Java03

Task1

1. 整型： byte , short , int , long 字符型： char 浮点型： float , double 布尔型： boolean
2. 如下表

数据类型	占用的字节数	表示范围
byte	1	-128 ~ 127
short	2	-32,768 ~ 32,767
int	4	$-2^{31} \sim 2^{31}-1$
long	8	$-2^{63} \sim 2^{63}-1$

(使用markdown语法时导出的PDF竟然无法表示出指数？！)

3. 涉及**隐式类型转换**；b的值是52；因为`int b = a + c;` 这里发生隐式类型转换,在计算a+c时，char类型的变量 c 会自动提升为 int 类型，然后再与 int 类型的 a 进行加法运算。字符在计算机中是用ASCII码（或Unicode码）表示的，而字符'0'的ASCII码值是48（十进制）。
4. 输出结果为：

```
false
true
false
```

原因如下： 第一组比较： `new Integer()` 每次都会在堆内存中创建一个新的对象（即使数值相同）。`x` 和 `y` 是两个不同的对象引用，指向不同的内存地址。`==` 比较的是引用地址，不是值，所以结果为 `false`。 第二组比较： `Integer.valueOf()` 方法使用了整数缓存池。Java 为 `-128` 到 `127` 之间的整数预先创建了缓存对象。当调用 `valueOf()` 获取这个范围内的整数时，会返回缓存池中相同的对象引用。所以 `z` 和 `k` 指向同一个缓存对象，`==` 比较结果为 `true`。 第三组比较： 数值 `300` 超出了默认的缓存范围（-128 到 127）。对于超出缓存范围的数值，`Integer.valueOf()` 会创建新的对象（类似于 `new Integer()`）。所以 `m` 和 `p` 是两个不同的对象引用，`==` 比较结果为 `false`。

Task2

5. 结果如下：

```
14
6 8
```

计算过程：第1、2行分别将5、7赋值给a、b；第3行先是a、b分别自增，然后将a与b之和赋值给c

6. 1. int 和 float 在计算机中的存储方式: **int（整型）的存储 - 补码表示** int 通常占 4字节（32位），使用 二进制补码 形式存储：
   - 最高位（第31位）：符号位
     - `0` 表示正数
     - `1` 表示负数
   - 剩余31位：数值位

     float（单精度浮点数）的存储 - IEEE 754标准 float 占 4字节（32位），按以下方式存储：
   - 第31位：符号位（1表示负，0表示正）
   - 第30-23位（8位）：指数部分（Exponent），采用移码表示（偏移量127）
   - 第22-0位（23位）：尾数部分（Mantissa/Fraction），隐含最高位1

   2. 因为**整数溢出**，即计算结果超出了该数据类型能表示的范围。

Task3

v1: 处理原始数据并找到错得最多的科目

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
//import java.util.Scanner;

class MistakeStatistics{
    Map<String ,Integer> map = new HashMap<>();
    public void add(String subject ,int number){
        if(this.map.containsKey(subject)){
            int temp =this.map.get(subject);
            this.map.put(subject,number+temp);
        }
        else{
            this.map.put(subject,number);
        }
    }
}
public class java03_task3 {

    public static void main(String[] args) {
        MistakeStatistics table = new MistakeStatistics();
```

```
        //Scanner sc = new Scanner(System.in);
        String initialData =
"math:5,English:10,Chinese:10,math:20,English:10,chemistry:30,math:10,math:20";
        String[] part1 = initialData.split(",");//切割字符串，便于在HashMap中添加元
素
        for(int i = 0;i < part1.length;i++){
            String[] part2 = part1[i].split(":");
            table.add(part2[0],Integer.parseInt(part2[1]));
        }
        String maxKey = "";
        int maxValue=0;
        List<String> keys = new ArrayList<>(table.map.keySet());
        for(int i=0;i< keys.size();i++){
            String key = keys.get(i);
            int value = table.map.get(key);
            if(value>maxValue){
                maxValue = value;
                maxKey = key;
            }
            System.out.println(key + ":" + value);//打印所有科目的错题统计
        }
        System.out.println("-------------" + '\n');
        System.out.println(maxKey + ":" + maxValue);//输出错题数最多的科目及错题数

