# 第5章（上）作业

（共100分）

一、程序阅读题，写出程序运行的输出结果，并进行解释（每题15分，共 60 分）

第1题

class worker {

int num;

worker() {

num += 5;

}

worker(int n) {

num = n;

}

void workShow() {

System.out.println("Inside worker method, num=" + num);

}

}

class programmer extends worker {

int num = 1;

programmer(int n) {

num += n;

}

void workShow() {

System.out.println("Inside programmer method, num=" + num + ", super.num=" + super.num);

}

}

public class Dispatch {

public static void main(String args[]) {

worker a = new programmer(10);

a.workShow();

}

}

Inside programmer method, num=11, super.num=5

解释：子类的num与workshow将父类的同名字段和方法进行覆盖，当创建子类对象时，默认调用父类的无参构造方法，父类的num由jvm赋值为0后又在构造方法中加5，故super.num为5，子类构造方法传入参数为10，故子类的num为输出为11

第2题

class Fu {

boolean show(char a) {

System.*out*.println(a);

return true;

}

}

class Demo extends Fu {

public static void main(String[] args) {

int i = 0;

Fu f = new Fu();

Fu f2 = new Demo();

for (f.show('A'); i < 2 || f2.show('B'); f.show('C')) {

i++;

f2.show('D');

}

}

@Override

boolean show(char a) {

System.*out*.println(a);

return false;

}

}

A

D

C

D

C

B

解释：f 指向父类对象,f.show返回true,f2实际指向子类对象,f2.show返回false.循环开始是打印A,由于使用的是短路或运算符,前两次均不会打印B,在i增加到二时会执行一次f2.show,打印B,返回false,循环结束.

第3题: 读程序题与填空

class Animal {

String name;

Animal(String name) {

this.name = name;

}

public void enjoy() {

System.out.println("动物" + name + "的叫声……");

}

}

class Cat extends Animal {

String eyesColor;

Cat(String n, String c) {

super(n);

eyesColor = c;

}

@Override

public void enjoy() {

System.out.println(eyesColor + "的猫" + name + "喵喵喵喵……");

}

}

class Dog extends Animal {

String furColor;

Dog(String n, String c) {

super(n);

furColor = c;

}

@Override

public void enjoy() {

System.out.println(furColor + "的狗" + name + "汪汪汪汪……");

}

}

class Bird extends Animal {

Bird() {

super("Poli");

}

@Override

public void enjoy() {

System.out.println("鸟儿" + name + "叽叽喳喳……");

}

}

class AnimalCalls {

private Animal pet;

AnimalCalls(Animal pet) {

this.pet = pet;

}

public void myPetEnjoy() {

pet.enjoy();

}

}

public class TestCall {

public static void main(String args[]) {

Cat c = new Cat("Tom", "蓝色");

Dog d = new Dog("Ben", "黑色");

Bird b = new Bird();

AnimalCalls c1 = new AnimalCalls(c);

AnimalCalls c2 = new AnimalCalls(d);

AnimalCalls c3 = new AnimalCalls(b);

c1.myPetEnjoy();

c2.myPetEnjoy();

c3.myPetEnjoy();

}

}

(1)请给出程序运行结果，并解释结果（结果5分，解释5分）

蓝色的猫Tom喵喵喵喵……

黑色的狗Ben汪汪汪汪……

鸟儿Poli叽叽喳喳……

在 main 方法中，首先创建了一个名为 c 的猫对象，名字为 "Tom"，眼睛颜色为 "蓝色"，然后创建了一个名为 d 的狗对象，名字为 "Ben"，毛发颜色为 "黑色"，最后创建了一个名为 b 的鸟对象，它的名字在构造函数中被设置为 "Poli"。

接下来，分别创建了三个 AnimalCalls 对象 c1、c2 和 c3，每个对象分别接收了不同的动物作为参数，即猫、狗和鸟。

在 main 方法中，依次调用了 c1.myPetEnjoy()、c2.myPetEnjoy() 和 c3.myPetEnjoy() 方法，分别输出了每个动物的特定叫声。

