考试要求

每题作为一个项目,非Web项目需要有main主函数,Web项目需要看运行效果,可以翻阅资料(不允许上网),考试时长2小时,要求:

1,项目名称为Exam<题号>

2,使用Maven构建项目,考试完成后使用git提交到github.

3,使用UTF-8编码

4,每个项目必须有README文件,说明如何运行各个程序

github地址： https://github.com/DingYunGang/Exam

第一题 简答题(每题两分)

1. Java实现多线程都有哪些方式？

答案：

1. 继承Thread类创建线程 2.实现Runnable接口创建线程

3.使用ExecutorService、Callable、Future实现有返回结果的多线程

2. 如何实现线程同步？举例说明

答案：

1.synchronized关键字修饰的同步方法 2.同步方法块，synchronized关键字修饰的语句块

3.使用特殊域变量(volatile)实现线程同步 4.使用重入锁实现线程同步

5.使用局部变量实现线程同步 6.使用阻塞队列实现线程同步

7.使用原子变量实现线程同步

3. 对比List/Set/Map，以及简要阐述其应用场景？

答案：

1.List,Set都是继承自Collection接口，Map则不是

2.List特点：元素有放入顺序，元素可重复 ，Set特点：元素无放入顺序，元素不可重复，重复元素会覆盖掉

3. set和list对比

Set：检索元素效率低下，删除和插入效率高，插入和删除不会引起元素位置改变。

List：和数组类似，List可以动态增长，查找元素效率高，插入删除元素效率低，因为会引起其他元素位置改变。

1. Map适合储存键值对的数据
2. 线程安全集合类与非线程安全集合类 LinkedList、ArrayList、HashSet是非线程安全的，Vector是线程安全的; HashMap是非线程安全的，HashTable是线程安全的; StringBuilder是非线程安全的，StringBuffer是线程安全的。

4. 简要阐述一下HashTable/HashMap/ConcurrentHashMap的区别和联系？

答案：

HashMap 是 Hashtable 的轻量级实现（非线程安全的实现） ，他们都完成了 Map 接口，主要区别在于

HashMap 允许将 null 作为一个 entry 的 key 或者 value，而 Hashtable 不允许。

HashMap 允许将 null 作为一个 entry 的 key 或者 value，而 Hashtable 不允许。

HashMap 把 Hashtable 的 contains 方法去掉了， 改成 containsvalue 和 containsKey。 因为contains 方法容易让人引起误解。

Hashtable 继承自 Dictionary 类， 而 HashMap 是 Java1.2引进的 Map interface 的一个实现。

最大的不同是，Hashtable 的方法是 Synchronize 的，而 HashMap 不是，在多个线程访问Hashtable 时，不需要自己为它的方法实现同步，而 HashMap 就必须为之提供外同步。

ConcurrentHashMap

1.ConcurrentHashMap是java.util.concurrent 包中的，它是线程安全的哈希表，它是通过“锁分段”来保证线程安全的。

2. ConcurrentHashMap将哈希表分成许多片段(Segment)，每一个片段除了保存哈希表之外，本质上也是一个“可重入的互斥锁”(ReentrantLock)。

3. 多线程对同一个片段的访问，是互斥的；但是，对于不同片段的访问，却是可以同步进行的。

第2-4题 上机编程题:

第二题(8分)

编程实现工资个人所得税计算程序：

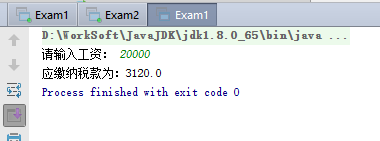


现设定各项社会保险金为0，从命令行中接收用户输入的工资数,在输出中打印出需缴纳的税费,形式如下:

**请输入工资: xxxx**

**所需要缴纳的税费为: xxxx**

**import** java.util.Scanner;  
  
**public class** Exam1 {  
 **public static void** main(String[] args) {  
 System.***out***.print(**"请输入工资"**);  
 Scanner in = **new** Scanner(System.***in***);  
 **double** sum =in.nextDouble();  
 **double** tex=0;  
 **double** salary = sum -3500;  
 **if** (salary>=0 && salary<1500){  
 tex = salary \*0.03;  
 }**else if** (salary>=1500 && salary<4500){  
 tex = salary \* 0.1 - 105;  
 } **else if**(salary>=4500 && salary<9000){  
 tex = salary \* 0.2 - 555;  
 }**else if**(salary>=9000 && salary<35000){  
 tex = salary\*0.25 - 1005;  
 }**else if**(salary>=35000 && salary<55000){  
 tex = salary\*0.3 -2755;  
 }**else if**(salary>=55000 && salary<80000){  
 tex = salary\*0.35 - 5505;  
 }**else if**(salary>=80000){  
 tex = salary\*0.45 - 13505;  
 }  
 System.***out***.print(**"应缴纳税款为："**+tex);  
 }  
}



第三题(8分)

输入某年某月某日，给出下一天的日期？当然输入日期不合法需要做出错误提示。

从控制台读取日期,日期输入的格式为”2015-07-26”形式.(不允许使用Calendar,Date,DateTime类中的函数),输入输出的形式如下:

