第三章 分支与循环

1. 分支语句（掌握）

1.1 分支语句的概述

1. 分支语句：就是判断语句，通过判断来选择程序的执行路径
2. 分支语句的生活案例：
    a. 门禁：
        i. 输入密码
        ii. 进行判断
            1. true:开门
            2. false：不开门
    b. 高速限速：
        i. 获得行驶中车的速度（speed = 160）
        ii. 获得当前行驶的车道（假设：110-120）
        iii. 判断
            1. speed > 144扣分

1.2 分支语句的使用

1.2.1 接收用户数输入的信息

代码块

```
package com.powernode.if02;

import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {
        System.out.println("请输入一个整数:");
        //接受用户输入的整数，赋值给x(目前不需要掌握)
        int x = new Scanner(System.in).nextInt();
        System.out.println("x = " + x);
```

```
11
12          System.out.println("请输入一个小数：");
13          double d = new Scanner(System.in).nextDouble();
14          System.out.println("d = " + d);
15
16          System.out.println("请输入一个字符串：");
17          String str = new Scanner(System.in).next();
18          System.out.println("str = " + str);
19      }
20  }
```

1.2.2 if语句（门禁卡基础版）

代码块

```
1   package com.powernode.if02;
2
3   import java.util.Scanner;
4
5   public class Test02 {
6       public static void main(String[] args) {
7           /**
8            * 需求：模拟门禁卡，用户输入密码，正确执行开门操作
9            *     1.拿到用户输入的密码
10           *     2.编写分支语句进行判断：
11           *       如果(密码正确){
12           *           执行开门
13           *       }
14           *       if(<条件表达式>){
15           *           执行开门 //语句语句块
16           *       }
17           *     3.执行过程：
18           *       1.<条件表达式>
19           *           1.true:执行if语句块，开门
20           *           2.false：不执行if语句块，继续往下执行
21           */
22          //1.拿到用户输入的密码
23          int password = new Scanner(System.in).nextInt();
24          //2.编写分支语句进行判断：
25          if(password == 2009){
26              System.out.println("开门");
27          }
28          System.out.println("if语句执行完毕我开始执行");
29
30      }
31  }
```

1.2.3 if-else（门禁卡完善）

代码块

```
package com.powernode.if02;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {
        /**
         * 需求：模拟门禁卡，用户输入密码，正确执行开门操作
         *    1.拿到用户输入的密码
         *    2.编写分支语句进行判断：
         *      如果(密码正确){
         *          执行开门
         *      }否则{
         *          密码错误，请重新输入
         *      }
         *      if(<条件表达式>){
         *          执行开门
         *      }else{
         *          密码错误，请重新输入
         *      }
         *    3.执行过程：
         *      1.<条件表达式>
         *          1.true:执行if语句块，开门
         *          2.false：执行else语句块，密码错误，请重新输入
         */
        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断：
        if(password == 2009){
            System.out.println("开门");
        }else{
            System.out.println("密码错误，请重新输入");
        }


    }
}
```

1.2.4 if-else if-else （门禁卡添加管理员）

```java
package com.powernode.if02;

import java.util.Scanner;

public class Test04 {
    public static void main(String[] args) {
        /**
         * 需求：模拟门禁卡，用户输入密码，正确执行开门操作
         *    1.拿到用户输入的密码
         *    2.编写分支语句进行判断：
         *      如果(员工密码 == 2009){
         *          执行开门
         *      }如果(管理员密码 == 20092009){、
         *          添加员工
         *          删除员工
         *      }否则{
         *          密码错误，请重新输入
         *      }
         *      ------------------------------
         *      if(<条件表达式>){
         *          执行开门
         *      }else if(<条件表达式>){
         *          添加员工
         *          删除员工
         *      }else{
         *          密码错误，请重新输入
         *      }
         */
        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断：
        if(password == 2009){
            System.out.println("开门");
        }else if(password == 20092009){
            System.out.println("添加员工");
            System.out.println("删除员工");
        } else{
            System.out.println("密码错误，请重新输入");
        }


    }
}
```

1.2.5 if嵌套（门禁卡添加管理员完善）

代码块

```java
package com.powernode.if02;

import java.util.Scanner;

public class Test05 {
    public static void main(String[] args) {
        /**
         * 需求：模拟门禁卡，用户输入密码，正确执行开门操作
         *    1.拿到用户输入的密码
         *    2.编写分支语句进行判断：
         *      如果(员工密码 == 2009){
         *          执行开门
         *      }如果(管理员密码 == 20092009){、
         *          添加员工
         *          删除员工
         *      }否则{
         *          密码错误，请重新输入
         *      }
         *      ------------------------------
         *      if(<条件表达式>){
         *          执行开门
         *      }else if(<条件表达式>){
         *          添加员工
         *          删除员工
         *      }else{
         *          密码错误，请重新输入
         *      }
         */
        //1.拿到用户输入的密码
        String info = new Scanner(System.in).next();
        //2.编写分支语句进行判断：
        if(info.equals("2009")){
            System.out.println("开门");
        }else if(info.equals("#")){
            System.out.println("请输入管理员密码：");
            int password = new Scanner(System.in).nextInt();
            if (password == 20092009) {
                System.out.println("添加员工");
                System.out.println("删除员工");
            }else{
                System.out.println("密码错误，请重新输入");
            }
        } else{
            System.out.println("密码错误，请重新输入");
        }


```

```
48      }
49  }
```

1.2.6 if-elseif-...else（门禁卡添加超级管理员）

代码块

```
1   package com.powernode.if02;
2
3   import java.util.Scanner;
4
5   public class Test06 {
6       public static void main(String[] args) {
7           /**
8            * 需求：模拟门禁卡，用户输入密码，正确执行开门操作
9            *     1.拿到用户输入的密码
10           *     2.编写分支语句进行判断：
11           *       如果(员工密码 == 2009){
12           *           执行开门
13           *       }如果(普通管理员密码 == 20092009){、
14           *           添加员工
15           *           删除员工
16           *       }如果(超级管理员 == 200909){
17           *           分配权限
18           *           取消权限
19           *       }否则{
20           *           密码错误，请重新输入
21           *       }
22           *       ------------------------------
23           *       if(<条件表达式>){
24           *           执行开门
25           *       }else if(<条件表达式>){
26           *           添加员工
27           *           删除员工
28           *       }else if(<条件表达式>){
29           *           分配权限
30           *           取消权限
31           *       }else{
32           *           密码错误，请重新输入
33           *       }
34           */
35          //1.拿到用户输入的密码
36          int password = new Scanner(System.in).nextInt();
37          //2.编写分支语句进行判断：
38          if(password == 2009){
39              System.out.println("开门");
```

```
        }else if (password == 20092009) {
            System.out.println("添加员工");
            System.out.println("删除员工");
        }else if (password == 200909) {
            System.out.println("分配权限");
            System.out.println("取消权限");
        }else{
            System.out.println("密码错误，请重新输入");
        }
    }
}
```

