《Java 程序设计》实验报告三

学生姓名：李季鸿　　班级：2021 级计本（3）班　　学号：20213002624

实验地点：理工楼 601 室　　指导教师：张春元

实验日期：2023-03-06　　共 2 学时

实验环境：Win10+JDK1.8+集成化开发工具（idea64）

1. 实验目的

学习分支语句的应用、类与对象，掌握集成化开发工具的使用。

2. 实验内容

（1） 用集成化开发工具完成实验教材 P18 实验 2 内容。

（2） 用集成化开发工具完成实验教材 P23 实验 1 内容，请把 void setPower(int p)改成 void setPower(int power)。

（3） 用集成化开发工具完成实验教材 P25 实验 2 内容。

（4） 用集成化开发工具编写如下程序：采用割圆术计算圆周率，即对一个圆先割出一个正方形，再割出 4 个等腰三角形，再割出 8 个等腰三角形，后面依此类推，把这些割出的图形面积相加即为圆的面积，然后除以圆的半径的平方，最终得到圆周率。

3. 实验过程

报告撰写具体要求：截屏显示或直接写出上述实验源码和运行结果，并完成各实验后的练习（不用抄题，直接写答案即可）。

实验内容（1）：

```
1.  /**
2.   * \* Created with IntelliJ IDEA.
3.   * \* User: lijihong
4.   * \* Date: 2023-03-05
5.   * \* Time: 23:16
6.   * \
7.   */
8.  import java.util.Scanner;
9.  import java.util.Random;
10.
11. public class _3_GuessNumber {
12.     public static void main(String[] args) {
13.         Scanner reader = new Scanner(System.in);
14.         Random random = new Random();
15.         System.out.println("给你一个 1~100 之间的整数，请猜测这个数");
16.         int realNumber = random.nextInt(100) + 1;  //random.nextInt(100)是
    [0,1000)中的随机整数
17.         int yourGuess = 0;
```

```
18.         System.out.print("输入您的猜测:");
19.         yourGuess = reader.nextInt();
20.         //当 yourGuess 的数字与随机数不同时，进入 while 循环，直到两数字大小一样
21.         while(yourGuess != realNumber){ // 循环条件
22.             if(yourGuess > realNumber){ //猜大了的条件代码
23.                 System.out.print("猜大了，再输入您的猜测:");
24.                 yourGuess = reader.nextInt();
25.             }
26.             else if(yourGuess < realNumber){ //猜小了的条件代码
27.                 System.out.print("猜小了，再输入您的猜测:");
28.                 yourGuess = reader.nextInt();
29.             }
30.         }
31.         //输出正确数字
32.         System.out.println("猜对了！ ");
33.     }
34. }
35.
```

```
给你一个1~100之间的整数，请猜测这个数
输入您的猜测:50
猜小了，再输入您的猜测:75
猜小了，再输入您的猜测:85
猜小了，再输入您的猜测:95
猜小了，再输入您的猜测:97
猜小了，再输入您的猜测:99
猜对了!
```

实验后的练习题：

（1）答：不可以，整体的循环条件应该是猜不对就一直循环，换了之后就只判断大于的情况，如果猜小了（或等于）就跳出循环，不符合题意，无法找出真正的数。

（2）答：因为不满足 while 语句的循环条件，这是 2 一定是猜对了，所以不用继续循环下去，因此可以跳出循环输出：“猜对了！ 。放在 while 循环体内部是不合理的。

