---------------------------------------------------

计划课程安排：

FUNDAMENTAL(7天)         该阶段已结束

OOP(11天)           该阶段已结束

JAVASE01(10天) 该阶段已结束

JAVASE02(13天) ←当前为该阶段第12天

WEBBASIC(7天)

JQUERY(2天)

DATABASE(6天)

JDBC(3天)

SERVLETJSP(12天)

SPRINGMYBATIS01(8天)

AJAX(2天)

SPRINGMYBATIS02(12天)

LINUX(2天)

NGINX(2天)

REDIS(2天)

T-STORE(5天)

预计合计：103天

预计剩余：64天

注：上述信息随时可能发生更改，仅供参考。

---------------------------------------------------

**2019年4月25日**

**Java就业班-第49天**

**8:30实训室2-晨讲**

今日晨讲主题——冒泡算法

这位兄弟没有怎么讲，就是在写代码。正好机会难得，把以前的知识复习一下也好。

package Training;

import java.util.Arrays;

import java.util.Random;

/\*\*

\* 冒泡算法

\* @author Administrator

\*

\*/

public class ArrayDemo {

public static void main(String[] args) {

//生成随机数组

Random rd = new Random();

int array[] = new int[10];

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

array[i] = rd.nextInt(100);

}

System.out.println("随机数组：" + Arrays.toString(array));

//冒泡排序

int temp = 0;

for (int i = 0; i < array.length-1; i++) {

for (int j = 0; j < array.length-1-i; j++) {

if(array[j]<array[j+1]) {//"<"表示从大到小排序

temp = array[j];

array[j] = array[j+1];

array[j+1] = temp;

}

}

}

System.out.println("冒泡排序后数组：" + Arrays.toString(array));

}

}

**9:00实训室2-正课**

开始上课。

今天首先介绍一下Date类，日期相关的功能。

Java中的时间计算原理，是按照固定的时间点纪元（epoch）,是UTC时间1970年1月1日00:00:00的时间点为基准，以long值进行毫秒级的计算。

※UTC（Universal Time Coordinated世界调整时间）

※GMT（Greenwich Mean Time格林威治时间）←当前现实中使用的时间

老版的Date类本身几乎已经不再使用了（过时了）。

**9:17实训室2-正课**

讲师开始聊自己在新疆上班时的故事……

穿毛衣去吐鲁番………

核桃大的葡萄干………

…………（话题转到蠕虫病毒）

98、99年期间蠕虫病毒泛滥。蠕虫病毒就是强制占用CPU的大量运算资源，导致电脑的运行速度非常慢。

当时的电脑杂志上疯狂宣传说千年虫（即2000年病毒爆发传闻）病毒的危害性，说的很玄幻。就是说到达2000年的一瞬间所有的蠕虫病毒统一时间爆发，严重的话可能会导致电脑爆炸！←当然是假的。

不过电脑历史上确实存在这样一个和时间有关的BUG，基本上都是每100年出现一次，属于是底层C语言的一个硬性bug…………

70年代的电脑配置差到难以想象，这就要说到银行了，银行使用计算机的传统就是能用则绝对不换新……

到底什么是bug，这个称呼是怎么来的呢？来源于微机时代，当初在运算时还主要是依靠晶体管，在美国计算机博物馆里保存着一只飞蛾虫子，这个虫子就是bug称呼的源头。说是当时电路在执行逻辑运算时，突然出现了预料之外的问题，后来发现是因为这只飞蛾在温度比较高的晶体管上停留导致飞蛾扑火的现象发生，于是留下了这么一个在计算机业界中只要出现硬件软件故障则称之为bug的说法。

**9:46实训室2-正课**

回到课上。

Date类

package date;

import java.util.Date;

/\*\*

\* java.util.Date

\* Date的每一个实例用于表示一个具体的时间点，内部维护一个long值，

\* 表示的是自1970年1月1日00:00:00

\* 到当前Date表示的时间之间所经过的毫秒。

\*

\* 由于Date存在时区等问题，导致大部分操作时间的方法都被声明为过时的，开发时不再建议使用。

\* 因此我们现在仅使用Date表示一个时间。

\* @author Administrator

\*

\*/

public class DateDemo01 {

public static void main(String[] args) {

Date date = new Date();

System.out.println(date);

//大部分方法都被声明为过时的

date.getYear();//例如调用此方法时，方法名上会出现删除线。

/\*

\* 获取Date内部维护的long值使当前Date表示这个时间

\*/

date.setTime(0);

System.out.println(date);

}

}

**10:31实训室2-正课**

SimpleDateFormat类：

package date;

import java.text.SimpleDateFormat;

import java.util.Date;