1.3 分支语句的简写

1.3.1 if

代码块

```
package com.powernode.if03;

import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {

        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断:
        /*if(password == 2009){
            System.out.println("开门");
        }*/
        /**
         * 1.如果if语句块中只有一条语句，{}可以省略
         * 2.省略后，相当于if语句后的第一条语句写在了{}中
         * 3.if语句块中有多条语句，不可以省略
         */
        if (password == 2009)
            System.out.println("开门");
            System.out.println("--------");

        System.out.println("if语句执行完毕我开始执行");

    }
}
```

1.3.2 if-else

代码块

```
package com.powernode.if03;

import java.util.Scanner;

public class Test02 {
    public static void main(String[] args) {

        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断:
        if(password == 2009) System.out.println("开门");
        else System.out.println("密码错误，请重新输入");
    }
}
```

1.3.3 if-elseif-else

代码块

```
package com.powernode.if03;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {

        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断:
        if(password == 2009)
            System.out.println("开门");
        else if (password == 20092009) {
            System.out.println("添加员工");
            System.out.println("删除员工");
        }else if (password == 200909) {
            System.out.println("分配权限");
            System.out.println("取消权限");
        }else
            System.out.println("密码错误，请重新输入");

    }
}
```

1.4 分支语句的总结

1. if语句可以单独出现
2. else和elseif不可以单独出现
3. if-else||if -else if ||if -else if -else 可以组队出现
4. else if 可以包含0到多个
5. if语句的简写
   a. 语句块中只有一条语句，可以省略{}，省略后相当于，语句块中的第一条语句写在了{}中
   b. 语句块中有多条语句，不可以省略

1.5 电信计费系统核心代码实现

1.5.1 电信电话计费项目（一）

代码块

```
package com.powernode.if04;

import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {
        /**
         * 需求：某电信公司电话计费规则如下：
         *   1.前3分钟，2角
         *   2.之后一分钟，1.5角
         *   3.不够一分钟，按照一分钟算
         *   4.求 x 秒，需要多少钱
         */
        //1.定义一个变量，获得打电话的秒数
        int seconds = new Scanner(System.in).nextInt();
        double telephoneCharges = 0;
        //超出的分钟数
        int count = 0;
        if (seconds <= 180) {//前3分钟
            telephoneCharges = 2;
        }else{//大于3分钟的(超出分钟数)
            /**
             * 1.整分整秒
             * 2.不是整分整秒
             */
            if (seconds % 60 == 0) {//1.整分整秒
```

```
                count = (seconds - 180) / 60;
            }else{//2.不是整分整秒
                count = (seconds - 180) / 60 + 1;//我们计算的是整数，舍弃了小数，所以+1
            }
        }
        if (seconds == 0) {
            System.out.println(0);
        }else{
            //总费用：前三分钟的 + 超出分钟数 * 1.5
            System.out.println(telephoneCharges = 2 + count * 1.5);
        }
        //System.out.println(telephoneCharges);
    }
}
```

1.5.2 电信电话计费项目（二） 代码优化

代码块

```
  package com.powernode.if04;

import java.util.Scanner;

public class Test02 {
    public static void main(String[] args) {
        /**
         * 需求：某电信公司电话计费规则如下：
         *   1.前3分钟，2角
         *   2.之后一分钟，1.5角
         *   3.不够一分钟，按照一分钟算
         *   4.求 x 秒，需要多少钱
         */
        //1.定义一个变量，获得打电话的秒数
        int seconds = new Scanner(System.in).nextInt();
        double telephoneCharges = 0;
        //超出的分钟数
        int count = 0;
        if (seconds <= 180) {//前3分钟
            telephoneCharges = 2;
        }else{//大于3分钟的(超出分钟数)
            /**
             * 1.整分整秒
             * 2.不是整分整秒
             */
            if (seconds % 60 == 0) {//1.整分整秒
```

```
                count = (seconds - 180) / 60;
            }else{//2.不是整分整秒
                count = (seconds - 180) / 60 + 1;//我们计算的是整数，舍弃了小数，所以+1
            }
        }
        /* if (seconds == 0) {
            System.out.println(0);
        }else{
            //总费用：前三分钟的 + 超出分钟数 * 1.5
            System.out.println(telephoneCharges = 2 + count * 1.5);
        }*/
        System.out.println(seconds == 0?0:(telephoneCharges = 2 + count * 1.5));
    }
}
```

1.5.3 电信电话计费项目（三） 代码优化

代码块

```
package com.powernode.if04;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {
        /**
         * 需求：某电信公司电话计费规则如下：
         *   1.前3分钟，2角
         *   2.之后一分钟，1.5角
         *   3.不够一分钟，按照一分钟算
         *   4.求 x 秒，需要多少钱
         */
        //1.定义一个变量，获得打电话的秒数
        int seconds = new Scanner(System.in).nextInt();
        double telephoneCharges = 2;
        //超出的分钟数
        int count = 0;
        if (seconds > 180){//大于3分钟的(超出分钟数)
            /**
             * 1.整分整秒
             * 2.不是整分整秒
             */
            if (seconds % 60 == 0) {//1.整分整秒
                count = (seconds - 180) / 60;
```

```
            }else{//2.不是整分整秒
                count = (seconds - 180) / 60 + 1;//我们计算的是整数，舍弃了小数，所以+1
            }
        }

        System.out.println(seconds == 0?0:(telephoneCharges = 2 + count * 1.5));
    }
}
```

1.6 使用三目运算符优化分支语句

- 语法：<条件表达式>?<表达式1>:<表达式2>
- 执行过程：
  - <条件表达式>
    - true:<表达式1>
    - false:<表达式2>

1.6.1 三目优化if-else

代码块

```
package com.powernode.if05;

import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {

        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        /*//2.编写分支语句进行判断:
        if(password == 2009){
            System.out.println("开门");
        }else{
            System.out.println("密码错误，请重新输入");
        }*/
        String info = password == 2009 ? "开门" : "密码错误，请重新输入";
        System.out.println(info);

    }
}
```

1.6.2 三目优化if-elseif-else

代码块

```
package com.powernode.if05;

import java.util.Scanner;

public class Test02 {
    public static void main(String[] args) {

        //1.拿到用户输入的密码
        int password = new Scanner(System.in).nextInt();
        //2.编写分支语句进行判断:
        /*if(password == 2009){
            System.out.println("开门");
        }else if (password == 20092009) {
            System.out.println("添加员工");
            System.out.println("删除员工");
        }else if (password == 200909) {
            System.out.println("分配权限");
            System.out.println("取消权限");
        }else{
            System.out.println("密码错误，请重新输入");
        }*/
        String info = password == 2009 ? "开门" :
                password == 20092009 ? "添加员工\n删除员工" :
                password == 200909 ? "分配权限\n取消权限" : "密码错误，请重新输入";
        System.out.println(info);

    }
}
```