实验内容（2）：

_3_Vehicle :

```
1.  /** /**
```

```
2.   * \* Created with IntelliJ IDEA.
3.   * \* User: lijihong
4.   * \* Date: 2023-03-06
5.   * \* Time: 0:26
6.   * \
7.   */
8.  public class _3_MainClass {
9.      public static void main(String args[]) {
10.         _3_TV haierTV = new _3_TV();
11.         haierTV.setChannel(5); //haierTV 调用 setChannel(int m),并向参数 m 传递 5
12.         System.out.println("haierTV 的频道是 " + haierTV.getChannel());
13.         _3_Family zhangSanFamily = new _3_Family();
14.         zhangSanFamily.buyTV(haierTV);//zhangSanFamily 调用 void buyTV(_3_TV tv)方
    法,并将 haierTV 传递给参数 TV
15.         System.out.println("zhangSanFamily 开始看电视节目");
16.         zhangSanFamily.seeTV();
17.         int m = 2;
18.         System.out.println("hangSanFamily 将电视更换到" + m + "频 道");
19.         zhangSanFamily.remoteControl(m);
20.         System.out.println("haierTV 的频道是 " + haierTV.getChannel());
21.         System.out.println("hangSanFamily 再看电视节目");
22.         zhangSanFamily.seeTV();
23.     }
24.
25. }


1.
2.   * \* Created with IntelliJ IDEA.
3.   * \* User: lijihong
4.   * \* Date: 2023-03-05
5.   * \* Time: 23:41
6.   * \
7.   */
8.  public class _3_Vehicle {
9.      double speed; //声明 double 型变量 speed,刻画速度
10.     int power; //声明 int 型变量 power,刻画功率
11.
12.     void speedUp(int s) {
13.         speed = speed + s; //将 参数s 的值与成员变量speed 的和赋值给成员变
    量 speed
14.     }
15.
16.     void speedDown(int d) {
```

```
        speed = speed - d; //将成 员变量 speed 与参数 d 的差赋值给成员变量 speed
    }

    void setPower(int power) {
        this.power = power;//将参数 p 的值赋值给成员变量 power
    }

    int getPower() {
        return power; //返回成员 变量 power 的值
    }

    double getSpeed() {
        return speed;
    }
}
```

_3_User :

```
/**
 * \* Created with IntelliJ IDEA.
 * \* User: lijihong
 * \* Date: 2023-03-06
 * \* Time: 0:00
 * \
 */

public class _3_User {
    public static void main(String args[]) {
        _3_Vehicle car1,car2;
        car1 = new _3_Vehicle(); //使用 new 运算符和默认的构造方法创建对象 car1
        car2 = new _3_Vehicle(); //使用 new 运算符和默认的构造方法创建对象 car2
        car1.setPower(128);
        car2.setPower(76);
        System.out.println("car1 的功率是： " + car1.getPower());
        System.out.println("car2 的功率是： " + car2.getPower());
        car1.speedUp(80); //car1 调用 speedUp 方法将自己的 speed 的值增加 80
        car2.speedUp(100); //car2 调用 speedUp 方法将自己的 speed 的值增加 100
```

```
20.         System.out.println("car1 目前的速度： " + car1.getSpeed());
21.         System.out.println("car2 目前的速度： " + car2.getSpeed());
22.         car1.speedDown(10);
23.         car2.speedDown(20);
24.         System.out.println("car1 目前的速度： " + car1.getSpeed());
25.         System.out.println("car2 目前的速度： " + car2.getSpeed());
26.     }
27. }
```

```
car1 的功率是：128
car2 的功率是：76
car1 目前的速度：80.0
car2 目前的速度：100.0
car1 目前的速度：70.0
car2 目前的速度：80.0
```

实验后的练习题：

（1）答：

改进：

```
3 个用法  新 *
void speedUp(int s) {
    if(speed + s <= 200){ //加速不得超过200
        speed = speed + s; //将参数s的值与成员变量speed的和赋值给成员变量speed
    }
}
```

20213002624李季鸿

测试：

```
car1.speedUp( s: 80); //car1 调用speedUp 方法将自己的 speed 的值增加 80
car2.speedUp( s: 100); //car2 调用 speedUp 方法将自己的 speed 的值增加 100

System.out.println("car1 目前的速度：" + car1.getSpeed());
System.out.println("car2 目前的速度：" + car2.getSpeed());
car1.speedUp( s: 500); //car1 调用 speedUp 方法将自己的 speed 的值增加 500 观察效果
System.out.println("car1 加了500之后， 目前的速度：" + car1.getSpeed());
```

测试效果：

```
car1 的功率是：128
car2 的功率是：76
car1 目前的速度：80.0
car2 目前的速度：100.0
car1 加了500之后， 目前的速度：80.0
car1 目前的速度：70.0
car2 目前的速度：80.0
```

（2）答：

改进：

```
2 个用法  新 *
void speedDown(int d) {
    if(speed - d >= 0){ //减速不得小于0
        speed = speed - d; //将成 员变量speed与参数d的差赋值给成员变量speed
    }
}
```


