pta？上机考，判断题10哥1分，选择15个2分，程序填空5个2分，主观50分，1-2简答题文字性质，3-5编程题多是最后几章（10，15，15），

考试章节

1-10（第一章记不得就看看，7，8，9可能简答，或者大题用上思想）
12（12.1-12.4，12.6）（不会答题，但小题的输入输出的类要看一下）
13（13.1-13.4）（）
14（14.1-14.5）（基本概念搞清楚）
19（19.1-19.4）

常用类

object
system
math
calender
date

第一章

编译（c = compile），产生.class的字节码，一个类产生一个字节码

```
javac xxx.java
```

运行主类名java字节码文件

```
java xxx
```

`package ch01;` -> 建立名叫ch01的软件包

`public static void main(String[] args)`定义主方法

第二章

标识符

- 只能使用字母（含汉字），下划线， 货币符号（如$和￥）和数字
- 必须以字母、下划线或者货币符号开头

基础数据类型

P17

String不是基础数据类型，是类（class）类型

表达式

```
System.out.prinln(2.3/0); //输出 Infinity
System.out.prinln(0.0/0); //输出 NaN
```

自动转换顺序

byte->short->int->long->float->double

第三章

方法头可声明为`statc`，静态方法属于类，调用时使用Class.fuck即可。非静态则必须先new obj，再obj.fuck调用

输入输出

`java.util`包的`Scanner`类的`nextBoolean`， `nextByte`，`nextShort`， `nextInt`，`nextLong`，`nextFloat`，`nextDouble`，`next`和`nextLine`

`System.out`的println和print方法，ln换行，print不换行。还有格式化输出printf。

- %b boolean
- %d byte, short, int, long等整型
- %f float或double
- %e 指数形式输出float
- %c 输出char
- %s String

控制数据宽度

- %md 占m列的整型
- %.nf 保留n位小数的浮点
- %m.nf

方法传递

值传递，只能从实参传给形参

第六章 异常

带参数异常处理

```
try {
    xxx
} catch (Exception e){
}
```

不同异常不同处理

```
try {   xxx }
catch (Exception e1 | Exception e2 ){ }
...
catch (Exception e){ }
```

不处理异常的try-finally

```
try {}
finally {}
```

完整异常处理块

```
try {
    xxx
} catch (Exception e){
}
finally {}
```

throw

```
throw new 异常类构造方法名(实参表)
```

例子

```
throw new Exception("xxx");
```

- throw 用于语句
- throws 用于异常类名称

自定异常类

P68

对于多个catch排列，正确的是

- 子类在先，父类在后

第七章 类与对象

定义类

```
class Circle {

    private double radius;
  private static int num; //静态字段属于整个类，能被所有对象共享使用
  private static final double PI = 3.1459;//final代表最终，不可修改的变量，也就是常量
  public Circle(){
    num++;
  }
  public Circle(double radius) throws Exception{
    if(radius<0) {throw new Exception("半径不能为负数");}
    this.radius = radius;
    num ++;
  }
}
```

类头可以选择`public`， `abstract`抽象，不能构建对象， `final`最终，不能派生子类

this

还可以用来调用本类的其他构造方法，如

```
public Circle(double radius) throws Exception{
    this();//调用Circle()构造方法
    setRadius(radius);
}
```

final

可以用来

- 声明变量
- 也可以用来声明方法

```
public final double area()  { return Math.PI * radius * radius;}
```

不能被之类重写，不能更改方法内容

- 声明类

```
public final class System{}
```

第八章 继承和多态

Extends

```
class Human1 {
}
class Student1 extends Human1{
}
```

proteceted

能被类所在包访问，但不能被其他包访问

super

子类中执行`super()`，自动调用父类的无参构造方法。或执行`super(实参表)`来调用对应的构造方法

```
super.父类字段
super.父类方法名(实参表)
```

调用对应的方法或者访问字段

类类型变量赋值

将子类对象赋值给父类声明的变量。子类对象上转为父类对象

`Human human = new Stuent()`

或者

```
Human human;
Stuent stu = new Student();
human = stu;
```

特征：

- 上转型对象不能操作子类新增的成员字段和成员方法
- 能使用父类被继承或重写的成员方法、被继承或隐藏的成员变量
- 若子类重写了父类方法，则上转型调用的必是重写后的方法（多态）
- 若子类重新定义父类同名字段，则上转型对象访问的是原有对象而不是新定义的

子类变量不能直接引用父类对象

子类对象上转时，父类变量可以直接引用子类的对象

子类变量不能直接引用父类的对象。父类也不能给子类，若使用强制类型转换，出错会引发异常

`Student4 stu = (Student4)human`

第九章 接口与包

抽象方法和抽象类

抽象类可以没有抽象方法，但是有抽象方法一定是抽象类

用`abstract`声明只有方法头没有方法体的抽象方法。抽象方法不能用private，protected或static声明

```
abstract class Shape {
 public abstract double area();
 public abstract double girth();
}

class Circle extends Shape{
    private double radius;
  public double area(){
    return Math.PI * Math.pow(radius, 2);
  }
  public double grith(){
    return 2 * Math.PI * radius;
  }
}
```