1.6.3 电信电话计费项目（四） 三目优化

代码块

```
package com.powernode.if05;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {
        /**
         * 需求：某电信公司电话计费规则如下:
         *   1.前3分钟，2角
```

```
10            *   2.之后一分钟，1.5角
11            *   3.不够一分钟，按照一分钟算
12            *   4.求 x 秒，需要多少钱
13            */
14           //1.定义一个变量，获得打电话的秒数
15           int seconds = new Scanner(System.in).nextInt();
16           double telephoneCharges = 2;
17           //超出的分钟数
18           /*int count = 0;
19           if (seconds > 180){//大于3分钟的(超出分钟数)
20               *//**
21                * 1.整分整秒
22                * 2.不是整分整秒
23                *//*
24               if (seconds % 60 == 0) {//1.整分整秒
25                   count = (seconds - 180) / 60;
26               }else{//2.不是整分整秒
27                   count = (seconds - 180) / 60 + 1;//我们计算的是整数，舍弃了小数，所以+1
28               }
29           }*/
30           int count = seconds > 180? seconds % 60 == 0? (seconds - 180) / 60: (seconds - 180) / 60 + 1: 0;
31           System.out.println(seconds == 0?0:(telephoneCharges = telephoneCharges + count * 1.5));
32       }
33   }
```

1.6.4 电信电话计费项目（五） 三目极致优化

- 三目工作中一般就是使用一个，或者两个

代码块

```
1   package com.powernode.if05;
2
3   import java.util.Scanner;
4
5   public class Test04 {
6       public static void main(String[] args) {
7           /**
8            * 需求：某电信公司电话计费规则如下：
9            *   1.前3分钟，2角
10           *   2.之后一分钟，1.5角
11           *   3.不够一分钟，按照一分钟算
12           *   4.求 x 秒，需要多少钱
13           */
```

```
        //1.定义一个变量，获得打电话的秒数
        int seconds = new Scanner(System.in).nextInt();
        double telephoneCharges = 2;
        System.out.println(seconds == 0?0:(telephoneCharges = telephoneCharges
    + (seconds > 180? seconds % 60 == 0? (seconds - 180) / 60: (seconds - 180) / 60
     + 1: 0) * 1.5));
    }
}
```

2. switch语句（掌握）

2.1 switch的概述

1. switch也是分支语句的一种
2. switch的语法：

代码块

```
switch (<表达式>) {
    case 常量1:
        语句块1;
        break;
    case 常量2:
        语句块2;
        break;
    case 常量3:
        语句块3;
        break;
    ...
    default :
       语句块4;
       break;
}
1.<表达式>的值：可以是 byte,short ,int ,long,char,String,enum(枚举)
2.常量：
   1.字面量，比如：1,2,3, 'a',"abc"
   2.不可以变量的变量（final修饰的变量）
```

2.2 switch和if的区别

代码块

```
package com.powernode.switch06;

```

```java
import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {
        int password = new Scanner(System.in).nextInt();
        if (password > 2009) {
            System.out.println("开门");
        }
        /**
         * 1.if 使用的 <布尔表达式>，switch使用的是<表达式>
         * 2.if可以做范围判断，switch只能做等值判断
         * 3.if比switch更加灵活
         * 4.switch比if效率更高(? )
         */
        //double d = 1.2;
        switch(password){
            case 2009:
                System.out.println("开门");
        }
    }
}
```

2.3 break防止case穿透

代码块

```java
package com.powernode.switch06;

import java.util.Scanner;

public class Test02 {
    public static void main(String[] args) {
        int password = new Scanner(System.in).nextInt();
        if (password == 2009) {
            System.out.println("开门");
        } else if (password == 20092009) {
            System.out.println("添加员工\n删除员工");
        }

        switch(password){
            case 2009:
                System.out.println("开门");
                break;//结束分支语句，防止case穿透
            case 20092009:
                System.out.println("添加员工\n删除员工");
                break;
```

```
        }
    }
}
```

2.4 default

代码块

```
package com.powernode.switch06;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {
        int password = new Scanner(System.in).nextInt();
        if (password == 2009) {
            System.out.println("开门");
        } else if (password == 20092009) {//普通管理员
            System.out.println("添加员工\n删除员工");
        }else if (password == 200909) {
            System.out.println("分配权限\n取消权限");
        }else{
            System.out.println("密码错误");
        }

        switch(password){
            case 2009:
                System.out.println("开门");
                break;//结束分支语句，防止case穿透
            case 20092009:
                System.out.println("添加员工\n删除员工");
                break;
            case 200909:
                System.out.println("分配权限\n取消权限");
            default:
                System.out.println("密码错误");
                break;//在实际的工作中，default一般写在最后，如果写在了最后break可以省略
        }
    }
}
```

2.5 switch和if的底层实现原理

2.5.1 switch底层tableswitch效率 是 o(1)

- 类似于【数组】

代码块

```
D:\powernode\02-JavaSE\03-code\JavaProject\day04\src\com\powernode\switch06>javac Test04.java

D:\powernode\02-JavaSE\03-code\JavaProject\day04\src\com\powernode\switch06>javap -c Test04
警告: 文件 .\Test04.class 不包含类 Test04
Compiled from "Test04.java"
public class com.powernode.switch06.Test04 {
  public com.powernode.switch06.Test04();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."<init>":()V
       4: return

  public static void main(java.lang.String[]);
    Code:
       0: new           #7                  // class java/util/Scanner
       3: dup
       4: getstatic     #9                  // Field java/lang/System.in:Ljava/io/InputStream;
       7: invokespecial #15                 // Method java/util/Scanner."<init>":(Ljava/io/InputStream;)V
      10: invokevirtual #18                 // Method java/util/Scanner.nextInt:()I
      13: istore_1
      14: iload_1  // 加载变量到栈顶
      15: tableswitch   { // 1 to 3 使用调转表指令
                    1: 40  // case 1 跳转到地址 40
                    2: 51
                    3: 62
              default: 70
          }
      40: getstatic     #22                 // Field java/lang/System.out:Ljava/io/PrintStream;获取System.out对象
      43: ldc           #26                 // String 开门  加载字符串开门
      45: invokevirtual #28                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V  调用println方法打印
      48: goto          78
      51: getstatic     #22                 // Field java/lang/System.out:Ljava/io/PrintStream;
```

```
      54: ldc           #34                 // String 添加员工\n删除员工
      56: invokevirtual #28                 // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
      59: goto          78
      62: getstatic     #22                 // Field
java/lang/System.out:Ljava/io/PrintStream;
      65: ldc           #36                 // String 分配权限\n取消权限
      67: invokevirtual #28                 // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
      70: getstatic     #22                 // Field
java/lang/System.out:Ljava/io/PrintStream;
      73: ldc           #38                 // String 密码错误
      75: invokevirtual #28                 // Method
java/io/PrintStream.println:(Ljava/lang/String;)V
      78: return
}