测试：

```
car1.speedDown( d: 10);
car2.speedDown( d: 20);
System.out.println("car1 目前的速度：" + car1.getSpeed());
System.out.println("car2 目前的速度：" + car2.getSpeed());

car2.speedDown( d: 1000);
System.out.println("car2 减了1000之后， 目前的速度：" + car2.getSpeed());
```

测试效果：

```
car1 的功率是：128
car2 的功率是：76
car1 目前的速度：80.0
car2 目前的速度：100.0
car1 加了500之后， 目前的速度：80.0
car1 目前的速度：70.0
car2 目前的速度：80.0
car2 减了1000之后， 目前的速度：80.0
```

（3）答：

改进：

```
2 个用法  新 *
void brake() {
    speed = 0;//将速度减为0
}
```


测试：

```
car2.speedDown( d: 1000);
System.out.println("car2 减了1000之后，  目前的速度：" + car2.getSpeed());
car1.brake(); //将car1的速度减为0
car2.brake(); //将car2的速度减为0
System.out.println("car1刹车后速度应该为0 目前的速度：" + car1.getSpeed());
System.out.println("car2刹车后速度应该为0 目前的速度：" + car2.getSpeed());
```

测试效果：

```
car1 的功率是：128
car2 的功率是：76
car1 目前的速度：80.0
car2 目前的速度：100.0
car1 加了500之后，  目前的速度：80.0
car1 目前的速度：70.0
car2 目前的速度：80.0
car2 减了1000之后，  目前的速度：80.0
car1刹车后速度应该为0 目前的速度：0.0
car2刹车后速度应该为0 目前的速度：0.0
```

实验内容（3）：

_3_TV.java:

```
1.  /**
2.   * \* Created with IntelliJ IDEA.
3.   * \* User: lijihong
4.   * \* Date: 2023-03-06
5.   * \* Time: 0:24
6.   * \
7.   */
8.  public class _3_TV {
9.      int channel; //电视频道
10.     void setChannel(int m) {
11.         if(m >= 1){
```

```
            channel = m;
        }
    }
    int getChannel(){
        return channel;
    }
    void showProgram(){
        switch(channel) {
            case 1 : System.out.println("综合频道");
                break;
            case 2 : System.out.println("经济频道");
                break;
            case 3 : System.out.println("文艺频道");
                break;
            case 4 : System.out.println("国际频道");
                break;
            case 5 : System.out.println("体育频道");
                break;
            default : System.out.println("不能收看" + channel + " 频道");
        }
    }
}
```

_3_Family.java:

```
/**
 * \* Created with IntelliJ IDEA.
 * \* User: lijihong
 * \* Date: 2023-03-06
 * \* Time: 0:26
 * \
 */
public class _3_Family {
    _3_TV homeTV;
    void buyTV(_3_TV tv) {
        homeTV = tv; //将参数 tv 赋值给 homeTV
    }
    void remoteControl(int m) {
        homeTV.setChannel(m);
    }
    void seeTV() {
        homeTV.showProgram(); //homeTV 调用 showProgram() 方法
    }
```

```
19.
20. }
```

_3_MainClass.java:

```
/**
 * \* Created with IntelliJ IDEA.
 * \* User: lijihong
 * \* Date: 2023-03-06
 * \* Time: 0:26
 * \
 */
public class _3_MainClass {
    public static void main(String args[]) {
        _3_TV haierTV = new _3_TV();
        haierTV.setChannel(5); //haierTV 调用 setChannel(int m),并向参数 m 传递 5
        System.out.println("haierTV 的频道是 " + haierTV.getChannel());
        _3_Family zhangSanFamily = new _3_Family();
        zhangSanFamily.buyTV(haierTV);//zhangSanFamily 调用 void buyTV(_3_TV tv)方法,并将 haierTV 传递给参数 TV
        System.out.println("zhangSanFamily 开始看电视节目");
        zhangSanFamily.seeTV();
        int m = 2;
        System.out.println("hangSanFamily 将电视更换到" + m + "频 道");
        zhangSanFamily.remoteControl(m);
        System.out.println("haierTV 的频道是 " + haierTV.getChannel());
        System.out.println("hangSanFamily 再看电视节目");
        zhangSanFamily.seeTV();
    }

}
```

