

**计算机与信息 学院实验报告**

|  |  |  |
| --- | --- | --- |
| 实验课程： | 面向对象程序设计实验 | |
| 实验编号： | 实验九 | |
| 实验名称： | 常用工具类 | |
| 实验人员： | 学号 | 22111302041 |
| 姓名 | 高蝶 |
| 班级 | 22级计算机科学与技术1班 |
| 指导教师： | 陶涛 | |
| 实验室： | 2号学苑楼202 | |
| 实验日期： | 2023.4.18 | |

# 实验目的

1. 理解String与StringBuffer类的区别；

2. 掌握String类的常用方法；

3. 掌握StringBuffer类的常用方法；

4. 掌握日期类的常用方法。

# 实验要求

1. 所编写的程序应符合类的封装、继承、实现接口要求，符合面向对象程序编写规范；

2. 能利用继承性、多态性、异常处理等进行复杂程序的设计和实现；

3. 对问题进行充分的需求分析，设计正确、全面的测试数据，设计类的定义框架及UML图后，再进行编程实现；

4. 所设计的测试数据全部通过后，再进行程序提交，提高程序的通过率。

# 实验内容

**1.验证型实验**

请检查该程序有无错误，若有则修改，使之正确。请写出程序的运行结果。

public class Main {

public static void main(String[] args){

String s="abcHDLmnkKl";

String news=s.replaceAll("[^A-Z]","");

StringBuffer sb=new StringBuffer(news);

String result=sb.reverse().toString();

System.out.println("结果----->"+ result);

}

}

**2.应用型实验**

**2.1 高精度计算**

double类型的精度是15-16位，有时需要有更高精度的小数计算，这时double类型就无法实现了，请你编写一个Java程序，解决此类问题。

输入：两个小数

输出：这两个小数的和、积

样例输入：

1.1

2.22

样例输出：

3.32

2.442

**2.2 随机数与文本加密**

RandomEncrypt类中包括明文字符串、密文字符串（字符编码的十六进制）以及密钥key（长整型数），使用key作为种子产生一个随机的短整数数列（k1,k2,k3,……，ki取值范围是[1，32767]），该数列的每一个数与明文（密文）字符串的每一个字符按照顺序进行异或^运算，从而完成加密或解密功能。

输入：若干个密钥（长整型数），整数n以及n个（以行为单位）文本；

输出：每个密钥分别与每个明文文本加密后的密文文本（十六进制）和解密文本（每行最多10个，以空格隔开）

样例输入：

12345678 20220418

2

Java2实用教程

安徽师范大学计算机与信息学院，21计算机类，面向对象程序设计

样例输出：

59E1 5796 742D 7113 285B 13DE 6F88 4124 1827

Java2实用教程

0222 084A 2A53 F271 714E 1326 9101 5FEA 0516 3DCA

26EF 2F67 7E7A D015 FEB4 2C44 4C31 D74B 7741 5FA3

18E6 FEF3 8E42 6042 6BC7 DE0C 2FA5 07E9 F01C 8BA4

安徽师范大学计算机与信息学院，21计算机类，面向对象程序设计

38E3 7EB6 313B 53F7 1E5C 7AB3 7DC8 74F7 6957

Java2实用教程

6320 216A 6F45 D095 4749 7A4B 8341 6A39 7466 77DA

05B9 0AAE 306A F873 A223 103E 3A77 BA7B 5958 3B59

4522 E565 D3BA 34A6 571F 8F4A 5F80 1C87 979A EEE9

安徽师范大学计算机与信息学院，21计算机类，面向对象程序设计

**2.3 消息筛选**

我们的手机每天都会收到很多消息，手机消息包含如下内容：发件人、消息发布时间与消息内容。现在要求按指定的时间段对手机消息进行筛选，将筛选后的信息按时间发布的先后顺序进行显示。

输入：n组手机消息，包括发件人、发布时间与内容，指定的时间段。

输出：按时间先后顺序显示筛选后的消息。

样例输入：

5

1001

2022-3-22 16:50:50

Hello,hello,hello

1001

2022-3-30 17:50:23

Hi,hi

1002

2022-3-17 9:20:22

Good

1003

2022-3-7 8:00:00

Happy birthday!