```

2.5.2 switch底层lookupswitch效率 是 o(n)

- 类似于【排序数组 + 二分查找】

代码块

```
D:\powernode\02-JavaSE\03-
code\JavaProject\day04\src\com\powernode\switch06>javac Test05.java

D:\powernode\02-JavaSE\03-
code\JavaProject\day04\src\com\powernode\switch06>javap -c Test05
警告: 文件 .\Test05.class 不包含类 Test05
Compiled from "Test05.java"
public class com.powernode.switch06.Test05 {
  public com.powernode.switch06.Test05();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."
<init>":()V
       4: return

  public static void main(java.lang.String[]);
    Code:
       0: new           #7                  // class java/util/Scanner
       3: dup
       4: getstatic     #9                  // Field
java/lang/System.in:Ljava/io/InputStream;
```

```
18          7: invokespecial #15                 // Method java/util/Scanner."
    <init>":(Ljava/io/InputStream;)V
19         10: invokevirtual #18                 // Method
    java/util/Scanner.nextInt:()I
20         13: istore_1
21         14: iload_1
22         15: lookupswitch  { // 3
23                    2009: 48
24                  200909: 70
25                20092009: 59
26                 default: 78
27               }
28         48: getstatic     #22                 // Field
    java/lang/System.out:Ljava/io/PrintStream;
29         51: ldc           #26                 // String 开门
30         53: invokevirtual #28                 // Method
    java/io/PrintStream.println:(Ljava/lang/String;)V
31         56: goto          86
32         59: getstatic     #22                 // Field
    java/lang/System.out:Ljava/io/PrintStream;
33         62: ldc           #34                 // String 添加员工\n删除员工
34         64: invokevirtual #28                 // Method
    java/io/PrintStream.println:(Ljava/lang/String;)V
35         67: goto          86
36         70: getstatic     #22                 // Field
    java/lang/System.out:Ljava/io/PrintStream;
37         73: ldc           #36                 // String 分配权限\n取消权限
38         75: invokevirtual #28                 // Method
    java/io/PrintStream.println:(Ljava/lang/String;)V
39         78: getstatic     #22                 // Field
    java/lang/System.out:Ljava/io/PrintStream;
40         81: ldc           #38                 // String 密码错误
41         83: invokevirtual #28                 // Method
    java/io/PrintStream.println:(Ljava/lang/String;)V
42         86: return
43       }
```

2.5.3 if语句底层if_icmpne效率 是 o(n)

代码块

```
1  D:\powernode\02-JavaSE\03-
   code\JavaProject\day04\src\com\powernode\switch06>javac Test06.java
2
3  D:\powernode\02-JavaSE\03-
   code\JavaProject\day04\src\com\powernode\switch06>javap -c Test06
```

```
警告: 文件 .\Test06.class 不包含类 Test06
Compiled from "Test06.java"
public class com.powernode.switch06.Test06 {
  public com.powernode.switch06.Test06();
    Code:
       0: aload_0
       1: invokespecial #1                  // Method java/lang/Object."<init>":()V
       4: return

  public static void main(java.lang.String[]);
    Code:
       0: new           #7                  // class java/util/Scanner
       3: dup
       4: getstatic     #9                  // Field java/lang/System.in:Ljava/io/InputStream;
       7: invokespecial #15                 // Method java/util/Scanner."<init>":(Ljava/io/InputStream;)V
      10: invokevirtual #18                 // Method java/util/Scanner.nextInt:()I
      13: istore_1
      14: iload_1         //加载变量password
      15: sipush        2009 //压入2009
      18: if_icmpne     32  // 如果不相等跳转到32
      21: getstatic     #22                 // Field java/lang/System.out:Ljava/io/PrintStream;
      24: ldc           #26                 // String 开门
      26: invokevirtual #28                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V
      29: goto          74
      32: iload_1       // 加载变量password
      33: ldc           #34                 // int 20092009
      35: if_icmpne     49 // 如果不相等跳转到49
      38: getstatic     #22                 // Field java/lang/System.out:Ljava/io/PrintStream;
      41: ldc           #35                 // String 添加员工\n删除员工
      43: invokevirtual #28                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V
      46: goto          74
      49: iload_1
      50: ldc           #37                 // int 200909
      52: if_icmpne     66
      55: getstatic     #22                 // Field java/lang/System.out:Ljava/io/PrintStream;
      58: ldc           #38                 // String 分配权限\n取消权限
      60: invokevirtual #28                 // Method java/io/PrintStream.println:(Ljava/lang/String;)V
```

```
41         63: goto          74
42         66: getstatic     #22                 // Field
    java/lang/System.out:Ljava/io/PrintStream;
43         69: ldc           #40                 // String 密码错误
44         71: invokevirtual #28                 // Method
    java/io/PrintStream.println:(Ljava/lang/String;)V
45         74: return
46   }
```

2.6 case条件合并

代码块

```
1   package com.powernode.switch07;
2
3   import java.util.Scanner;
4
5   public class Test01 {
6       public static void main(String[] args) {
7           /**
8            * 一年四个季度:
9            *  第一季度: 1 、2 、3 : 冬季
10           *  第二季度: 4 、5 、6 : 春季
11           *  第三季度: 7 、8 、9 : 夏季
12           *  第四季度: 10 、11 、12 : 秋季
13           * 接收一个月份，判断是第几季度，并输出季度和季节
14           */
15          int month = new Scanner(System.in).nextInt();
16          if (month == 1 || month == 2 || month == 3) {
17              System.out.println("第一季度");
18              System.out.println("冬季");
19          } else if (month == 4 || month == 5 || month == 6) {
20              System.out.println("第二季度");
21              System.out.println("春季");
22          }else if (month == 7 || month == 8 || month == 9) {
23              System.out.println("第三季度");
24              System.out.println("夏季");
25          }else if (month == 10 || month == 11 || month == 12) {
26              System.out.println("第四季度");
27              System.out.println("秋季");
28          }else{
29              System.out.println("输入月份不合法");
30          }
31          System.out.println("----------------------");
32          switch (month){
33              case 1,2,3:
```

```
                System.out.println("第一季度");
                System.out.println("冬季");
                break;
            case 4,5,6:
                System.out.println("第二季度");
                System.out.println("春季");
                break;
            case 7,8,9:
                System.out.println("第三季度");
                System.out.println("夏季");
                break;
            case 10,11,12:
                System.out.println("第四季度");
                System.out.println("秋季");
                break;
            default:
                System.out.println("输入月份不合法");

        }
    }
}
```

