**作业**

任务一：抽象类

**员工抽象类**

包含3个属性：工号，姓名，工资。

一个抽象方法：工作描述（例如：不同员工有不同的工作内容）

**项目经理子类**

另外还有一个奖金属性

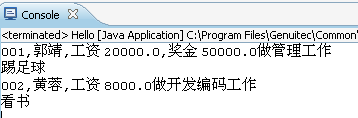
普通方法：踢足球

**程序员子类**

普通方法：看书

**测试类：**

测试经理和员工



package day10.homeWork;

abstract class Employee {

int no;

String name;

double salary;

public Employee(int no, String name, double salary) {

this.no = no;

this.name = name;

this.salary = salary;

}

public abstract void show();

}

class JingLi extends Employee {

double JiangJin;

public JingLi(int no, String name, double salary, double JiangJin) {

super(no, name, salary);

this.JiangJin = JiangJin;

// TODO Auto-generated constructor stub

}

public void TiQiu() {

System.out.println("踢足球");

}

@Override

public void show() {

// TODO Auto-generated method stub

System.out.println(no + "," + name + ",工资" + salary + ",奖金" + JiangJin + "做管理工作");

}

}

class ChengXuYuan extends Employee{

public ChengXuYuan(int no, String name, double salary) {

super(no, name, salary);

}

@Override

public void show() {

System.out.println(no + "," + name + ",工资" + salary + ",做开发编码工作");

}

public void KanShu() {

System.out.println("看书");

}

}

public class ZuoYe1 {

public static void main(String[] args) {

JingLi j = new JingLi(1, "张五", 4000, 10000);

ChengXuYuan c = new ChengXuYuan(2, "张三", 4500);

j.show();

j.TiQiu();

c.show();

c.KanShu();

}

}

任务二：接口

飞行的接口：

IFly

飞机类：

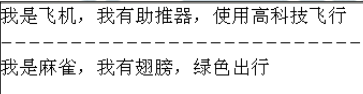
Plane

麻雀类：

Spadger

测试类：

FlyTest



package day10.homeWork;

interface IFly{

public abstract void fly();

}

class Plane implements IFly{

@Override

public void fly() {

// TODO Auto-generated method stub

System.out.println("我是飞机，我有助推器，使用高科技飞行");

}

}

class Spadger implements IFly{

@Override

public void fly() {

// TODO Auto-generated method stub

System.out.println("我是麻雀，我有翅膀，绿色出行");

}

}

public class ZuoYe2 {

public static void main(String[] args) {

Plane p = new Plane();

p.fly();

Spadger s = new Spadger();

s.fly();

}

}

任务三：

## 接口的定义和使用

### 任务描述

定义一个接口Shape，该接口有一个计算面积的方法area()，该方法返回一个double类型的返回值。

定义一个三角形类(Triangle)，该类实现了Shape接口。该类有两个属性height(三角形的高)，和length(三角形的底边长)，该类实现了area()方法，以计算三角形的面积。

定义一个圆形类(Circle)，该类实现了Shape接口。该类有一个属性radius(圆的半径)，该类实现了area()方法，以计算圆形的面积。

定义Main类，在其main方法中，创建一个三角形类和一个圆形类，分别调用它们的area()方法以计算其面积。。

package day10.homeWork;

interface Shape{

double area();

}

class Triangle implements Shape{

double height;

double length;

public Triangle(double height, double length) {

super();

this.height = height;

this.length = length;

}

@Override

public double area() {

// TODO Auto-generated method stub

return height\*length/2;

}

}

class Circle implements Shape{

double radius;

public Circle(double radius) {

super();

this.radius = radius;

}

@Override

public double area() {

// TODO Auto-generated method stub

return Math.PI\*radius\*radius;

}

}

public class ZuoYe3 {

public static void main(String[] args) {

Triangle t = new Triangle(3, 4);

Circle c = new Circle(5);

System.out.println("三角形的面积为"+t.area());

System.out.println("圆的面积为"+c.area());

}

}

任务四：

编写程序描述兔子和青蛙

需求说明：使用面向对象的思想，设计自定义类描述兔子和青蛙。

实现思路及关键代码

1. 分析兔子和青蛙的共性
2. 根据共性，定义抽象的动物类

属性：名字、颜色、类别（哺乳类、非哺乳类）

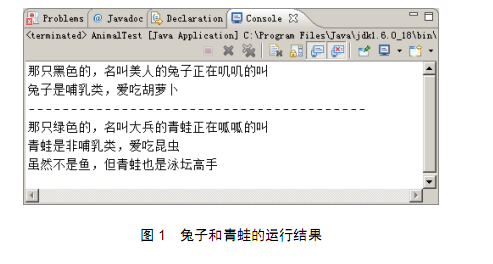
方法：吃饭，发出叫声

1. 根据青蛙会游泳 抽象游泳的接口

方法：游泳

1. 定义兔子继承动物类，青蛙继承动物同时实现游泳接口

程序运行结果如图所示。



package day10.homeWork;

import com.sun.xml.internal.ws.addressing.W3CAddressingConstants;

abstract class Animal1{

private String name;

private String color;

private boolean type;

public abstract void eat();

public abstract void shout();

public Animal1(String name, String color, boolean type) {

super();

this.name = name;

this.color = color;

this.type = type;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

public boolean isType() {

return type;