1002

2022-3-31 13:00:03

Ok

2022-3-10

2022-3-30

样例输出：

1002

2022-03-17 09:20:22

Good

1001

2022-03-22 16:50:50

Hello,hello,hello

1001

2022-03-30 17:50:23

Hi,hi

**3. 设计型实验**

**3.1 获奖人员的信息公布**

某一抽奖活动，需要接收用户输入的姓名和手机号码，然后根据手机号码将参与者信息排序后，从中选择第偶数下标的参与者作为最终的获奖人员，并将获奖人员信息打印输出

(1)定义一个活动参与者类Participant;

(2)成员变量：姓名和手机号码，加密后的姓名和手机号码；

(3)构造方法：默认的空方法，带参构造方法Participant (String name, String phoneNumber);

(4)成员方法：encrypt()加密参与者的信息；printEncryptedMess() 打印参与者加密后的信息；

(5)参与者信息有效性校验：姓名长度大于1；收件号码合法（11位数字、以1开头）；无效参与者信息直接丢弃；

(6)参与者信息加密规则：姓名只保留首尾两位字符，如果只有两位字符只保留第0位字符，其他位置使用’\*’代替；手机号码保留前三位数字和最后三位数字，其他位置使用’\*’代替

定义测试类，编写main函数，接收用户输入的若干个活动参与者的信息（最多不超过100位），筛选其中有效的信息，根据若干个参与者的手机号码进行排序，选择排序后的第偶数下标元素作为获奖者，将其信息加密后输出公布获奖信息；

输入：若干组参与者的姓名和手机号码，每一行代表一名参与者信息；

输出：按顺序输出的获奖者信息；

样例输入：

001 18225802237

a1 15209822169

a2 17775333598

b1 17353775037

0 18755098123

003 97855220484

样例输出：

winner list:

a\* 152\*\*\*\*\*169

a\* 177\*\*\*\*\*598

# 程序清单

2.应用型实验

（1）

测试数据

第1组：**1.1**

**2.22**

第2组：**1.11111111**

**2.222222222222**

第3组：**0.000000001**

**9.99999999999999999**

类的UML图：无

程序代码：package prj\_gaojinduadd;

import java.math.BigDecimal;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner reader=new Scanner(System.in);

BigDecimal n1=new BigDecimal(reader.next());

BigDecimal n2=new BigDecimal(reader.next());

System.out.println(n1.add(n2));

System.out.println(n1.multiply(n2));

}

}

（2）

测试数据

第1组：**12345678 20220418**

**2**

**Java2**

**static Scanner reader=new Scanner(System.in);**

第2组：**19034578 13242354 5854325484254**

**2**

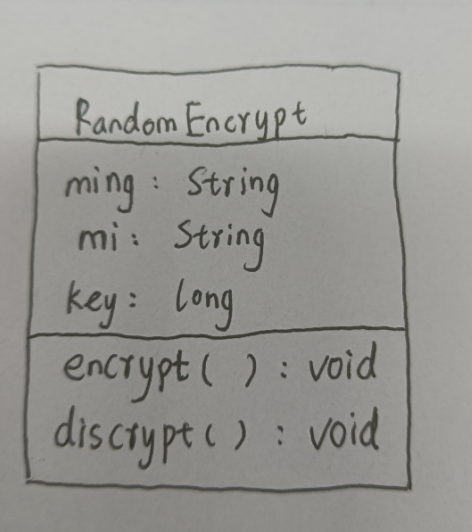
**jdsacn**

**dajnc liulbc lcjC scnkv;**

第3组：**123456789**

**1**

**javajava**

类的UML图：

程序代码：**package** prj\_random\_and\_encrypt;

**import** java.util.\*;

**public** **class** Main {

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

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

String **s**=reader.nextLine();

String [] **ss**=s.split(" ");

Long[] **num**=**new** Long[ss.length];

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

num[i]=Long.*parseLong*(ss[i]);