2.7 JDK新特性

2.7.1 新特性switch基础语法

代码块

```
package com.powernode.switch08;

import java.util.Scanner;

public class Test01 {
    public static void main(String[] args) {
        int password = new Scanner(System.in).nextInt();
        switch (password){
            case 2009:
                System.out.println("普通员工");
                break;
            case 20092009:
                System.out.println("普通管理员");
                break;
        }
        /**
         * jdk新特性语法:
         *   1.冒号: 替换为 ->
```

```
19            *   2.删除break;
20            */
21           switch (password){
22              /* case 2009->
23                   System.out.println("普通员工");
24               case 20092009->
25                   System.out.println("普通管理员");*/
26               case 2009-> System.out.println("普通员工");
27               case 20092009-> System.out.println("普通管理员");
28           }
29
30       }
31   }
```

2.7.2 case条件合并新语法

代码块

```
1   package com.powernode.switch08;
2
3   import java.util.Scanner;
4
5   public class Test02 {
6       public static void main(String[] args) {
7
8           int month = new Scanner(System.in).nextInt();
9           /*switch (month){
10              case 1,2,3:
11                  System.out.println("第一季度");
12                  System.out.println("冬季");
13                  break;
14              case 4,5,6:
15                  System.out.println("第二季度");
16                  System.out.println("春季");
17                  break;
18              case 7,8,9:
19                  System.out.println("第三季度");
20                  System.out.println("夏季");
21                  break;
22              case 10,11,12:
23                  System.out.println("第四季度");
24                  System.out.println("秋季");
25                  break;
26              default:
27                  System.out.println("输入月份不合法");
28
```

```
        }*/
        //有多条语句添加{}
        switch (month){
            case 1,2,3->{
                System.out.println("第一季度");
                System.out.println("冬季");
            }
            case 4,5,6->{
                System.out.println("第二季度");
                System.out.println("春季");
            }
            case 7,8,9->{
                System.out.println("第三季度");
                System.out.println("夏季");
            }
            case 10,11,12->{
                System.out.println("第四季度");
                System.out.println("秋季");
            }
            default-> System.out.println("输入月份不合法");

        }
    }
}
```

2.7.3 switch作为表达式返回一个结果

代码块

```
package com.powernode.switch08;

import java.util.Scanner;

public class Test03 {
    public static void main(String[] args) {
        /**
         * 公司抽奖
         *   一等奖：100w
         *   二等奖：50w
         *   三等奖：10w
         *   其他：0w
         *  输入几等奖，返回奖金金额
         */
        int bonus = new Scanner(System.in).nextInt();
        int money = switch (bonus) {
            case 1 -> 100;
```

```
            case 2 -> 50;
            case 3 -> 10;
            default -> 0;//switch作为表达式返回一个结果是必须，写上default
        };
    }
}
```

2.8 switch使用char类型

代码块

```
package com.powernode.switch08;

import java.util.Scanner;

public class Test04 {
    public static void main(String[] args) {
        char c = 'b';
        switch (c){
            case 'a' -> System.out.println('a');
            case 'b' -> System.out.println('b');
            case 'c' -> System.out.println('c');
            default -> System.out.println("非法字符");
        }
    }
}
```

2.9 switch表达式的兼容性（了解）

代码块

```
package com.powernode.switch08;

public class Test05 {
    public static void main(String[] args) {
        char c = 'b';
        /**
         * 1.switch表达式中的数据类型可以和case条件不一样
         * 2.能兼容即可（不用强制转换直接赋值）
         * 3.注意：一般不这么写
         */
        switch (c){
            case 'a' -> System.out.println('a');
            case 98 -> System.out.println('b');
            case 'c' -> System.out.println('c');
```

```
            default -> System.out.println("非法字符");
        }
        //char c1 = 98;
    }
}
```

2.10 switch中使用String（JDK1.7开始支持）

代码块

```
package com.powernode.switch08;

import java.util.Scanner;

public class Test06 {
    public static void main(String[] args) {
        String week = new Scanner(System.in).next();
        switch (week){
            case "星期一","星期二","星期四","星期五","星期六" ->{
                System.out.println("上课");
                System.out.println("上自习");
            }
            case "星期三" -> System.out.println("上自习");
            case "星期日" -> System.out.println("休息");
            default -> System.out.println("输入数据不合法");
        }
    }
}
```

作业

1. 使用分支语句模拟温度计
   a. 如果大于36.5并且小于40 摄氏度，输出发烧
   b. 如果大于等于 40 摄氏度，请立即就医，避免发生意外
   c. 否则输出，温度正常，注意多喝开水

2. 使用简单写法（省略{}），改写如上练习

3. 使用三目运算符优化（选做）

4. 滴滴打车（有难度）
   a. 前三公里 8元
   b. 之后一公里2.3 元（不够1公里按照1公里算）
   c. 求x公里，多少钱

代码块

```
package com.powernode.exercise01;

import java.util.Scanner;

public class Test04 {
    public static void main(String[] args) {
        double kilometer = new Scanner(System.in).nextDouble();
        //定义一个变量存储费用
        double money = 8;
        //超出的公里数
        int count = 0;
        if (kilometer > 3) {
            if (kilometer % 1 == 0) {//整公里数
                count = (int)kilometer - 3;
            }else{//不是整公里数
                //count = (int)kilometer - 3 + 1;
                count = (int)kilometer - 2;
            }
        }
        //总费用 = 前三公里的 + 超出的公里数 * 2.3
        money = money + count * 2.3;
        System.out.println(kilometer ==0 ? 0 : money);
    }
}
```

5. 使用switch（老语法）从控制台接收整数参数.如果该数为1-7，打印对应的星期值，否则打印“非法参数”

6. 使用switch新语法进行改写

7. 把上课讲的案例，都敲一遍

3. 循环语句（掌握）

3.1 循环的概述

1. 循环有什么用
    - 让一段代码，重复执行

代码块

```
package com.powernode.while02;

public class Test01 {
    public static void main(String[] args) {
        System.out.println("*****");
        System.out.println("*****");
        System.out.println("*****");
        System.out.println("-------------");
        int i= 0;
        while(i < 3){
            System.out.println("*****");
            i++;
        }
    }
}
```

2. 怎么使用
    a. while
    b. do-while
    c. for

3.2 while

3.2.1 while循环语法

代码块

```
package com.powernode.while02;

public class Test02 {
    public static void main(String[] args) {
        /**
```

```
6            * 1.语法:
7            *    1、定义循环变量
8            *    2. while (<布尔表达式>){
9            *         循环体;
10           *    }
11           * 2.执行过程:
12           *    <布尔表达式>:
13           *       1.true:执行循环体
14           *       2.false:不执行循环体
15           */
16          //1.定义循环变量
17          int i = 0;
18          //2.编写循环语句
19          while(i < 3){
20              System.out.println("*****");
21              i++;
22          }
23      }
24  }
```