接口类型

`可选 public interface 接口名 { 常量字段和方法成员 }`

第十章

声明方法`元素类型[] 数组变量`
如`int[] nums`

创建实例`nums = new int[10]`

数组声明`int[] nums = {11,22,33}`是声明，创建，元素赋值三合一

或者`int[] nums = nwe int[]{11,22,33}`因为长度可从大括号得到，所以创建的时候不再需要单独给出

专门遍历数组和集合的for语句`for(元素类型 变量 : 数组或集合) {代码}`

返回各元素的字符串
`Arrays.toString(数组名)`

排序
`Arrays.sort()`

数组复制
`Arrays.copyOf(原数组,目标数组)`
`System.arraycopy()`

地址传递

可变字符串类StringBuilder，属于引用类型

可变数目参数方法

`类型...形参代表`

如

```
public static double sum(double...sum){
    double tot=0;
    for(double n : nums) {tot +=n; }
    return tot;
}
```

字符串类

String属于引用类型，存放的都是地址。对象是字符串常量，创建后不允许修改。

StringBuffer类

可变字符序列。增删改不需要重新构建对象。可以安全的用于多线程

StringBuilder类

单线程速度更快

第十二章 文件输入输出

java.io包中，常用类

- File 包含路径的文件类
- FileInputStream，文件字节流输入流，从指定文件读取字节
- FileOutputStream，字节输出流，把字节输出到指定的文件中
- FileReader，文件字符输入流，从文件读取一个或多个字符
- FileWriter，文件字符输出流

- RandomAccessFile，同时具备输入输出功能，能读写多种数据类型，并能指定读写位置。

序列化

将对象的所有属性值按线性顺序以字节流方式记录保存下来，以便需要的时候从中恢复，这就是序列化

`public class xxx implement Serializable`
先文件输出`FileOutputSteam`，再`ObjectOutputSteam oos;oos.writeObject()`，读取则FileInput和ObjectInput

第十三章 多线程

```
public class Animal extends Thread {
    public Animal(String name) {
        super(name);
    }
    public void run { //重写线程运行方法，否则将执行空操作
        ....
        try {            //必须处理休眠异常
            Thread.sleep((long)(Math.random()*1000));
        }
        catch (InterruptedException e) {}
    }
}
```

用Runnable对象构建Thread

已继承一父类的情况，不能再继承Thread类。采用实现Runnable接口的方法来编写相关类，再以该类对象为参数构建Thread类

```
public class Animal2 implements Runnable {
    xxxx //同上
}

//Ex2.java

public class Ex2 {
    public static void main(String[] args){
        Animal2 rabbit = new Animal2('rabbit');
        Thread t1 = new Thread(rabbig);
        t1.start();
    }
}
```

多种构造方法

- Thread() 没有参数的构造方法，自动起名Thread-n，名可用getName获取
- Thread(String name)指定名字
- Thread(Runnable target)如上，也是自动起名
- Thread(Runnable target, String name),指定线程名，同上

相关字段

setPriority 更改优先级（10-1 5是默认）高优先级执行机会多，速度快

setPriority有三个静态int优先级常量

- MAX_PRIORITY 10
- MIN_XXX 1
- NORM_XXX 5

线程生命周期

isAlive判断是否存活

其他方法

- sleep
- interrupt() 中途打断线程。线程在休眠可调用该线程的interrupt方法吵醒，使之回到就绪运行
- cuttentThread 返回正在执行的线程名
- activeCount

第十四章 集合和泛型

泛型`<E>`就是所使用的类型广泛，允许匹配各种类型

```
class Student<T> {
    private T property;
    public void setProperty(T property){
        this.property = property;
    }
}

public class Ex {
    xxxxx
        Student<String> s1 = new Student<>();
}
```

遍历集合的四个方法

1. 直接sysout co 会隐形调用toString
2. 使用Lambda
3. for(Object e : co) {sysout}
   4.如下

```
Iterator<Object> it = co.iterator();
while(it.hasNext()) {
    sysout
}
```

集合封装类Collections常用方法

- copy
- fill
- indexOfSubList
- lastIndexOfSubList
- max
- min
- sort
- reverse
- retate 轮换元素-

第十九章 数据库

放置jdbc在jdk子目录`jre\lib\ext`中，然后使用`Class.forName("com.mysql.jdbc.Driver")`（语句加载可省略）

连接

```
String url = "jdbc:mysql//localhost:3306/dbname?
useUnicode=true&characterEncoding = UTF-8";
Connection con = DriverManager.getConnection(url,
"root","root");
Statement stmt = con.createStatment(); //连接创建语句
ResultSet rs = stmt.executeQuery("select * from xxx");
while(rs.next()) {
    rs.getString();
    xxx
}
```