}

**int** **n**=reader.nextInt();

String [] **source**=**new** String [n];

reader.nextLine();

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

source[i]=reader.nextLine();

}

**int** **i**,**j**;

RandomEncrypt **randomencrypt**;

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

**for**(j=0;j<n;j++) {

randomencrypt=**new** RandomEncrypt (source[j],num[i]);

randomencrypt. encrypt();

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

System.***out***.println(source[j]);

}

}

}

}

**class** RandomEncrypt{

String ming,mi;

**long** key;

**RandomEncrypt**(){}

**RandomEncrypt**(String **ming**,**long** **key**){

**this**.ming=ming;

**this**.key=key;

}

**void** **encrypt**(){

**int** **len**=ming.length();

**int** [] **keys**=**new** **int**[len];

Random **random**=**new** Random(key);

**char**[] **mings**=ming.toCharArray();

**int** [] **ans**=**new** **int** [len];

**int** **cnt**=0;

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

keys[i]=random.nextInt(32768);

ans[i]= keys[i]^mings[i];

cnt++;

System.***out***.printf("%04X ",ans[i]);

**if**(cnt%10==0) {

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

cnt=0;

}

}

}

**void** **discrypt**() {

}

}

（3）

测试数据

第1组：

**5**

**1001**

**2022-3-22 16:50:50**

**Hello,hello,hello**

**1001**

**2022-3-30 17:50:23**

**Hi,hi**

**1002**

**2022-3-17 9:20:22**

**Good**

**1003**

**2022-3-7 8:00:00**

**Happy birthday!**

**1002**

**2022-3-31 13:00:03**

**Ok**

**2022-3-10**

**2022-3-30**

第2组：

**3**

**1001**

**2022-3-22 16:50:50**

**Hello,hello,hello**

**1001**

**2022-3-30 17:50:23**

**Hi,hi**

**1002**

**2022-3-17 9:20:22**

**Good**

**2021-12-12**

**2022-4-2**

第3组：

**3**

**1002**

**2022-3-17 9:20:22**

**Good**

**1003**

**2022-3-7 8:00:00**

**Happy birthday!**

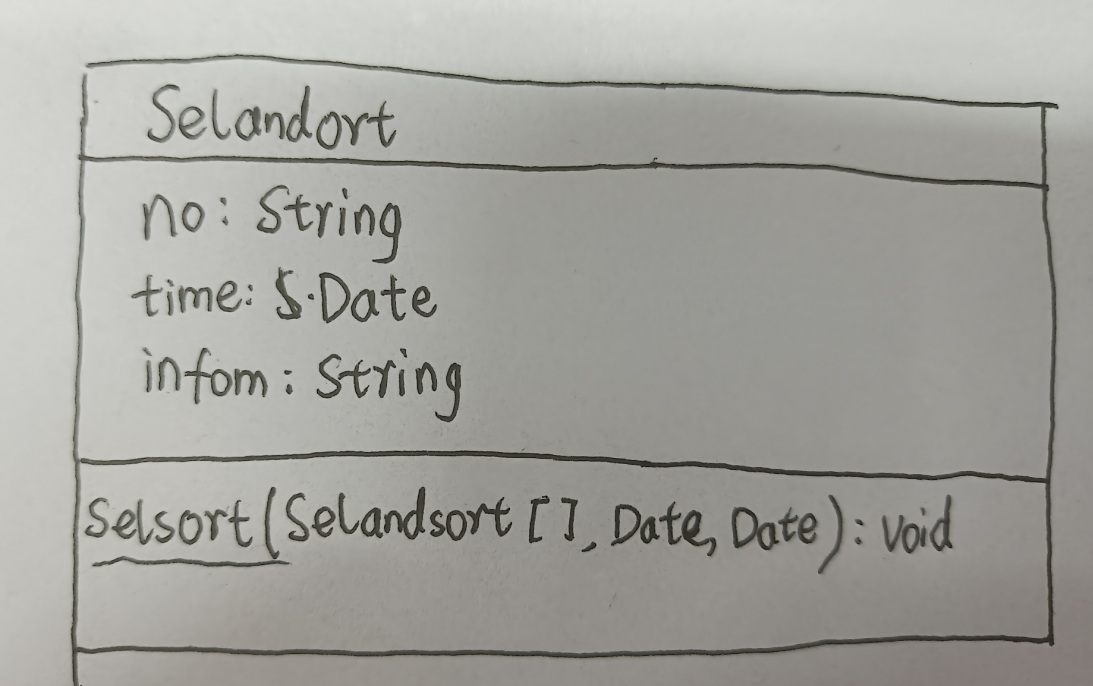
**1002**

**2022-3-31 13:00:03**

**Ok**

**2022-3-1**

**2023-1-1**

类的UML图：

程序代码：**package** prj\_xiaoxishaixuan;

**import** java.text.ParseException;

**import** java.text.SimpleDateFormat;

**import** java.util.\*;

**public** **class** Main {

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