3.2.2 while循环执行过程

代码块

```
1   package com.powernode.while02;
2
3   public class Test03 {
4       public static void main(String[] args) {
5           /**
6            * 1.语法:
7            *    1、定义循环变量
8            *    2. while (<布尔表达式>){
9            *         循环体;
10           *    }
11           * 2.执行过程:
12           *    <布尔表达式>:
13           *       1.true:执行循环体
14           *       2.false:不执行循环体
15           */
16          //1.定义循环变量
17          int i = 0;
18          //2.编写循环语句
19          while(i < 3){
20              System.out.println("i = " + i);
21              i++;
22          }
```

```
        /**
         * 第一次循环:
         * 1.i = 0
         * 2.i < 3 : 0 < 3 : true
         * 3.System.out.println("i = " + i); i = 0
         * 4.i++; i = 1
         * 第二次循环:
         * 5.i < 3 : 1 < 3 : true
         * 6.System.out.println("i = " + i);i = 1
         * 7.i++; i = 2
         * 第三次循环:
         * 8.i < 3 :  2 < 3 : true
         * 9.System.out.println("i = " + i);i = 2
         * 10.i++; i = 3
         * 第四次循环:
         * 11.i < 3 :  3 < 3 :false ,结束了循环
         */
    }
}
```

3.2.3 使用循环打印2行3列的*

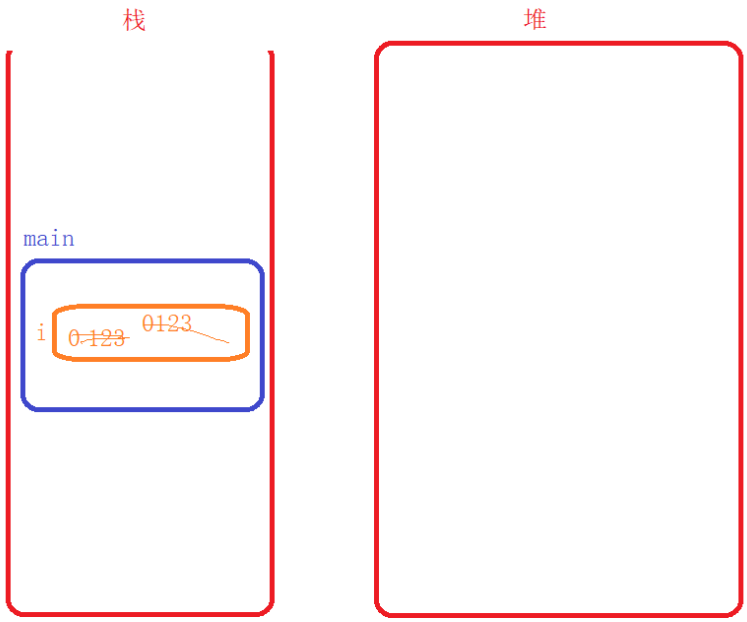
代码块

```
package com.powernode.while02;

public class Test04 {
    public static void main(String[] args) {
        /**
         * 1.语法:
         *    1、定义循环变量
         *    2. while (<布尔表达式>){
         *         循环体;
         *    }
         * 2.执行过程:
         *    <布尔表达式>:
         *       1.true:执行循环体
         *       2.false:不执行循环体
         */
        //使用循环打印2行3列的*
        /*
           ***
           ***
        */
        int i = 0;
        while(i < 3){
```

```
23                 System.out.print("*");
24                 i++;
25             }
26             System.out.println();
27 
28             int j = 0;
29             while(j < 3){
30                 System.out.print("*");
31                 j++;
32             }
33             System.out.println();
34         }
35     }
```

3.2.4 代码优化（一）及内存分析

```
public class Test05 {
    public static void main(String[] args) {
        /*
           ***
           ***
        */
        int i = 0;
        while(i < 3){
            System.out.print("*");
            i++;
        }
        System.out.println();

        i = 0;
        while(i < 3){
            System.out.print("*");
            i++;
        }
        System.out.println();
    }
}
```



代码块

```
1  package com.powernode.while02;
2  
3  public class Test05 {
4      public static void main(String[] args) {
5  
6          /*
7             ***
8             ***
9          */
10         int i = 0;
11         while(i < 3){
```

```
            System.out.print("*");
            i++;
        }
        System.out.println();

        i = 0;
        while(i < 3){
            System.out.print("*");
            i++;
        }
        System.out.println();
    }
}
```

3.2.5 代码优化（二）循环嵌套

代码块

```
package com.powernode.while02;

public class Test06 {
    public static void main(String[] args) {

        /*
           ***
           ***
        */
        int j = 0;
        while(j < 2){
            int i = 0;
            while(i < 3){
                System.out.print("*");
                i++;
            }
            System.out.println();
            j++;
        }
        /*
        int i = 0;
        while(i < 3){
            System.out.print("*");
            i++;
        }
```

```
26            System.out.println();
27
28            i = 0;
29            while(i < 3){
30                System.out.print("*");
31                i++;
32            }
33            System.out.println();*/
34        }
35    }
```

3.2.6 代码优化（三） 及内存分析

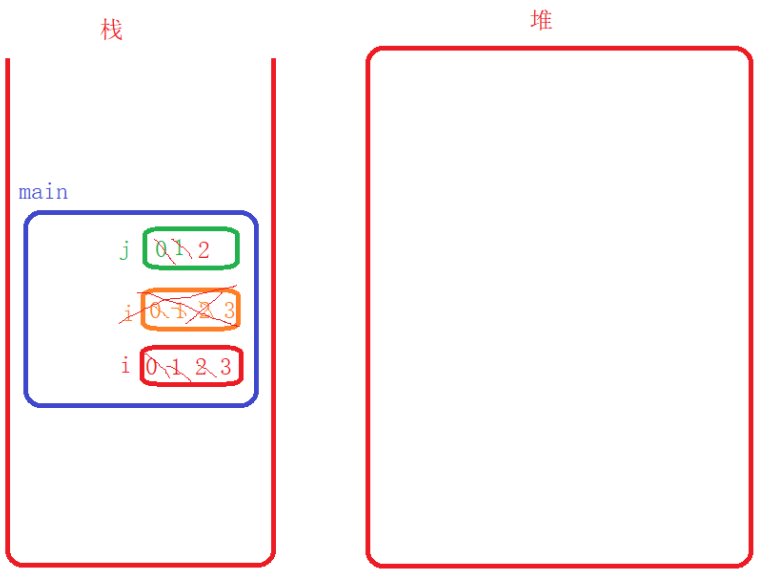
```
int j = 0;
while(j++ < 2){
    int i = 0;
    while(i++ < 3){
        System.out.print("*");
    }
    System.out.println();
}
```