/\*\*

\* java.text.SimpleDateFormat

\* 可以按照指定的日期格式在Date与String之间相互转换。

\* @author Administrator

\*

\*/

public class SimpleDateFormatDemo01 {

public static void main(String[] args) {

Date date = new Date();

System.out.println(date);

/\*

\* 时间表示格式：

\* 2019年4月25日10:23:37 上午 星期四

\* yyyy-MM-dd HH:mm:ss a E

\*/

SimpleDateFormat sdf = new SimpleDateFormat("yyy-MM-dd HH:mm:ss a E");

/\*

\* String format(Date date)

\* 将给定的Date对象按照SDF指定的日期格式转换为字符串

\*/

String str = sdf.format(date);

System.out.println(str);

}

}

**10:48实训室2-正课**

package date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

/\*\*

\* 讲一个字符串解析为Date对象

\* @author Administrator

\*

\*/

public class SimpleDateFormatDemo02 {

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

String str = "2008-08-08 20:08:08";

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

/\*

\* Date parse(String str)

\* 将给定的字符串按照SDF指定的日期格式解析为Date对象。

\* 若给定的字符串与SDF指定的日期格式不匹配时

\* 会抛出解析异常。

\*/

Date date = sdf.parse(str);

System.out.println(date);

}

}

**11:43实训室2-正课**

随意练习：

package date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Scanner;

/\*\*

\* 编写一段程序：

\* 用户在控制台输入自己的生日，格式如：yyyy-MM-dd

\* 然后经过程序计算，输出到今天为止一共活了多少天。

\* 在输出其出生10000天的纪念日是哪天，格式同上。

\* @author Administrator

\*

\*/

public class Demo {

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

Date date = new Date();

SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMdd");

Scanner sc = new Scanner(System.in);

long now = date.getTime();//获取至今全部时间(ms)

System.out.println("输入生日：（例：20190425）");

date = sdf.parse(sc.nextLine());

long birthday = date.getTime();//获取至生日全部时间(ms)

long alive = (now-birthday)/1000/60/60/24;//存活日(day)

System.out.println("至今存活："+alive+ "天");

long theDay = 10000L\*24\*60\*60\*1000;//10000天时间(ms)

long time = birthday+theDay;

date = new Date(time);

System.out.println("10000天纪念日：" + date);

}

}

※注意纪念日的计算在编译中是默认按照int型计算的，所以需要加上L。

**11:50实训室2-午休时间**

终于下课了。

不行了，好热好困，我顶不住了，我得睡会……

**14:05实训室2-正课**

睡醒了，真的舒服。最主要的是机构终于开空调了！唯一的美中不足是做噩梦了，梦到了班主任作妖……

开始上课了。

Calendar类。

Calendar是抽象类，其具体子类针对不用国家的日历系统，其中应用最广泛的是GregorianCalendar(格里高利历，即通用的阳历)，对应世界上绝大多数国家/地区使用的标准日历系统。

package date;

import java.util.Calendar;

import java.util.Date;

/\*\*

\* java.util.Calendar 日历类

\* Calendar是一个抽象类，定义了操作时间的一系列方法。

\* 常用实现类GregorianCalendar,即：阳历

\*

\* Calendar提供了静态方法getInstance,可以根据当前系统

\* 所在地区获取一个适用的实现类实例，大部分获取的都是阳历。

\* @author Administrator

\*

\*/

public class CalendarDemo01 {

public static void main(String[] args) {

Calendar calendar = Calendar.getInstance();

System.out.println(calendar);

/\*

\* 将当前Calendar表示的日期以Date实例形式返回。

\*/

Date date = calendar.getTime();//返回Date对象

/\*

\* void setTime(Date date)

\* 调整Calendar表示给定的Date对象所表示的日期

\*/

System.out.println(date);

}

}

单独案例：

package date;

import java.util.Calendar;

import java.util.Date;

/\*\*

\* Calendar提供了获取指定时间分量对应值的方法：

\* int get(int field)

\*

\* Calendar定义了大量的常量表示不同的时间分量

\* @author Administrator

\*

\*/

public class CalendarDemo02 {

public static void main(String[] args) {

Calendar calendar = Calendar.getInstance();

//获取年

int year = calendar.get(Calendar.YEAR);

System.out.println(year);

//获取月 ※月是从0开始计算的，所以取值要+1

int month = calendar.get(Calendar.MONTH)+1;

System.out.println(month);

//获取日

int day = calendar.get(Calendar.DAY\_OF\_MONTH);

System.out.println(year+"-"+month+"-"+day);

//2019-4-25

/\*

\* 获取日

\* 与"天"相关的时间分量

\* DAY\_OF\_MONTH :月中天

\* DAY\_OF\_WEEK :周中天

\* DAY\_OF\_YEAR :年中天

\* DATE :月中天

\*/

//※星期日为一周中的第一天，所以在取值时需要-1。

int day1 = calendar.get(Calendar.DAY\_OF\_WEEK)-1;