对于猫 c1，它的 enjoy 方法被重写，在输出时使用了猫的眼睛颜色和名字，因此输出了 "蓝色的猫Tom喵喵喵喵……"。狗c2同理

对于鸟 c3，虽然它的构造函数中指定了名字为 "Poli"，但由于没有指定眼睛颜色或毛发颜色，所以输出了默认的 "鸟儿Poli叽叽喳喳……"。

(2) 方法

AnimalCalls(Animal pet) {

this.pet = pet;

}

是 AnimalCalls 类的 构造 方法。（2分）

(3)该程序中， Animal 为超类， cat dog bird 为子类。（2分）

(4) c2.myPetEnjoy()调用的是 Dog 类的enjoy方法（1分）

4. 读程序题与填空：先补齐程序中的下划线处，再写出程序在console端的输出结果，并解释。

abstract class Person {

public Person(String n) {

name = n;

}

public abstract String getDescription();

public String getName() {

return name;

}

private String name;

}

class Employee extends Person {

public Employee(String n, double s) {

super( n );

salary = s ;

}

public double getSalary() {

return salary;

}

@Override

public String getDescription() {

return String.format("an employee with a salary of $%.2f", salary);

}

public void raiseSalary(double byPercent) {

double raise = salary \* byPercent / 100;

salary += raise;

}

private double salary;

}

class Student extends Person {

public Student(String n, String m) {

super(n);

major = m;

}

@Override

public String getDescription() {

return "a student majoring in " + major;

}

private String major;

}

public class PersonTest {

public static void main(String[] args) {

Person[] people = new Person[2];

people[0] = new Employee("Harry Hacker", 50000);

( (Employee) people[0]).raiseSalary(10);

people[1] = new Student("Maria Morris", "computer science");

for (Person p : people ) {

System.out.println(p.getName() + ", " + p.getDescription());

}

}

}

填空部分每空1分，共5分

输出结果（5分）：

Harry Hacker, an employee with a salary of $55000.00

Maria Morris, a student majoring in computer science

解释（5分）

定义了一个person类型的数组,第一个元素是Employee类型的子类对象,第二个是student类型的子类对象,分别进行初始化,由于raiseSalary是子类特有的方法,需要对people[0]进行向下转型,且people[0]实际指向的就是Employee类型对象,故转型成功,最后使用循环依次输出两个对象的name和描述由于getName()和getDescription()均是父类中已经定义的方法,且子类均已经实现,直接使用对象名访问即可.

二、编程题（每题20分，共 40 分）

1、先定义一个接口Perarea，它有两个方法，一个是计算图形的面积，另一个是计算图形周长：

double get\_area();

double get\_perimeter();

再实现一个长方形和一个圆形的类，它们都实现了这个接口；再分别构造长为10，宽为5的长方形，以及半径为5的圆，然后通过接口分别打印出这两个图形的面积和周长。要写出输出结果。

interface Perarea{

  double get\_area();

  double get\_perimeter();

}

class rectangle implements Perarea{

  double lon;

  double width;

  rectangle( double a,double b){

    lon=a;

    width=b;

  }

 public double get\_area(){

    return lon\*width;

  }

 public double get\_perimeter(){

    return 2\*(lon+width);

  }

}

class round implements Perarea{

  double r;

  round(double a){

    r=a;

  }

 public double get\_area(){

    return Math.PI\*r\*r;

  }

 public double get\_perimeter(){

    return 2\*Math.PI\*r;

  }

}

class test{

  public static void main(String[] args) {

    rectangle changfangxing=new rectangle(10, 5);

    round yuan=new round(5);

    System.out.println("长方形的面积：" +changfangxing.get\_area());

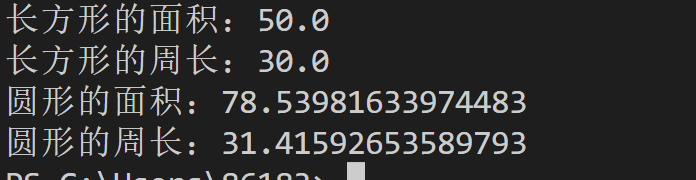
    System.out.println("长方形的周长："+changfangxing.get\_perimeter());