PreparedStatement 预编译语句

能预先编译语句，反复执行，还能带参数

常用方法

- ResultSet executeQuery() 执行语句返回结果集
- int executeUpdate 执行insert, update或者delete等DML，返回操作行数。若DDL语句返回0
- boolean execute 执行语句，如第一个结果为结果集则为true
- void setString 设置参数，如下

```
sql = "update stus set Sex = ?, Sp = ?";
prpstmt = com.prepareStatment(sql);
prpstmt.setString(1,"女");
prpstmt.executeUpdate();
```

常用类方法

String

```
String str = '';
str.length();//长度
double d = Double.parseDouble(str);//转变为对应数值类型
//字符串转为数值方法是parseXxx(String)形式，Xxx对应不同的数值类型，前缀是基本数值类型对应的类名
```

oj

二维数组

```
import java.util.Scanner;
import java.math.*;
public class Main
{

    public static void main(String[] args)
    {
        Scanner scan=new Scanner(System.in);
        int[][] temp = new int[15][15];
        int n,m,sum;
        m = scan.nextInt();
        n = scan.nextInt();
        for(int i=0;i<m;i++)
            for(int j=0;j<n;j++)
                temp[i][j] = scan.nextInt();
        for(int i=0;i<m;i++) {
            sum=0;
            for (int j = 0; j < n; j++)
                sum+= temp[i][j];
            System.out.println(sum);
        }
    }
}
```

字符统计

```
import java.util.Scanner;
import java.math.*;
public class Main
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
```

```
        String str =  new String();
        str = scan.nextLine();
        int num=0,alpha=0,space=0,other=0;
        for(int i=0;i<=str.length()-1;i++)
        {
            if(Character.isDigit(str.charAt(i)))
                num++;
            else if (Character.isLetter(str.charAt(i)))
                alpha++;
            else if(str.charAt(i) == 32)
                space++;
            else
                other++;

        }
        System.out.printf("%d %d %d
%d\n",alpha,num,space,other);
    }
}
```

插入字符串

```
import java.util.Scanner;
import java.math.*;
public class Main
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        String t =  new String();
        String s =  new String();
        StringBuffer sb = new StringBuffer();
        s = scan.nextLine();
        t = scan.nextLine();
        int pos = scan.nextInt();
```

```
        for(int i=0;i<=t.length()+s.length()-1;i++)
        {
            if(i<pos)
                sb.append(t.charAt(i));
            else if(i<pos+s.length())
                sb.append(s.charAt(i-pos));
            else
                sb.append(t.charAt(i-s.length()));
        }
        System.out.println(sb);
    }
}
```

是否合法邮箱

```
import java.util.*;

public class Main {
    public static void main(String args[]) {
        Scanner cin = new Scanner(System.in);
        String t = cin.nextLine();
        if (t.matches("[a-zA-Z0-9]+@[a-zA-Z0-9]+.com"))
{
            System.out.print("YES");
        } else {
            System.out.print("NO");
        }

        cin.close();
    }
}
```

输出公共子串

```
import java.util.*;

public class Main {
    public static void main(String args[]) {
        Scanner cin = new Scanner(System.in);
        String a = cin.next();
        String b = cin.next();

        System.out.println(LSS(a, b));
        cin.close();
    }

    public static String LSS(String s1, String s2) {
        String max = null;
        String min = null;
        max = (s1.length() > s2.length()) ? s1 : s2;
        min = max.equals(s1) ? s2 : s1;
        for (int i = 0; i < min.length(); i++) {
            for (int a = 0, b = min.length() - i; b !=
min.length() + 1; a++, b++) {
                String sub = min.substring(a, b);
                if (max.contains(sub))
                    return sub;
            }
        }
        return null;
    }
}
```

复习题

D ArrayList用法

```
import java.util.ArrayList;
import java.util.Scanner;

public class main {
    public static void main(String args[])
    {
        Scanner cin = new Scanner(System.in);
        ArrayList array = new ArrayList<>();
        int allnum = cin.nextInt();
        for(int j=0;j<allnum;j++)
        {
            array.add(cin.next());
        }

        int removenum = cin.nextInt();
        array.remove(removenum);
        int addnum = cin.nextInt();
        String tmp = cin.next();
        array.add(addnum, tmp);
        System.out.println(array.toString());
    }
}
```

数组初始化

`int [] mark = {0,0,0,0};`

String to int

`int num1 = Integer.parseInt(tmp[0]);`

String比较，单引号字符双引号字符串

`tmp[2].equals("S")`

大数对比

```
import java.util.*;
```

```
import java.math.BigInteger;

public class Main {
    public static void main(String[] args) {
        Scanner cin = new Scanner(System.in);
        while (cin.hasNext()) {
            BigInteger a = cin.nextBigInteger();
            BigInteger b = cin.nextBigInteger();
            if (a.equals(BigInteger.ZERO) &&
b.equals(BigInteger.ZERO))
                break;
            int flag = a.compareTo(b);
            if (flag == -1) {
                System.out.println("a<b");
            } else if (flag == 0) {
                System.out.println("a==b");
            } else {
                System.out.println("a>b");
            }
        }
        cin.close();
    }
}
```

char数组

```
char[] chars = scan.nextLine().toCharArray();
```