外层循环

内层循环

2行3列的*
外层循环执行1次
内层循环执行3次
***
***
外层循环控制行
内层循环控制列

```
public class Test07 {
    public static void main(String[] args) {
        int j = 0;
        while(j++ < 2){
            int i = 0;
            while(i++ < 3){
                System.out.print("*");
            }
            System.out.println();
        }

    }
}
```



代码块

```
1   package com.powernode.while02;
2
3   public class Test07 {
```

```
    public static void main(String[] args) {
        int j = 0;
        while(j++ < 2){
            int i = 0;
            while(i++ < 3){
                System.out.print("*");
            }
            System.out.println();
        }

    }
}
```

3.2.7 死循环

代码块

```
package com.powernode.while02;

import java.util.Scanner;

public class Test08 {
    public static void main(String[] args) {
        /**
         * 死循环
         *   1.没有结束循环的条件
         *   2.循环体一直执行
         */
        while(true){
            System.out.println("输入第一个整数: ");
            int x = new Scanner(System.in).nextInt();
            System.out.println("输入第二个整数: ");
            int y = new Scanner(System.in).nextInt();
            System.out.println("两数之和: " + (x + y));
        }

    }
}
```

3.3 do-while

```
代码块
1  package com.powernode.dowhile03;
2
3  public class Test01 {
4      public static void main(String[] args) {
5          /**
6           * 1.语法:
7           *    do{
8           *        循环体;
9           *    }while(<布尔表达式>);
10          * 2.执行过程:
11          *   1.循环体;
12          *   2.<布尔表达式>
13          *        1.true:循环体;
14          *        2.false:结束循环
15          */
16         int i = 0;
17         do{
18             System.out.println("i = " + i);
19         }while(i++ <2);
20         System.out.println("i = " + i);
21         /**
22          * 第一次循环:
23          * 1.i = 0
24          * 2.System.out.println("i = " + i); i = 0
25          * 3.i++ <2
26          *   1.i < 2 : 0 < 2 : true
27          *   2.i++ : i = 1
28          * 第二次循环
29          * 4.System.out.println("i = " + i); i= 1
30          * 5.i++ <2
31          *   1.i < 2 : 1 < 2 : true
32          *   2.i++ : i = 2
33          * 第三次循环
34          * 6.System.out.println("i = " + i); i = 2
35          * 7.i++ <2
36          *   1.i < 2 : 2 < 2 :false
37          *   2.i++ : i = 3 //会执行
38          */
39
40     }
41 }
```

3.3.1 do-while应用场景（简易版的增删改查的逻辑）

```
代码块
```

```java
package com.powernode.dowhile03;

import java.util.Scanner;

public class Test02 {
    public static void main(String[] args) {
        /**
         * 1.do-while应用场景（简易版的增删改查的逻辑）
         * 2.界面如下:
         *    请选择如下操作:
         *       1.增加    2.删除    3.修改    4.查询    5.退出
         * 3.选择对应的编号执行相应的操作
         */
        int index = 0;
        do {
            //1.输出界面
            System.out.println("请选择如下操作: ");
            System.out.print("1.增加\t");
            System.out.print("2.删除\t");
            System.out.print("3.修改\t");
            System.out.print("4.查询\t");
            System.out.println("5.退出\t");
            System.out.print("请选择: ");
            index = new Scanner(System.in).nextInt();
            switch (index) {
                case 1 -> System.out.println("-----执行增加操作-----");
                case 2 -> System.out.println("-----执行删除操作-----");
                case 3 -> System.out.println("-----执行修改操作-----");
                case 4 -> System.out.println("-----执行查询操作-----");
                case 5 -> System.out.println("-----执行退出操作-----");
                default -> System.out.println("无效操作，请选择(1-5)");
            }
        }while(index != 5);//如果用户输入的不是5，循环一直执行，输入5结束
    }
}
```

3.3.2 do-while和if嵌套

- 输出【1-6】之间2的倍数

代码块

```java
package com.powernode.dowhile03;

import java.util.Scanner;

public class Test03 {
```

```
6       public static void main(String[] args) {
7           //输出【1-6】之间2的倍数
8           //输出1-6
9           int i = 1;
10          do{
11              //加入判断，如果是2的倍数就输出
12              if (i % 2 == 0) {
13                  System.out.println("i = " + i);
14              }
15          }while(i ++ < 6);
16
17      }
18  }
```

3.3.3 while和if嵌套

- 输出【1-6】之间2的倍数

代码块

```
1   package com.powernode.dowhile03;
2
3   public class Test04 {
4       public static void main(String[] args) {
5           //输出【1-6】之间2的倍数
6           //输出1-6
7           int i = 0;
8           while(i++ < 6){
9               if (i % 2 == 0) {
10                  System.out.println("i = " + i);
11              }
12          }
13
14      }
15  }
```

3.4 for循环

代码块

```
1       package com.powernode.for04;
2
3   public class Test01 {
4       public static void main(String[] args) {
```

```
        /**
         * 1.语法:
         *    for(<循环变量初始化>;<判断循环变量>;<修改循环变量>){
         *        循环体;
         *    }
         */
        /*int i = 0;
        while(i < 3){
            System.out.println("i = " + i);
            i++;
        }*/
        for(int i = 0;i < 3;i++){
            System.out.println("i = " + i);
        }
        /**
         * 第一次循环
         * 1.int i = 0
         * 2.i < 3 :  0 < 3 : true
         * 3.System.out.println("i = " + i); i = 0
         * 第二次循环
         * 4.i++ : i = 1
         * 5.i < 3 : 1 < 3 : true
         * 6.System.out.println("i = " + i); i = 1
         * 第三次循环
         * 7.i++ : i = 2
         * 8.i < 3 : 2 < 3 :true
         * 9.System.out.println("i = " + i); i = 2
         * 第四次循环
         * 10.i++ : i = 3
         * 11.i < 3 :  3 < 3 :false : 结束循环
         */
    }
}
```

3.5 *Unreachable statement*

代码块

```
package com.powernode.for04;

public class Test02 {
    public static void main(String[] args) {
        for(;;){
            System.out.println("*");
        }
        /**
```

```
 9            * Unreachable statement :遥不可及的声明
10            * 说明如下代码没有机会执行
11            */
12           System.out.println("-------");
13       }
14   }
```