}

public void setType(boolean type) {

this.type = type;

}

}

interface Swim{

void swim();

}

class Rabbit extends Animal1{

public Rabbit(String name, String color, boolean type) {

super(name, color, type);

// TODO Auto-generated constructor stub

}

@Override

public void eat() {

// TODO Auto-generated method stub

System.out.println("兔子是"+(isType()?"哺乳类":"非哺乳类")+"爱吃胡萝卜");

}

@Override

public void shout() {

System.out.println("那只"+getColor()+"的，名叫"+getName()+"的兔子正在叽叽的叫");

}

}

class Frog extends Animal1 implements Swim{

public Frog(String name, String color, boolean type) {

super(name, color, type);

// TODO Auto-generated constructor stub

}

@Override

public void eat() {

// TODO Auto-generated method stub

System.out.println("青蛙是"+(isType()?"哺乳类":"非哺乳类")+"爱吃虫子");

}

@Override

public void shout() {

System.out.println("那只"+getColor()+"的，名叫"+getName()+"的兔子正在叽叽的叫");

}

@Override

public void swim() {

System.out.println("虽然不是鱼，但青蛙也是泳坛泳坛高手");

}

}

public class ZuoYe4 {

public static void main(String[] args) {

Rabbit r = new Rabbit("小红", "白色", true);

Frog f = new Frog("小跳蛙","原谅色",false);

r.eat();

r.shout();

f.eat();

f.shout();

f.swim();

}

}

**选做题：**

## 抽象类及其实现类的使用

### 任务描述

编写抽象员工类及其子类

编写抽象类Employee。该类有如下属性：

id(编号int型)

sex(性别 int型)

name(姓名String型)

duty(职务String型)

salary(薪水double型)

holidays(请假天数 int型)

该类有如下方法：

display()，无返回值，该方法能打印员工的姓名、性别以及职务三个属性

该类还有一个抽象方法

getSubsidy(int day) 返回值是int型。该方法能计算员工正常出勤补助工资。

编写普通员工类(NormalEmployee)，该类是Employee的直接子类，并实现了getSubsidy (int day)方法。普通员工补助工资的计算方法是：如果请假天数小于等于3，则补助工资为20× 出勤天数；如果请假天数超过3天，则补助工资为15 × 出勤天数。参数day表示请假天数。

编写董事类(Director)，该类也是Employee的直接子类，并实现了getSubsidy (int day)方法。董事补助工资的计算方法是：如果请假天数小于等于3，则补助工资为30× 出勤天数；如果请假天数超过3天，则补助工资为20 × 出勤天数。参数day表示请假天数。

编写Main类，在该类的main方法中，分别创建一个NormalEmployee和一个Director类的对象，并由用户输入其请假天数来计算他们正常出勤补助工资。

package day10.homeWork;

import java.util.Scanner;

abstract class Employee1{

int id;

int sex;

String name;

String duty;

double salary;

int holidays;

public Employee1(int id, int sex, String name, String duty, double salary, int holidays) {

super();

this.id = id;

this.sex = sex;

this.name = name;

this.duty = duty;

this.salary = salary;

this.holidays = holidays;

}

public void display() {

String s = "";

if(sex ==1) {

s="女";

}else if(sex==0) {

s="男";

}else {

s = "未知";

}

System.out.println("姓名："+name+"性别："+s+"职务："+duty+"工资："+salary);

}

public abstract int getSubsidy(int day);

}

class NorMalEmployee extends Employee1{

public NorMalEmployee(int id, int sex, String name, String duty, double salary, int holidays) {

super(id, sex, name, duty, salary, holidays);

// TODO Auto-generated constructor stub

}

@Override

public int getSubsidy(int day) {

// TODO Auto-generated method stub

holidays = day;

if(day<=3) {

salary += 20\*(30-day);

return (int)salary;

}else {

salary += 15\*(30-day);

return (int)salary;

}

}

}

class Director extends Employee1{

public Director(int id, int sex, String name, String duty, double salary, int holidays) {

super(id, sex, name, duty, salary, holidays);

// TODO Auto-generated constructor stub

}

@Override

public int getSubsidy(int day) {

holidays = day;

if(day<=3) {

salary += 30\*(30-day);

return (int)salary;

}else {

salary += 20\*(30-day);

return (int)salary;

}

}

}

public class ZuoYe5 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

NorMalEmployee n = new NorMalEmployee(1, 1, "LY", "打代码", 4500, 1);

Director d = new Director(1, 1, "HTL", "监督", 5000, 1);

System.out.print("输入普通员工请假天数：");

n.getSubsidy(sc.nextInt());

n.display();

System.out.print("输入董事长请假天数：");

d.getSubsidy(sc.nextInt());

d.display();

}

}