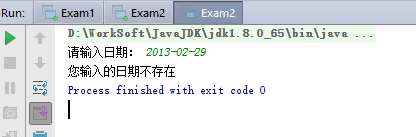
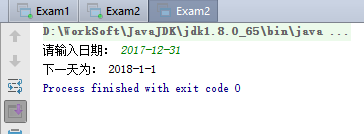
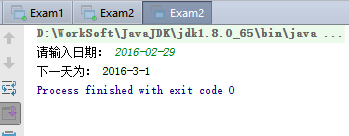
**请输入日期: 2017-02-28**

**下一天为 2017-03-1**

**请输入日期: 2017-02-29**

**“您输入的日期不存在”**

**import** java.util.Scanner;  
  
*/\*\*  
 \* Created by 丁云刚 on 2018/8/4.  
 \*/***public class** Exam2 {  
 **public static void** main(String[] args) {  
 System.***out***.print(**"请输入日期"**);  
 Scanner in = **new** Scanner(System.***in***);  
 String date = in.nextLine();  
 String[] ary = date.split(**"-"**);  
 **int** year = Integer.*parseInt*(ary[0]);  
 **int** month = Integer.*parseInt*(ary[1]);  
 **int** day = Integer.*parseInt*(ary[2]);  
 **int**[] mon;  
 mon = **new int**[]{0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};  
 **if**(year < 1 || month < 1 || month > 12) { *//判断日期是否合理* System.***out***.print(**"您输入的日期不存在"**);  
 **return**;  
 }  
 **boolean** flag =**false**;  
 **int** i=0;  
 **if**((year%4==0&&year%100!=0) || year%400==0) { *//判断日期是否为闰年* flag = **true**;  
 i=1;  
 }  
 **if**( ((day < 1 || day > mon[month]) && (month != 2) ) || (day < 1||(month == 2) && (day > 28 + i)) ) {  
 System.***out***.print(**"您输入的日期不存在"**); *//判断日期是否合理* **return**;  
 }  
  
 *//下面是计算下一天的日期* **if**(month==2)  
 {  
 **if**((day==28&&!flag)||(day==29&&flag)) {  
 day = 1;  
 month++;  
 }  
 **else** day++;  
 }  
 **else if**(month==12)  
 {  
 **if**(day==31)  
 {  
 day=1;  
 month=1;  
 year++;  
 }  
 **else** day++;  
 }  
 **else if**(month==1||month==3||month==5||month==7||month==8||month==10)  
 {  
 **if**(day==31)  
 {  
 day=1;  
 month++;  
 }  
 **else** day++;  
 }  
 **else** {  
 **if**(day==30)  
 {  
 day=1;  
 month++;  
 }  
 **else** day++;  
 }  
 System.***out***.print(**"下一天为： "**+year+**"-"**+month+**"-"**+day);  
 }  
}



第四题(8分)

随机生成50个小于100的整数,放入List中,将List中的数据除以10,以结果的整数值作为key放入Map中,得到如{1=>[11,10,12],2=>[21,24,23]}的Map,再将Map中key对应的数组进行排序,得到如{1=>[10,11,12],2=>[21,23,24]}

输出的形式如下:

**随机生成50个小于100的数,分别为: xxx,xxxx,xxxx**

**Map中的数据为: {1=>[11,10,12],2=>[21,24,23]}**

**排序后的Map为: {1=>[10,11,12],2=>[21,23,24]}**

**import** java.util.\*;  
  
*/\*\*  
 \* Created by 丁云刚 on 2018/8/4.  
 \*/***public class** Exam3 {  
 **public static void** main(String[] args) {  
 **int** x [] = **new int** [50];  
  
  
 Map map = **new** HashMap();  
 Map map2 = **new** HashMap();  
 System.***out***.print(**"随机生成50个小于100的数,分别为:"**);  
 **for**(**int** i=0;i<50;i++) {  
 x[i] = (**int**) (Math.*random*() \* 100);  
 System.***out***.print(x[i]+**" "**);  
 }  
 System.***out***.println();  
 System.***out***.println(**"Map中的数据为:"**);  
 **for**(**int** i=0;i<10;i++){  
 List list = **new** ArrayList();  
 **for**(**int** j=0;j<50;j++){  
 **if**(x[j]/10==i){  
 list.add(x[j]);  
 }  
 }  
  
 map.put(i,list);  
 }  
  
 **for**(**int** i=0;i<10;i++){  
 System.***out***.println(map.get(i)+**" "**);  
 }  
  
 System.***out***.println(**"排序后的Map为:"**);  
  
 **for**(**int** i=0;i<10;i++){  
 List list2 = **new** ArrayList();  
 **for**(**int** j=0;j<50;j++){  
 **if**(x[j]/10==i){  
 list2.add(x[j]);  
 }  
 }  
 Collections.*sort*(list2);  
 map2.put(i,list2);  
 }  
  
 **for**(**int** i=0;i<10;i++){  
 System.***out***.println(map2.get(i)+**" "**);  
 }  
  
 Comparator c = **new** Comparator<Integer>() { *//重写comparator类对arraylist进行排序*   
 **public int** compare(Integer o1, Integer o2) {  
  
 **if**((**int**)o1<(**int**)o2)  
 **return** 1;  
 **else return** -1;  
 }  
 };  
 }  
}

