分=").append(minScore).append(",平均分=").append(avg);

// 21.创建FileWriter字符输出流

FileWriter fw = **new** FileWriter("score.txt");

// 22.使用字符输出流将拼接好的字符串写到文件中

fw.write(sb.toString());

// 23.关闭流

fw.close();

}

}

## 训练案例11

### 训练知识点

1.Map集合

### 训练描述

有一个Map集合里面存储了学生的名字,年龄,内容如下:{赵四=21, 王二=17, 张三=21},1将里面的元素使用两种遍历方式打印的控制台,2将赵四的年龄改为18,3将元素存入到student.txt中.一行保存格式:姓名,年龄

### 操作步骤答案

**public** **class** Test11 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建HashMap集合key存放姓名,value存放年龄

HashMap<String, Integer> hm = **new** HashMap<>();

// 2.使用put添加对应的数据

hm.put("赵四", 21);

hm.put("王二", 17);

hm.put("张三", 21);

// 3.使用keySet遍历Map

Set<String> keySet = hm.keySet();

**for** (String key : keySet) {

Integer value = hm.get(key);

System.***out***.println(key + " -- " + value);

}

// 4.使用entrySet遍历Map

Set<Entry<String, Integer>> entrySet = hm.entrySet();

**for** (Entry<String, Integer> entry : entrySet) {

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

}

// 5.使用put修改赵四的年龄为18岁

hm.put("赵四", 18);

// 6.创建BufferedWriter字符输出流

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter("student.txt"));

// 7.使用keySet遍历Map

Set<String> keySet2 = hm.keySet();

**for** (String key : keySet2) {

// 8.根据key拿到对应的value

Integer value = hm.get(key);

// 9.使用字符输出流将拼接好的字符串写到文件中

bw.write(key + " -- " + value);

bw.newLine();

}

// 10.关闭流

bw.close();

}

}

## 训练案例12

### 训练知识点

1.Map集合

### 训练描述

使用键盘循环录入:书名,价格.创建一个Map集合,key存储书名,value存储书的价格. 录入以下5本书数据:{Java=32.5,C++=25,PHP=28.6,UI=21.3,Python=10.2} 要求;

1.删除集合中的C++这本书

2.Java的这本书涨价为38.5元

3.使用keySet和entrySet两种不同方式对集合遍历输出

### 操作步骤答案

**public** **class** Test12 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建HashMap集合key存放姓名,value存放年龄

HashMap<String, Double> hm = **new** HashMap<>();

// 2.创建键盘录入对象Scanner

Scanner sc = **new** Scanner(System.***in***);

// 3.循环判断HashSet的数量是否小于5

**while** (hm.size() < 5) {

// 4.如果HashSet的数量小于5,让用户输入一本书的信息

System.***out***.println("请录入一本书(书名,价格):");

String line = sc.nextLine();

// 5.使用,分割书名和价格

String[] strings = line.split(",");

// 6.将书名和价格添加到HashMap中

hm.put(strings[0], Double.*valueOf*(strings[1]));

}

// 7.使用remove方法删除集合中的C++这本书

hm.remove("C++");

// 8.使用put方法将Java的这本书涨价为38.5元

hm.put("Java", 38.5);

// 9.使用keySet遍历Map

Set<String> keySet = hm.keySet();

**for** (String key : keySet) {

Double value = hm.get(key);

System.***out***.println(key + " -- " + value);

}

// 10.使用entrySet遍历Map

Set<Entry<String, Double>> entrySet = hm.entrySet();

**for** (Entry<String, Double> entry : entrySet) {

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

}

}

}

## 训练案例13

### 训练知识点

1.Map集合

### 训练描述

Map中有{张三=23,李四=24,王五=15,赵六=36,孙七=17,田七=57,周八=35}这些元素key为姓名value为年龄.将周八的年龄修改为22,并将年龄小于30岁人的信息写入到perosn.txt中,person.txt每行数据格式为:姓名=年龄

### 操作步骤答案

**public** **class** Test13 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建HashMap集合key存放姓名,value存放年龄

HashMap<String, Integer> hm = **new** HashMap<>();

// 2.使用put方法添加对应的元素

hm.put("张三", 23);

hm.put("李四", 24);

hm.put("王五", 15);

hm.put("赵六", 36);

hm.put("孙七", 17);

hm.put("田七", 57);

hm.put("周八", 35);