System.out.println("星期" + (day1));

int day2 = calendar.get(Calendar.DAY\_OF\_WEEK\_IN\_MONTH);

System.out.println("月中第" + day2 + "周");

int day3 = calendar.get(Calendar.DAY\_OF\_YEAR);

System.out.println("年中第" + day3 + "天");

int h = calendar.get(Calendar.HOUR\_OF\_DAY);//日中时

int m = calendar.get(Calendar.MINUTE);//分

int s = calendar.get(Calendar.SECOND);//秒

System.out.println(h + ":" + m + ":" + s);

//汉字显示

String weekDay[] = {"日","一","二","三","四","五","六"};

System.out.println("今天是星期" + weekDay[day1]);

//获取指定时间分量所允许的最大值

int days = calendar.getActualMaximum(Calendar.DAY\_OF\_YEAR);

System.out.println("今年共有" + days + "天");

days = calendar.getActualMaximum(Calendar.DAY\_OF\_MONTH);

System.out.println("当月共有" + days + "天");

}

}

**15:33实训室2-正课**

通过Calendar调整时间。

package date;

import java.util.Calendar;

/\*\*

\* void set(int field, int value)

\* 调整当前Calendar指定时间为给定的值

\* @author Administrator

\*

\*/

public class CalendarDemo03 {

public static void main(String[] args) {

Calendar calendar = Calendar.getInstance();

/\*

\* 调整文2008-08-08 20:08:08

\*/

//调整年为2008

calendar.set(Calendar.YEAR, 2008);

System.out.println(calendar.getTime());

calendar.set(Calendar.MONTH, Calendar.AUGUST);

System.out.println(calendar.getTime());

calendar.set(Calendar.HOUR\_OF\_DAY,20);

calendar.set(Calendar.MINUTE,8);

calendar.set(Calendar.SECOND,8);

System.out.println(calendar.getTime());

}

}

**15:45实训室2-正课**

增减调整时间：

package date;

import java.util.Calendar;

/\*\*

\* void add(int field, int amount)

\* 对指定时间分量加上给定的值，若给定的值为负数则是减去

\* @author Administrator

\*

\*/

public class CalendarDemo04 {

public static void main(String[] args) {

Calendar calendar = Calendar.getInstance();

/\*

\* 3年2个月零25天以后是哪天？

\*/

//加3年

calendar.add(Calendar.YEAR, 3);

System.out.println(calendar.getTime());

//加两个月

calendar.add(Calendar.MONTH, 2);

System.out.println(calendar.getTime());

//加25天

calendar.add(Calendar.DAY\_OF\_YEAR, 25);

System.out.println(calendar.getTime());

}

}

**16:53实训室2-正课**

随意练习：

package date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

import java.util.Scanner;

/\*\*

\* 计算商品促销日期

\* 程序启动后要求输入一个商品的生产日期：yyyy-MM-dd

\* 然后再输入一个数字表示保质期的天数。

\* 经过程序计算，输出该商品的促销日，格式同上。

\*

\* 计算规则：商品过期日前两周的周三

\* @author Administrator

\*

\*/

public class Demo02 {

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

Calendar calendar = Calendar.getInstance();

System.out.println("输入商品的生产日期：(例：2008-01-01)");

Scanner sc = new Scanner(System.in);

String made = sc.nextLine();

System.out.println("输入保质期天数：（例：30）");

int limit = sc.nextInt();

int year = Integer.parseInt(made.substring(0,4));

int month = Integer.parseInt(made.substring(5,7));

int day = Integer.parseInt(made.substring(8));

calendar.set(year, month-1, day);

//计算促销日

calendar.add(Calendar.DAY\_OF\_YEAR, limit-14);

calendar.set(Calendar.DAY\_OF\_WEEK, 4);

Date date = calendar.getTime();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String sellDay = sdf.format(date);

System.out.println("促销日为："+ sellDay);

}

}

**17:00实训室2-正课**

最后，现阶段的最后一个内容——Lambda表达式。

package lambda;

/\*\*

\* lambda表达式 JDK8推出后推出的一个新特性

\* lambda旨在让我们可以以“函数式编程”

\*

\* lambda可以用更简便的语法创建匿名内部类

\* 语法：

\* （参数列表）->{

\* 方法体

\* }

\*

\* 需要注意，lambda创建的匿名内部类所述的接口

\* 必须只能有一个抽象方法，否则编译不通过。

\* @author Administrator

\*

\*/

public class LambdaDemo01 {

public static void main(String[] args) {

//常规写法

Runnable r1 = new Runnable() {

public void run() {

System.out.println("hello");

}

};

//lambda表达式写法

Runnable r2 = ()->{

System.out.println("hello");