运行结果：

```
haierTV 的频道是 5
zhangSanFamily 开始看电视节目
体育频道
hangSanFamily 将电视更换到2频 道
haierTV 的频道是 2
hangSanFamily 再看电视节目
经济频道
```

实验后的练习题：

（1）答：可以通过编译，这里默认给了 m 一个值：0， 程序输出：

```
haierTV 的频道是 0
zhangSanFamily 开始看电视节目
不能收看0 频道
hangSanFamily 将电视更换到2频 道
haierTV 的频道是 2
hangSanFamily 再看电视节目
经济频道
```

不设置初值，默认形参为 0；

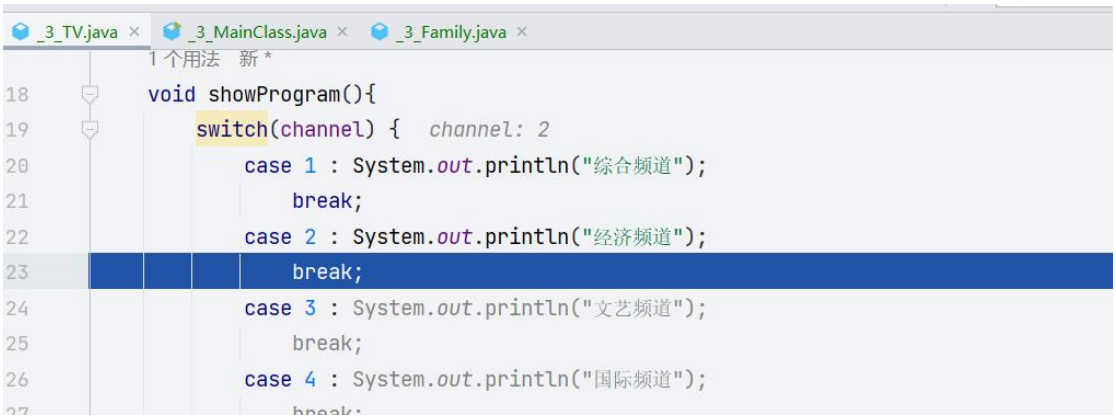
（2）答：添加后：

```
_3_Family lisiFamily = new _3_Family();
lisiFamily.buyTV(haierTV);
lisiFamily.seeTV();
```

运行效果：

```
haierTV 的频道是 5
zhangSanFamily 开始看电视节目
体育频道
hangSanFamily 将电视更换到2频 道
haierTV 的频道是 2
hangSanFamily 再看电视节目
经济频道
经济频道
```

解释效果：上面代码将 haierTV 对象的 m 已经修改为 2，将它作为参数传递给新的 lisiFamily 对象时，属性不会改变。因此，lisiFamily.seeTV 是经济频道。



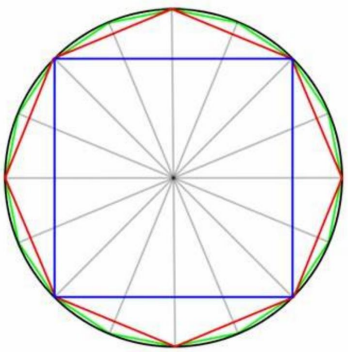
通过 DEBUG 验证了我的这一个想法，这里直接步入 case 2 的代码里面，此处默认 channel 是 2。

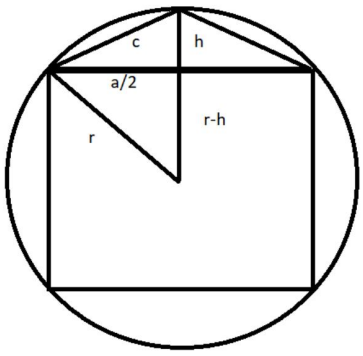
实验内容（4）：

在程序中，我们使用 for 循环来计算割圆次数。在每一次循环中， 我们先计算等腰角形的底边长和高，然后计算出该等腰二角形的面积，并将其累加到所有图形面积之和中。最后，我们将所有图形面积之和除以正方形的面积(即圆的半径的平方) ,得到圆周率的近似值。