// 3.使用put将周八的年龄修改为22

hm.put("周八", 22);

// 4.创建BufferedWriter字符输出流

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter("person.txt"));

// 5.使用keySet遍历HashMap集合

Set<String> keySet = hm.keySet();

**for** (String key : keySet) {

// 6.根据key获取到对应的value

Integer age = hm.get(key);

// 7.将key和value按格式拼接好写入到person.txt文件中

bw.write(key + "=" + age);

bw.newLine();

}

// 8.关闭流

bw.close();

}

}

## 训练案例14

### 训练知识点

1. 字符串分割
2. Map集合

### 训练描述

现有一串字符串”常州极客营,无锡极客营,武汉极客营,镇江极客营,苏州极客营”,要求使用程序统计出”极客营”在该字符串中出现的次数

### 操作步骤答案

/\*

现有一串字符串”常州极客营,无锡极客营,武汉极客营,镇江极客营,苏州极客营”,

要求使用程序统计出”极客营”在该字符串中出现的次数

\*/

**public** **class** Test14 {

**public** **static** **void** main(String[] args) {

// 1.定义极客营字符串

String str = "常州极客营,无锡极客营,武汉极客营,镇江极客营,苏州极客营";

// 2.创建HashMap保存极客营的次数

HashMap<String, Integer> gky = **new** HashMap<>();

// 3.使用put添加key"极客营"

gky.put("极客营", 0);

// 4.使用,分割字符串

String[] strings = str.split(",");

// 5.使用增强for遍历获取每个分割出来的字符串

**for** (String name : strings) {

// 6.如果分割后的字符串串包含"极客营",让HashSet中"极客营"的value+1

**if** (name.contains("极客营")) {

Integer count = gky.get("极客营");

gky.put("极客营", count + 1);

}

}

// 8.打印HashMap的内容

System.***out***.println(gky);

}

}

## 训练案例15

### 训练知识点

1.Map集合

### 训练描述

有2个数组,第一个数组内容为:[黑龙江省,浙江省,江西省,广东省,福建省],第二个数组为:[哈尔滨,杭州,南昌,广州,福州],将第一个数组元素作为key,第二个数组元素作为value存储到Map集合中.如{黑龙江省=哈尔滨, 浙江省=杭州, …}

### 操作步骤答案

**public** **class** Test15 {

**public** **static** **void** main(String[] args) {

// 1.定义第一个数组arr1

String[] arr1 = {"黑龙江省", "浙江省", "江西省", "广东省", "福建省"};

// 2.定义第二个数组arr2

String[] arr2 = {"哈尔滨", "杭州", "南昌", "广州", "福州"};

// 3.创建HashMap,key存放省,value存放市

HashMap<String, String> hm = **new** HashMap<>();

// 4.使用普通for循环遍历arr1

**for** (**int** i = 0; i < arr1.length; i++) {

// 5.根据索引到arr1中获取到省

String key = arr1[i];

// 6.根据索引到arr2中获取到省会城市

String value = arr2[i];

// 7.将省和省会城市添加到HashMap中

hm.put(key, value);

}

// 8.输出HashMap中的内容

System.***out***.println(hm);

}

}

# 关卡3

## 训练案例1

### 训练知识点

1.Map集合

### 训练描述

Map中有{张三=3,李四=4,王五=5,赵六=6,孙七=7,田七=7,周八=5}这些元素.将Map中value相同的元素从Map中去除(value为7和5是相同的,都要删除).再将Map中剩余元素的key存放到一个ArrayLis中,并打印到控制台上

### 操作步骤答案

**public** **class** Test01 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建HashMap集合key存放姓名,value存放年龄

HashMap<String, Integer> hm = **new** HashMap<>();

// 2.使用put方法添加对应的元素

hm.put("张三", 3);

hm.put("李四", 4);

hm.put("王五", 5);

hm.put("赵六", 6);

hm.put("孙七", 7);

hm.put("田七", 7);

hm.put("周八", 5);

// 3.定义一个ArrayList存放重复的值

ArrayList<Integer> repeatValues = **new** ArrayList<>();

// 4.拿到所有的键放到ArrayList中

ArrayList<Integer> values = **new** ArrayList<>(hm.values());

// 5.遍历所有的值

**for** (**int** i = 0; i < values.size(); i++) {

// 6.拿到一个值

Integer value = values.get(i);