};

/\*

\* 如果方法只有一句代码，那么lambda的{}也可以省略

\*/

Runnable r3 = ()->System.out.println("hello");

}

}

※在API文档中凡是注解有@Functioninterface的接口都可以使用lambda表达式写法。

**17:22实训室2-正课**

当lambda表达式里的方法有参时的写法：

package lambda;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

/\*\*

\* 有参方法

\* @author Administrator

\*

\*/

public class lambdaDemo02 {

public static void main(String[] args) {

List<String> list = new ArrayList<>();

list.add("Kamui");

list.add("Subaru");

list.add("Luna");

Collections.sort(list);

// Comparator<String> c = new Comparator<String>() {

// public int compare(String o1,String o2) {

// return o1.length()-o2.length();

// }

// };

//第一种写法

// Comparator<String> c = (o1,o2)->{

// return o1.length()-o2.length();

// };

//第二种写法

Comparator<String> c = (o1,o2)->

o1.length()-o2.length();

Collections.sort(list, c);

System.out.println(list);

}

}

**17:30实训室2-下课时间**

今天的课程就到这里了，讲师的课程也就到这里了，现阶段也就到这里结束了。

明天开始会是新的讲师接手，并开始正式学习Web，首先就学习静态页面的制作。

以上。

平心而论，这位讲师还是很不错的，虽然时不时会放飞自我夸天夸地，但课程内容讲的还是比较通俗易懂，也不至于很催眠。

**18:32实训室2-晚自习**

开始考试了，仍然是没有人监考。

错题集：

1.下列方法被调用后，变量result的结果是（ ）。

public static void main(String[] args) {

StringBuilder str = new StringBuilder("abc");

int result = getResult(str);

System.out.println(result);

}

public static int getResult(StringBuilder str){

str.append("abc");

str.delete(0,3);

str.insert(0, "abc");

str.replace(0,3, "abc");

return str.length();

}

2. 运行下面程序:

public class Test01{

public static void main(String[] args) {

try {

test();

System.out.println("1...");

} catch (ArithmeticException e) {

System.out.println("2...");

} catch (Exception e) {

System.out.println("3...");

} finally {

System.out.println("end...");

}

}

public static void test() {

String str = "cc";

str.compareTo("abc");

}

}

输出的结果是：（）。

3. 给出以下代码，请问该程序的运行结果是（ ）。

String s1 ="ab";

String s2 ="abcd";

String s3="cd";

String s4 =s1+s3;

s1=s4;

System.out.println("s1"+((s1==s2)?"==":"!=")+"s2");

4. 下面属于线程安全的类是（ ）。

A. ArrayList

B. StringBuilder

C. HashMap

D. Hashtable

5. 对下列运算结果，判断正确的是（ ）；

A. "1az098".matches("\\d[a-z]{2,8}[0-9]+")

结果为true

B. "1az098".matches("\\d[a-z]{2,8}[0-9]+")

结果为false

C. "张三,,,李四,,王五,,,,,,马六,,".split("[,]+").length == 1;

该表达式结果返回true

D. "张三,,,李四,,王五,,,,,,马六,,".split("[,]+").length == 4;

该表达式结果返回true

以上只是从错题中摘出来的几道，不是全部。

**20:25实训室2-晚自习**

终于是做完了，根据我自己的统计至少错了14道题（共50道题，真实总分72分或以下），而考试成绩肯定是高分，毕竟我是百度和开发工具都用上了。

现在很清楚我的水平了，但即便是如此，我也要拿个高分出来打班主任的脸！

看样子今天是就到这里结束了。

该死的机构晚上就不开空调了，正准备赞扬一下的，前言收回。

下个星期开始班上应该会少一些人吧，先回宿舍了。

**21:06宿舍-休息**

JavaSE阶段终于是结束了，java的语言课程似乎是到这里就暂时告一段落了，之后想要提升自身对java的理解似乎只能通过自身的不断阅读书籍、源代码和实战项目的经验来提高了。而我的基础自认为打得并不牢固，如果说想要进一步强化自身的基础水平，我觉得我需要的是更多的练习。

明天再上一天的课，晚自习我打算翘课回家（实训室太热），后天休息一天，再上四天课就可以稍稍放松一下地休个爽了。不过，估计也逃脱不了给家里老一辈人当司机开车接送的活吧……哎，又得听他们嘴臭了。

今日感想总结：没想到培训机构也这么热衷于应试考试，出的题一个比一个偏……似乎是在告诉我面试的题也会是这么偏。

今日评分：5/10分

今天就到这里，明天继续投稿。

感想支持！

================================================================

**Acer程序员群：790482850**

**up主的java学习日记word文件及全部代码（随日记同步更新）：**

Adress：github.com/wt62635