代码如下所示：

```
1. public class PiCalculation {
2.     public static void main(String[] args) {
3.         int n = 100000; // 定义割圆次数
4.         double r = 1.0; // 圆的半径
5.         double s = r * r; // 正方形面积
6.         double sum = s; // 所有图形面积之和
7.         for (int i = 1; i <= n; i++) {
8.             double x = r * Math.sin(Math.PI / (2 * i)); // 等腰三角形底边长的一半
9.             double y = r * Math.cos(Math.PI / (2 * i)); // 等腰三角形的高
10.            double area = 4 * x * y; // 每个等腰三角形的面积
11.            sum += area; // 所有图形面积之和
12.        }
13.        double pi = sum / (r * r); // 计算圆周率
14.        System.out.println("Pi is approximately " + pi);
15.    }
16.}
```





4. 实验总结

写出实验中的心得体会（对第 3 章理论课重点简述）。

心得体会：

Java 提供了许多算术运算符：

赋值运算符以符号“=”表示，它是一个二元运算符（对两个操作数作处理），其功能是将右方操作数所含的赋值给左方的操作数。由于赋值运算符“=”处理时

会先取得右方表达式处理后的结果，因此一个表达式中若含有两个以上的“=”运算符，会从最右方的“=”开始处理。

Java 中的算术运算符主要有+（加）、-（减）、*（乘）、/（除）、%（余数），它们都是二元运算符。

自增、自减运算符是单目运算符，可以放在操作元之前，也可以放在操作元之后。操作元必须是一个整型或浮点型变量。自增、自减运算符的作用是使变量的值增 1 或减 1.放在操作元前面的自增、自减运算符，会先将变量的值加 1（减 1），然后再使用该变量参与表达式的运算。放在操作元后面的自增、自减运算符，会先使变量参与表达式的运算符，然后再使该变量加 1（减 1）。

比较运算符属于二元运算符，用于程序中的变量之间、变量和白变量之间以及其他类型的信息之间的比较。比较运算符的运算结果是 boolean 型。当运算符对应的关系成立时，运算结果为 true，否则为 false。所有比较运算符通常作为判断的依据用在条件语句中。比较运算符共有 6 个。

返回类型为布尔类型的表达式，如比较运算符，可以被组合在一起构成一个更复杂的表达式。这是通过逻辑运算符来实现的。逻辑运算符包括&（&&）（逻辑与）、||（逻辑或）、！（逻辑非）。逻辑运算符的操作元必须是 boolean 型数据。在逻辑运算符中，除了“！”是一元运算符，其他都是二元运算符。逻辑运算符的用法和含义如图所示：

运算符	用法	含义
&	两者都为1，结果才是1	与
|	只要有一者为1，结果就是1	或
~	1变0，0变1	取反
^	两者相同即为0，不同为1	异或

位运算符除“按位与”和“按位或”运算符外，其他只能用于处理整数的操作数。包括 byte、short、char、int 和 long 等数据类型。位运算是完全针对位方面的操作。整型数据在内存中以二进制的形式表示，如 int 型变量 7 的二进制表示是 00000000 00000000 0000000 000011。左边最高位是符号位，最高位是 0 表示正数，若为 1 则表示负数。负数采用补码表示，如-8 的二进制表示 111111111 111111111 111111 11111000。这样就可以对整型数据进行核位运算。

还可以对数据按二进制位进行移位操作。Java 中的移位运算符有以下 3 种：

运算符	用法
<<	左移
>>	右移
>>>	无符号右移

左移就是将运算符左边的操作数的二进制数据，按照运算符右边操作数指定的位数向左移动。自分补 0。右移则复杂一些。当使用“>>”符号时，如果最高位是 0.右移空的位就现入 0，如果最高位是 1，右移空的位就填入 1。

三元运算符的使用格式为：条件式?值 1:值 2

如果从低精度数据类型向高精度数据类型转换，则永远不会溢出，并且总是成功的;而把高数据类型向低精度数据类型转换时，则会有信息丢失，有可能失败。

隐式类型转换规则

操作数1的数据类型	操作数2的数据类型	转换后的数据类型
byte、short、char	int	int
byte、short、char、int、	long	long
byte、short、char、int、long	float	float
byte、short、char、int、long、float	double	double

当把高精度的变量的值赋给低精度的变量时，必须使用显式类型转换运算（又称强制类型转换）。执行显式类型转换时，可能会导致精度损失。除 boolean 类型外，其他基本类型都能以显式类型转换的方式实现转换。