// 7.和后面的那些值比较,看是否有重复的

**for** (**int** j = i + 1; j < values.size(); j++) {

Integer v2 = values.get(j);

// 8.和后面的比较相同,说明重复

**if** (value == v2) {

// 9.将重复的值放入repeatValues中

repeatValues.add(value);

}

}

}

System.***out***.println(repeatValues);

// 10.定义一个ArrayList存放值相同的键.因为只能拿键去删除

ArrayList<String> repeatKeys = **new** ArrayList<>();

// 11.使用keySet遍历HashSet

Set<String> keySet = hm.keySet();

**for** (String key : keySet) {

Integer value = hm.get(key);

// 12.如果值是重复的,那么将这个件放入repeatKeys中

**if** (repeatValues.contains(value)) {

repeatKeys.add(key);

}

}

System.***out***.println(repeatKeys);

// 13.遍历重复的键,通过键去HashMap中删除元素

**for** (String key : repeatKeys) {

hm.remove(key);

}

// 14.输出删除后的HashMap

System.***out***.println(hm);

}

}

## 训练案例2

### 训练知识点

1.HashMap嵌套HashMap

### 训练描述

极客营包含2个班级:Java基础班,Java就业班

Java基础班:

001 李晨

002 范冰冰

Java就业班:

001 马云

002 马化腾

使用HashMap嵌套HashMap方式完成,1使用keySet和增强for遍历出里面的元素,2使用entrySet和迭代器遍历出里面的元素

### 操作步骤答案

**public** **class** Test02 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.定义Java基础班的HashMap<String, String>

HashMap<String, String> JavaSEMap = **new** HashMap<String, String>();

// 2.存放2个学生到Java基础班的Map

JavaSEMap.put("001", "李晨");

JavaSEMap.put("002", "范冰冰");

// 3.定义Java就业班的HashMap<String, String>

HashMap<String, String> JavaEEMap = **new** HashMap<>();

// 4.存放2个学生到Java就业班的Map

JavaEEMap.put("001", "马云");

JavaEEMap.put("002", "马化腾");

// 5.定义极客营集合HashMap<String, HashMap<String, String>>

HashMap<String, HashMap<String, String>> czbkMap = **new** HashMap<>();

// 6.将Java基础班和Java就业班存放到极客营HashMap集合中

czbkMap.put("Java基础班", JavaSEMap);

czbkMap.put("Java就业班", JavaEEMap);

// 7.使用keySet和增强for遍历出里面的元素

// 7.1使用keySet获取极客营集合的所有key

Set<String> keySet = czbkMap.keySet(); // "Java基础班", "Java就业班"

// 7.2使用增强for获取极客营集合每个key

**for** (String czbkKey : keySet) {

System.***out***.println(czbkKey);

// 7.3通过key找value,value是Java基础班的HashMap和Java就业班的HashMap

HashMap<String, String> classMap = czbkMap.get(czbkKey);

// 7.4拿到班级的HashMap所有key

Set<String> classKeys = classMap.keySet();

// 7.5遍历班级的每个key(学号)

**for** (String key : classKeys) {

// 7.6根据班级的key(学号)去找value(姓名)

String value = classMap.get(key);

System.***out***.println("\t" + key + " -- " + value);

}

}

System.***out***.println("-------------------");

// 8.使用entrySet和迭代器遍历出里面的元素

// 8.1使用entrySet获取极客营集合的所有Entry

Set<Entry<String, HashMap<String, String>>> entrySet = czbkMap.entrySet();

// 8.2拿到极客营集合的所有Entry的迭代器

Iterator<Entry<String, HashMap<String, String>>> czbkItr = entrySet.iterator();

// 8.3循环判断极客营所有Entry的迭代器是否有下一个元素

**while** (czbkItr.hasNext()) {

// 8.4通过极客营所有Entry的迭代器获取一个Entry

Entry<String, HashMap<String, String>> czbkEntry = czbkItr.next();

// 8.5输出Entry的key(班级名称)

System.***out***.println(czbkEntry.getKey());

// 8.6获取Entry的value(某个班级Map集合)

HashMap<String, String> classMap = czbkEntry.getValue();

// 8.7获取班级Map的所有Entry

Set<Entry<String, String>> classEntry = classMap.entrySet();