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

SimpleDateFormat **format1** = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

SimpleDateFormat **format2** = **new** SimpleDateFormat("yyyy-MM-dd");

**int** **n**=reader.nextInt();

reader.nextLine();

String **no**,**infom**,**time**;

Selandsort [] **selandsort**=**new** Selandsort[n];

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

no=reader.nextLine();

time=reader.nextLine();

infom=reader.nextLine();

Date **temp** = format1.parse(time);

selandsort[i]=**new** Selandsort(no,temp,infom);

}

String **begin**=reader.nextLine();

String **end**=reader.nextLine();

Date **endtime** = format2.parse(end);

Calendar **cal** = Calendar.*getInstance*();

cal.setTime(endtime);

cal.add(Calendar.***DATE***, 1);

endtime= cal.getTime();

Selandsort.*selsort*(selandsort,format2.parse(begin),endtime);

}

}

**class** Selandsort{

String no;

Date time;

String infom;

**Selandsort**(){}

**Selandsort**(String **no**,Date **time**,String **infom**){

**this**.no=no;

**this**.time=time;

**this**.infom=infom;

}

**static** **void** **selsort**(Selandsort[] **sort** ,Date **begin**,Date **end**){

Arrays.*sort*(sort, Comparator.*comparing*(**a** -> a.time));

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

**if**(sort[i].time.compareTo(begin)>0 && sort[i].time.compareTo(end)<0){

System.***out***.println(sort[i].no);

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

String **str** = sdf.format(sort[i].time);

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

System.***out***.println(sort[i].infom);

}

}

}

}

3.设计型实验

（1）

测试数据

第1组：**001 18225802237**

**a1 15209822169**

**a2 17775333598**

**b1 17353775037**

**0 18755098123**

**003 97855220484**

第2组：**asdef 19134568745**

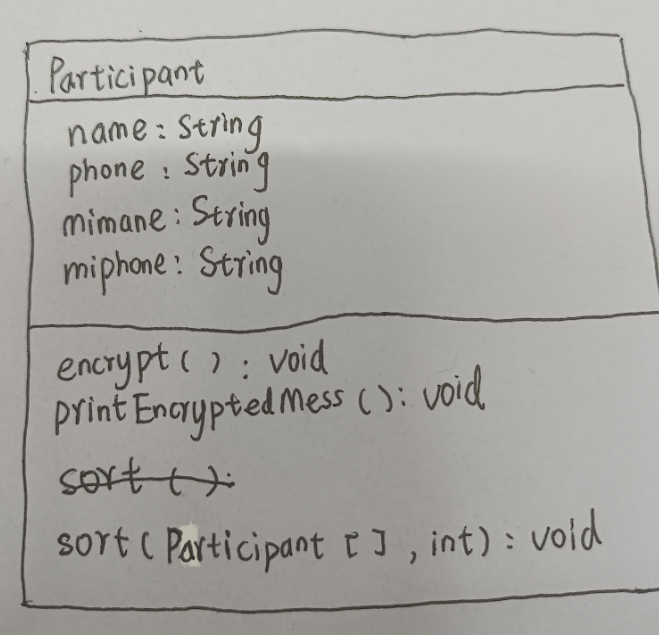
**qw 12345678970**

**1 17878787878**

第3组：**132 23123541545**

**awsed 52154544452**

**a 17890564580**

类的UML图：

程序代码：package prj\_huojiangrenyuanxinxigongbu;

import java.util.\*;

import java.util.Arrays;

public class Main {

public static void main(String[] args) {

Scanner reader=new Scanner(System.in);