    System.out.println("圆形的面积：" +yuan.get\_area());

    System.out.println("圆形的周长："+yuan.get\_perimeter());

  }

}



2、本程序的数值默认都是double型。

（1）编写一个矩形类Rect，包含成员：

1. 矩形的宽width；矩形的高height。
2. 两个构造方法：一个带有两个参数，用于width和height属性初始化；一个不带参数，将矩形初始化为宽和高都为10。
3. 两个实例方法：一个求矩形面积area()，另一个求矩形周长perimeter()。

（2）通过继承Rect类，编写一个具有确定位置的矩形类PlainRect，其确定位置用矩形的左上角坐标来标识。添加成员：

1. 两个属性（矩形左上角坐标startX和startY）。
2. 两个构造方法：一个带4个参数，用于对startX、startY、width和height属性初始化；一个不带参数，将矩形初始化为左上角坐标、长和宽都为0的矩形。
3. 一个实例方法：判断某个点是否在矩形内部isInside(double x,double y)。如在矩形内，返回true, 否则，返回false。 （提示：点在矩形内需满足条件：x >= startX && x <= (startX+width) && y < startY && y >= (startY-height) ）

（3）编写PlainRect类的测试程序

1. 创建一个左上角坐标为（10，10），长为20，宽为10的PlainRect对象；计算并打印输出矩形的面积和周长；
2. 判断点(25.5，13)是否在矩形内，并打印输出相关信息。

class Rect {

  double width;

  double height;

  Rect(double width, double height) {

      this.width = width;

      this.height = height;

  }

  Rect() {

      this.width = 10;

      this.height = 10;

  }

  double area() {

      return width \* height;

  }

  double perimeter() {

      return 2 \* (width + height);

  }

}

class PlainRect extends Rect {

  double startX;

  double startY;

  PlainRect(double startX, double startY, double width, double height) {

      super(width, height);

      this.startX = startX;

      this.startY = startY;

  }

  PlainRect() {

      super(0, 0);

      this.startX = 0;

      this.startY = 0;

  }

  boolean isInside(double x, double y) {

      return x >= startX && x <= (startX + width) && y >= startY && y <= (startY - height);

  }

}

 class TestPlainRect {

  public static void main(String[] args) {

      PlainRect rect = new PlainRect(10, 10, 20, 10);

      System.out.println("矩形的面积：" + rect.area());

      System.out.println("矩形的周长：" + rect.perimeter());

      double x = 25.5;

      double y = 13;

      if (rect.isInside(x, y)) {

          System.out.println("(" + x + ", " + y + ") 在矩形内部。");

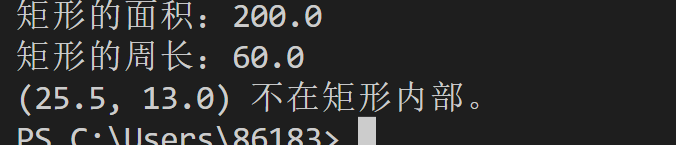
      } else {

          System.out.println("(" + x + ", " + y + ") 不在矩形内部。");

      }

  }

}



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

提交方式：

1. 上传到[坚果云链接中](https://send2me.cn/AQeFyPti/REiEsoPLV8hy3Q)（上传时请勿挂vpn或其他梯子，否则可能上传失败）
2. 文件名严格按照：学号-姓名-第几次作业.docx(或pdf)的方式命名
3. 每次作业在截止日期之前，若发现提交内容有误，可以再提交一次。但每次作业最多只能提交两次。第二次提交的文件后面加上“（2）”的字样。
4. 作业提交形式：单一的word或pdf文档。不要上传Eclipse、VScode、IDEA项目文件，不要压缩打包，不要以文件夹形式上传。
5. 截止时间：9月24日（周日）晚23:59。