    }
}
```

v2：处理后续输入的数据

> 输入格式规定： 后续输入数据由多行组成，每行包括：科目名、冒号、错题数，且中间不包含空格。
> 当一行中仅包括“exit”或不满足输入格式时，输入的循环终止。

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;

class MistakeStatistics{
    Map<String ,Integer> map = new HashMap<>();
    public void add(String subject ,int number){
        if(this.map.containsKey(subject)){
            int temp =this.map.get(subject);
            this.map.put(subject,number+temp);
        }
        else{
            this.map.put(subject,number);
        }
    }
}
public class java03_task3 {
```

```
    public static void main(String[] args) {
        MistakeStatistics table = new MistakeStatistics();
        Scanner sc = new Scanner(System.in);
        String initialData =
"math:5,English:10,Chinese:10,math:20,English:10,chemistry:30,math:10,math:20";
        String[] part1 = initialData.split(",");//切割字符串，便于在HashMap中添加元
素
        for(int i = 0;i < part1.length;i++){
            String[] part2 = part1[i].split(":");
            table.add(part2[0],Integer.parseInt(part2[1]));
        }
        //-------------------------------------------------------------
        //v2新增
        while(true){
            String input = sc.nextLine().trim();
            String[] part = input.split(":");
            if(input.equals("exit") || part.length!=2){
                break;
            }
            table.add(part[0].trim(),Integer.parseInt(part[1].trim()));
        }
        //-------------------------------------------------------------
        String maxKey = "";
        int maxValue=0;
        List<String> keys = new ArrayList<>(table.map.keySet());
        for(int i=0;i< keys.size();i++){
            String key = keys.get(i);
            int value = table.map.get(key);
            if(value>maxValue){
                maxValue = value;
                maxKey = key;
            }
            System.out.println(key + ":" + value);//打印所有科目的错题统计
        }
        System.out.println("-------------" + '\n');
        System.out.println(maxKey + ":" + maxValue);//输出错题数最多的科目及错题数

    }
}
```

v3：支持多个学生输入数据 （折腾半天,选择使用静态方法 :clown_face: ）

> 输入格式规定： 后续输入数据由多行组成，每行包括：姓名、冒号、科目名、冒号、错题数，且中间不包含空格。 当一行中仅包括“exit”或不满足输入格式时，输入的循环终止。

```
import java.util.*;

public class java03_task3_plus {
    static Map<String ,Map<String ,Integer>> studentMistakes = new HashMap<>();
    public static void main(String[] args){
```

```
        Scanner sc = new Scanner(System.in);
        String initialData
="math:5,English:10,Chinese:10,math:20,English:10,chemistry:30,math:10,math:20";
        parseInitialData("小明" ,initialData);
        while(true){
            String input = sc.nextLine().trim();
            String[] part1 = input.split(":");
            if(input.equals("exit") || part1.length!=3){
                break;
            }
            String name = part1[0].trim();
            String subject =part1[1].trim();
            int count = Integer.parseInt(part1[2].trim());
            addMistake(name ,subject ,count);
        }
        getMax();
    }

    // 添加错题数据
    static void addMistake(String name ,String subject ,int count){
        if(!studentMistakes.containsKey(name)){
            studentMistakes.put(name ,new HashMap<>());
        }
        Map<String ,Integer> temp =studentMistakes.get(name);
        temp.put(subject ,count +temp.getOrDefault(subject ,0));
    }

    // 解析初始数据
    static void parseInitialData(String name ,String data){
        String[] part1 = data.split(",");
        for(String entry : part1){
            String[] part2 = entry.split(":");
            String subject = part2[0].trim();
            int count = Integer.parseInt(part2[1].trim());
            addMistake(name ,subject ,count);
        }
    }

    //输出结果
    static void getMax(){
        for(String name : studentMistakes.keySet()){
            Map<String ,Integer> mistakes = studentMistakes.get(name);
            Map.Entry<String ,Integer> maxEntry = null;
            for(Map.Entry<String ,Integer> entry : mistakes.entrySet()){
                if(maxEntry == null){
                    maxEntry = entry;
                } else{
                    int count = entry.getValue() , maxCount = maxEntry.getValue();
                    if(count > maxCount){
                        maxEntry = entry;
                    }
                }
            }
            System.out.println(name + " 错的最多的科目: " + maxEntry.getKey() + "
```

```
  (" + maxEntry.getValue() +"题) ");
          }
      }

  }
```