// 8.8拿到班级Map的所有Entry的迭代器

Iterator<Entry<String, String>> classItr = classEntry.iterator();

// 8.9循环判班级Map的所有Entry的迭代器是否有下一个元素

**while** (classItr.hasNext()) {

// 8.10 使用班级Map的所有Entry的迭代器拿出一个Entry

Entry<String, String> entry = classItr.next();

// 8.11按格式输出entry的key和entry的value

System.***out***.println("\t" + entry.getKey() + " -- " + entry.getValue());

}

}

}

}

## 训练案例3

### 训练知识点

1.Map集合

### 训练描述

已知Map中保存如下信息:{“及时雨”=”宋江”, “玉麒麟”=”卢俊义”, “智多星”=”吴用”},其中键表示水浒中人物的外号,value表示人物的姓名.1往Map中添加 “入云龙”=”公孙胜”, ”豹子头”=”林冲”两位好汉.2删除“玉麒麟”=”卢俊义”,3将key为“智多星”的value修改为null,4.将“及时雨”=”宋江”,修改为”呼保义”=” 宋江”

### 操作步骤答案

**public** **class** Test03 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.定义HashMap<String, String>key是外号,value是姓名

HashMap<String, String> heroMap = **new** HashMap<String, String>();

// 2.使用put方法添加默认的任务外号和姓名

heroMap.put("及时雨", "宋江");

heroMap.put("玉麒麟", "卢俊义");

heroMap.put("智多星", "吴用");

// 3.使用put方法往Map中添加 “入云龙”=”公孙胜”, ”豹子头”=”林冲”两位好汉.

heroMap.put("入云龙", "公孙胜");

heroMap.put("豹子头", "林冲");

// 4.使用remove方法删除“玉麒麟”=”卢俊义”

heroMap.remove("玉麒麟");

// 5.使用put方法将key为“智多星”的value修改为null

heroMap.put("智多星", **null**);

// 6.使用remove方法删除“及时雨”=”宋江”

heroMap.remove("及时雨");

// 7.使用put添加”呼保义”=”宋江”

heroMap.put("呼保义", "宋江");

System.***out***.println(heroMap);

}

}

## 训练案例4

### 训练知识点

1. HashSet
2. Map

### 训练描述

有四种水果(苹果，香蕉，西瓜，橘子)

1、给每种水果设定一个商品号，商品号是8个0-9的随机数，商品号码不能重复, 最小值 "00000001", 最大值 "99999999"

2、根据商品号查询对应的商品。

如果查不到输出：“查无此商品”

如果能查到打印：”根据商品号：12345678，查询到对应的商品为：西瓜”

### 操作步骤答案

**public** **class** Test04 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.定义四种水果数组

String[] fruit = { "苹果", "香蕉", "西瓜", "橘子" };

// 2.创建随机数对象

Random ran = **new** Random();

// 3.创建HashSet保存4个不相同的商品号

HashSet<String> ids = **new** HashSet<>();

// 4.循环判断HashSet是否小于4.

**while** (ids.size() < 4) {

// 5.产生一个商品编号,是8个0-9的数

String id = "";

**for** (**int** i = 0; i < 8; i++) {

id += ran.nextInt(10);

}

// 6.添加到HashSet中

ids.add(id);

}

// 7.定义HashMap,存放商品id和商品名称

HashMap<String, String> hm = **new** HashMap<>();

// 8.遍历ids,拿到每个商品id,再去找一个对应的商品名称,存储到HashMap中

**int** index = 0;

**for** (String id : ids) {

// 找到一个对应的商品

String product = fruit[index];

hm.put(id, product);

index++;

}

System.***out***.println(hm);

// 9.让用户输入要查询的商品id

System.***out***.println("请输入要查询商品的id:");

Scanner sc = **new** Scanner(System.***in***);

String id = sc.next();

// 10.根据id去map中查找对应的商品

String product = hm.get(id);

**if** (product == **null**) {

// 11.没有找到商品打印:查无此商品

System.***out***.println("查无此商品");

} **else** {

// 12.找到商品打印对应商品的信息

System.***out***.println("根据商品号：" + id + "，查询到对应的商品为：" + product);

}

}

}

## 训练案例5

### 训练知识点

1. 斗地主洗牌发牌

### 训练描述

完成课上斗地主洗牌发牌案例(牌需要排序)

### 操作步骤答案

请参考课上代码