Participant [] participants=new Participant[110];

int cnt=0;

try {

while(true) {

String s=reader.nextLine();

String [] ss=s.split(" ");

if(islogical(ss[0],ss[1])){

participants[cnt]=new Participant(ss[0],ss[1]);

cnt++;

}

if(!reader.hasNextLine()) {

break;

}

}

Participant p=new Participant();

p.sort( participants,cnt );

}catch(Exception e) {

System.out.println(e);

}

}

static boolean islogical(String ss0,String ss1) {

String format="1[3-9]\\d{9}";

if(ss0.length()>1&&ss1.length()==11) {

if(ss1.matches(format)) {

return true;

}

}

return false;

}

}

class Participant{

String name,phone,miname,miphone;

Participant(){}

Participant (String name, String phoneNumber){

this.name=name;

phone=phoneNumber;

}

void encrypt(){

if(name.length()>2) {

int len=name.length();

// miname=name.substring(0, 1)+name.substring(name.length()-1,name.length());

char [] nname=new char[len];

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

if(i==0||i==len-1) {

nname[i]=name.charAt(i);

}else {

nname[i]='\*';

}

}

miname=new String (nname);

}else {

miname=name.substring(0, 1)+"\*";

}

miphone=phone.substring(0,3)+"\*\*\*\*\*"+phone.substring(phone.length()-3,phone.length());

}

void printEncryptedMess() {

System.out.println(miname+" "+miphone);

}

void sort(Participant [] participants,int cnt ) {

// Arrays.sort(participants,(a,b)->{

// return a.phone.compareTo(b.phone);

// });

int i,j;

Participant t= new Participant();

for(i=1;i<=cnt-1;i++) {

for(j=1;j<=cnt-i;j++) {

if(participants[j-1].phone.compareTo(participants[j].phone)>0) {

t=participants[j-1];

participants[j-1]=participants[j];

participants[j]=t;

}

}

}

if(cnt!=0) System.out.println("winner list:");

for( i=0;i<cnt;i++) {

participants[i]. encrypt();

if(i%2==0) {

participants[i].printEncryptedMess();

}

//participants[i].printEncryptedMess();

}

}

}

# 实验结果及分析

1.验证型实验

程序有无错误，改正后（若有错）的运行结果是什么？

**无错误，运行结果如下：**

**结果----->KLDH**

2.应用型实验

（1）

各个类、方法的作用：**主类进行输入，创建BigDecimal类的对象，调用其中的方法完成加法及乘法运算。**

运行结果：**一:3.32**

**2.442**

**二:3.333333332222**

**2.46913579999975308642**

**三：10.00000000099999999**

**9.99999999999999999E-9**

（2）

各个类、方法的作用：**主类：输入秘钥和原文，双重循环调用RandomEncrypt中的方法完成加密解密。RandomEncrypt类中具体实现加密解密方法。**

