1. **求数列之和**

例：求5+55+555+...的前4项和

public class Prime1 {

public static void main(String[] args) {

long sum=0,a=5,item=a,n=4,i=1;

for(i=1;i<=n;i++) {

sum=sum+item;

item=item\*10+a;

}

System.out.println(sum);

}

}

1. **素数**

例：求101-200素数之和

public class Prime1 {

public static void main(String[] args) {

    for (int i = 101; i <=200; i++) {

                int j;

for (j = 2; j <= i/2; j++) {

    if (i % j == 0) {

break;

    }

                }

                if (j > i/2) {

    System.out.print(i + " ");

}

    }

}

}

**3.完全数：若一个数恰好等于除其本身外其它的因子之和，则这个数为“完全数”**

例：求1000以内的完全数

求public class Prime1 {

public static void main(String[] args) {

int sum,j;

for(int i=1; i<=1000; i++) {

for(j=1,sum=0; j<=i/2; j++) {

if(i%j == 0) {

sum += j;

}

}

if(sum == i) {

System.out.println(i);

}

}

}

}

1. **水仙花**

例：求100-1000的水仙花

public class Main {

public static void main(String[] args) {

flower();

}

public static void flower() {

int x = 0,y = 0,z = 0;

for(int i = 100;i<1000;i++) {

x = i/100;

y = (i%100)/10;

z = (i%100)%10;

if(x\*x\*x + y\*y\*y +z\*z\*z == i) {

System.out.print(i+" ");

}

}

}

}

**5.统计字符串中某种字符的格式**

**例：输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数。**

import java.util.\*;

public class first {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

int num1=0,num2=0,num3=0,num4=0;

//num2数字，num1英文字母，num3空格，num4其他

int sum=s.length();

char[] c=s.toCharArray();

for(int i=0;i<sum;i++)

{

if((c[i]<'z'&&c[i]>'a')||(c[i]>'A'&&c[i]<'Z'))

{

num1++;

}

else if(c[i]>'0'&&c[i]<'9')

{

num2++;

}

else if(c[i]==' ')

{

num3++;

}

else {

num4++;

}

}

System.out.println("英文字母："+num1);

System.out.println("数字："+num2);

System.out.println("空格："+num3);

System.out.println("其他："+num4);

}

}

1. **类设计**

**例：设计一个矩形类，按以下要求编写程序**

**(1) 创建一个 Rectangle 类，添加 width 和 height 两个成员变量，表示表示矩形的宽与长；**

**(2) 有参数的和无参的构造方法设计**

**(2) 在 Rectangle 中添加两种方法分别计算矩形的周长和面积**

**(3) 编写 Test 类测试 Rectangle 类，创建 1 个 Rectangle 对象 rect1 ，宽为 10， 长为 20，计算 该矩形的周长和面积并输出。**

public class Rectangle {

double width, height;

double getArea() {

double area=width \* height;

return area;

}

double getZHOU() {

double zhou=(width+height)\*2;

return zhou;

}

Rectangle()

{

width=1;

height=2;

}

Rectangle(double width,double height)

{

this.width=width;

this.height=height;

}

public static void main(String[] args) {

Rectangle rect1 = new Rectangle(10,20);

System.out.println("周长是：" + rect1.getZHOU());

System.out.println("面积是：" + rect1.getArea());

}

}

**7.抽象类的设计并被子类中继承，以及抽象方法的重写。在应用程序中，创建子类对象并调用其方法。**

1,例：设计一个动物声音“模拟器”，希望“模拟器”可以模拟许多动物的叫声，要求如下：

A、编写抽象类Animal

该抽象类有2个抽象方法：cry()和getAnimalName()，即要求实现该接口的各种具体动物类给出自己的叫声和种类名称。

B、编写模拟器类Simulator

该类有一个playSound(Animal animal)方法，该方法的参数为Animal类型，则animal可以调用实现Animal接口类重写的cry()方法播放具体动物的声音，调用重写的getAnimalName()方法显示具体动物种类的名称。

C、编写实现Animal子类的Dog类和Cat类。

public abstract class Animal {

public abstract void cry();

public abstract String getAnimalName();

}

public class Simulator {

public void playSound(Animal animal) {

System.out.print("现在播放"+animal.getAnimalName()+"类的声音:");

animal.cry();

}

}

public class Cat extends Animal {

public void cry() {

System.out.println("喵喵...喵喵");

}

public String getAnimalName() {

return "猫";

}

}

public class Dog extends Animal {

public void cry() {

System.out.println("汪汪...汪汪");

}

public String getAnimalName() {

return "狗";

}

}

public class Main{

public static void main(String args[]) {

Simulator simulator = new Simulator();

simulator.playSound(new Dog());

simulator.playSound(new Cat());

}

}

1. 一个职业抽象类，一个学生子类一个医生子类

abstract class Emp {

public abstract String position();

}

class Student extends Emp{

public String position() {

return "我是一名学生";

}

}

class Doctor extends Emp{

public String position() {

return "我是一名医生";

}

}

class Work{

public void recruitment(Emp e){//招聘员工

System.out.println("你是做什么的？");

System.out.println(e.position());

}

}

public class Test {

public static void main(String[] args) {

Work work = new Work();

Emp s = new Student();

work.recruitment((Emp)s);

Doctor d = new Doctor();

work.recruitment((Emp)d);

}

**8.接口的设计并在类中实现，以及抽象方法的重写。在应用程序中，创建子类对象并调用其方法。**

**例：按以下要求编写程序：**

**1）定义一个接口Shape，它包含一个抽象方法double getArea()。**

**2）定义一个实现接口Shape的类Circle，它包含成员1个变量double radius（表示圆的半径），一个无参构造函数Circle( )（半径为1），及一个有参构造函数Circle(double r)，用该参数r初始化圆的半径；并在类Circle中重写Shape中的方法getArea()，用于计算圆的面积。**

**3）编写 Test 类测试Circle 类，在Test 类的main方法中创建 1 个Circle对象 cir1 ，通过该对象来计算一个半径为5.0的圆的面积并输出**

interface Shape{

double getArea();

}

class Circle implements Shape{

double radius;

Circle(){

radius=1;

}

Circle(double r){

radius=r;

}

public double getArea() {

return Math.PI\*radius\*radius;

}

}

public class Text{

public static void main(String[] args) { // 声明对象和创建对象

Circle cir1 = new Circle(5.0);

System.out.println("Area：" + cir1.getArea());

}

}

**10.定义接口 Student，该接口中有一个无参、无返回值的抽象方法 prtMsg()；定义类 College，包括的私有**

**属性有学生学号 id 和姓名 name，包括一个初始化学生学号和姓名的构造方法，并且实现 Student 接口。**

interface Student{

public abstract void prtMsg();

}

class College implements Student{

private String id,name;

public College(String id,String name){

this.id = id;

this.name = name;

}

public void prtMsg() {

System.out.println(id + name);

}

}