运行结果：**一：**

**59E1 5796 742D 7113 285B**

**Java2**

**59D8 5783 743A 7106 2800 4823 1A80 242E 624F 73A5**

**6960 4F66 2579 4605 0198 2C04 4C65 5C8B 0CB2 38FC**

**64EF 01C2 194E 3436 3049 524D 55FD 5905 7BC3 006B**

**3B07 5215 4B8F 236B 000F 0499 0AA2 7DBA 0AA1 338E**

**71CB 157D 608B 0F0B 3CBD**

**static Scanner reader=new Scanner(System.in);**

**38E3 7EB6 313B 53F7 1E5C**

**Java2**

**38DA 7EA3 312C 53E2 1E07 214E 08C0 11FD 133F 39B5**

**4A36 6AAF 6B69 6E63 5D0F 107E 3A23 31BB 22AB 5C06**

**392B 1A54 44B6 60D2 0C91 030B 25D8 426B 1C45 6526**

**6707 6DA7 11E1 354C 25E0 1E67 01B1 2A7B 1B36 1780**

**58FF 5DE2 0A8E 4D9A 2955**

**static Scanner reader=new Scanner(System.in);**

**二：**

**5D47 755D 0BA6 3B64 7297 09EC**

**jdsacn**

**5D49 7558 0BBF 3B6B 7297 09A2 18B3 62C1 3EC3 1BE8**

**36D4 70A4 20BA 6159 6DAC 272B 7846 2582 48F2 46A8**

**4DF5 0131 6DF9 24EE**

**dajnc liulbc lcjC scnkv;**

**313E 20EC 2BD6 11C4 3CF6 5A84**

**jdsacn**

**3130 20E9 2BCF 11CB 3CF6 5ACA 4722 641A 19E1 5904**

**5762 4C9B 79CD 2E4E 49AA 0C16 66D6 6F3F 2B19 27E7**

**315B 57A9 79E6 703C**

**dajnc liulbc lcjC scnkv;**

**5DED 2E3D 6FC2 4953 0E90 0613**

**jdsacn**

**5DE3 2E38 6FDB 495C 0E90 065D 4576 7FD8 1D61 0D33**

**1C9D 3B5C 427A 350F 39FA 241E 3196 7FCF 29A2 1458**

**4384 694B 2BBC 1B45**

**dajnc liulbc lcjC scnkv;**

**三：**

**5495 61D5 3A0B 20FE 3196 1BD8 722F 3617**

**javajava**

（3）

各个类、方法的作用：**主类输入并用SimpleDateFormat将时间格式化再转为Canlendar便于比较排序再输出；Selandsort具体实现排序和按规定输出。**

运行结果：

一：

**1002**

**2022-03-17 09:20:22**

**Good**

**1001**

**2022-03-22 16:50:50**

**Hello,hello,hello**

**1001**

**2022-03-30 17:50:23**

**Hi,hi**

**二：**

**1002**

**2022-03-17 09:20:22**

**Good**

**1001**

**2022-03-22 16:50:50**

**Hello,hello,hello**

**1001**

**2022-03-30 17:50:23**

**Hi,hi**

**三：**

**1003**

**2022-03-07 08:00:00**

**Happy birthday!**

**1002**

**2022-03-17 09:20:22**

**Good**

**1002**

**2022-03-31 13:00:03**

**Ok**

3.设计型实验

（1）

各个类、方法的作用：**主类循环输入，并调用方法判断姓名和电话是否合法，然后调用方法将合法的排序后输出；Participant具体实现排序加密和输出。**

运行结果：**一：**

**winner list:**

**a\* 152\*\*\*\*\*169**

**a\* 177\*\*\*\*\*598**

**二：**

**winner list:**

**a\*\*\*f 191\*\*\*\*\*745**

三：

# 实验小结

[本次实验的反思、收获和体会]

**反思：1：还不太习惯查手册弄清一些类和方法的使用。**

**2：正则表达式老是不会写，匹配整数的，小数的，汉字的，合法日期，合法号码的，变化多样。**

**3：题目意思有时不能理解，有时还理解错了，带来不少麻烦。如“姓名只保留首尾两位字符，如果只有两位字符只保留第0位字符，其他位置使用’\*’代替”，其他位置使用’\*’代替这一句是对于前面两句说的，我的理解只有后者，题目的样例测试数据还不涉及这个问题，问了同学才找到错误。**

**4：Arrays.sort()函数不会用，老是用错，于是每次就用冒泡排序总觉得有点麻烦。**

**收获：1：了解到BigDecimal 类可用于高精度的计算，还挺好用的。**

**2：了解到随机数的种子得是Long型的，以及产生一定范围的随机数的写法，如random.nextInt();**

**3:字符编码的十六进制输出。**

**4：SimpleDataFormat用于日期格式化，及Date,Calendar类型的相互转化。**

**体会：对于一些方法的使用不熟悉，遇到题目也想不起来要用，只能通过多看题多敲代码来培养自己的思维。**