3.6 输出【1-6】之间的质数

3.6.1 模拟计算

代码块

```
1   package com.powernode.for05;
2
3   public class Test01 {
4       public static void main(String[] args) {
5           /**
6            * 1.输出【1-6】之间的质数
7            * 2.什么是质数
8            *     1.除了1和本身外，不能被其他自然数整除
9            *     2.1 既不是质数，也不是合数
10           *     2.2是质数
11           * 3.拿到一个数，怎么知道它是不是质数?
12           *    1.除以比它小的数
13           *    2.除尽了说明不是质数
14           *    3.除不尽说明是质数
15           * 4.模拟:
16           *    3 % 2
17           *    4 % 2    4 % 3
18           *    5 % 2    5 % 3   5 % 4
19           */
20          //双层循环模拟: 3 % 2
21          for (int i = 3; i < 4; i++) {
22              for (int j = 2; j < 3; j++) {
23                  System.out.print(i + " % " + j);
24              }
25          }
26          System.out.println();
27          //4 % 2    4 % 3
28          for (int i = 4; i < 5; i++) {
29              for (int j = 2; j < 4; j++) {
30                  System.out.print(i + " % " + j + "\t");
31              }
32          }
33          System.out.println();
```

```
34            //5 % 2   5 % 3  5 % 4
35            for (int i = 5; i < 6; i++) {
36                for (int j = 2; j < 5; j++) {
37                    System.out.print(i + " % " + j + "\t");
38                }
39            }
40        }
41    }
```

3.6.2 代码优化

```
//双层循环模拟：3 % 2
for (int i = 3; i < 4; i++) {
    for (int j = 2; j < i; j++) {
        System.out.print(i + " % " + j);
    }
}
System.out.println();
//4 % 2   4 % 3
for (int i = 4; i < 5; i++) {
    for (int j = 2; j < i; j++) {
        System.out.print(i + " % " + j + "\t");
    }
}
System.out.println();
//5 % 2   5 % 3  5 % 4
for (int i = 5; i < 6; i++) {
    for (int j = 2; j < i; j++) {
        System.out.print(i + " % " + j + "\t");
    }
}
```

代码块

```
1   package com.powernode.for05;
2
3   public class Test02 {
4       public static void main(String[] args) {
5
6           /* //双层循环模拟：3 % 2
7           for (int i = 3; i < 4; i++) {
8               for (int j = 2; j < i; j++) {
9                   System.out.print(i + " % " + j);
10              }
11          }
12          System.out.println();
```

```
        //4 % 2   4 % 3
        for (int i = 4; i < 5; i++) {
            for (int j = 2; j < i; j++) {
                System.out.print(i + " % " + j + "\t");
            }
        }
        System.out.println();
        //5 % 2   5 % 3  5 % 4
        for (int i = 5; i < 6; i++) {
            for (int j = 2; j < i; j++) {
                System.out.print(i + " % " + j + "\t");
            }
        }*/
        for (int i = 3; i < 6; i++) {
            for (int j = 2; j < i; j++) {
                System.out.print(i + " % " + j + "\t");
            }
            System.out.println();
        }

    }
}
```

3.6.3 求质数算法

代码块

```
package com.powernode.for05;

public class Test03 {
    public static void main(String[] args) {

        /**
         * 1.输出【1-6】之间的质数
         * 2.什么是质数
         *    1.除了1和本身外，不能被其他自然数整除
         *    2.1 既不是质数，也不是合数
         *    2.2是质数
         * 3.拿到一个数，怎么知道它是不是质数?
         *   1.除以比它小的数
         *   2.除尽了说明不是质数
         *   3.除不尽说明是质数
         * 4.模拟:
         *   3 % 2
         *   4 % 2   4 % 3
         *   5 % 2   5 % 3  5 % 4
```

```
20            */
21           for (int i = 2; i <= 6; i++) {
22               //假设所有的i都是质数
23               boolean flag = true;
24               for (int j = 2; j < i; j++) {
25                   if (i % j == 0 ) {//不是质数
26                       //假设不成立
27                       flag = false;
28                       break;//结束内层循环
29                   }
30                   //System.out.print(i + " % " + j + "\t");
31               }
32               //输出质数
33               if (flag) {
34                   System.out.println("i = " + i);
35               }
36           }
37
38       }
39   }
```

3.7 break关键字

3.7.1 break结束内层循环

- break结束内层循环
- break后面不可以写语句，因为没有机会执行

代码块

```
1   package com.powernode.for05;
2
3   public class Test04 {
4       public static void main(String[] args) {
5           for (int i = 0; i < 1; i++) {
6               for (int j = 0; j < 5; j++) {
7                   System.out.println("======内层循环======");
8                   break;//结束的是内层循环
9                   //System.out.println("----------");Unreachable statement
10
11              }
12              System.out.println("======外层循环======");
13          }
14          System.out.println("======main方法======");
15
16
```

```
    }
}
```

3.7.2 break结束标记处循环

代码块

```
package com.powernode.for05;

public class Test05 {
    public static void main(String[] args) {
        aaa:for (int i = 0; i < 1; i++) {
            for (int j = 0; j < 5; j++) {
                System.out.println("======内层循环======");
                break aaa;//结束的是内层循环
            }
            System.out.println("======外层循环======");
        }
        System.out.println("======main方法======");


    }
}
```

3.8 continue关键字

代码块

```
package com.powernode.for06;

public class Test01 {

    public static void main(String[] args) {
        //输出【1-6】之间2的倍数
        for (int i = 1; i < 7; i++) {
            if (i % 2 == 0) {
                System.out.println("i = " + i);
            }
        }
        System.out.println("----------------");
        for (int i = 1; i < 7; i++) {
            if (i % 2 != 0) {//如果不是2的倍数
                continue;//结束当前当次循环，继续下一次循环
                //System.out.println("----");Unreachable statement
            }
```

```
            System.out.println("i = " + i);
        }


    }
}
```

3.9 九九乘法表的

3.9.1 模拟计算

代码块

```
package com.powernode.for07;

public class Test01 {
    public static void main(String[] args) {
        /**
         * 九九乘法表
         * 1×1=1
         * 1×2=2  2×2=4
         * 1×3=3  2×3=6  3×3=9
         * 1×4=4  2×4=8  3×4=12 4×4=16
         * 1×5=5  2×5=10 3×5=15 4×5=20 5×5=25
         */
        //1×1=1
        for (int i = 1; i <= 1 ; i++) {
            for (int j = 1; j <= 1 ; j++) {
                System.out.print(j + " * " + i + " = " + (j * i));
            }
        }
        System.out.println();
        //1×2=2  2×2=4
        for (int i = 2; i <= 2 ; i++) {
            for (int j = 1; j <= 2 ; j++) {
                System.out.print(j + " * " + i + " = " + (j * i) + "\t");
            }
        }
        System.out.println();
        //1×3=3  2×3=6  3×3=9
        for (int i = 3; i <= 3; i++) {
            for (int j = 1; j <= 3 ; j++) {
                System.out.print(j + " * " + i + " = " + (j * i) + "\t");
            }
        }
        System.out.println();
```

```
34          for (int i = 4; i <= 4; i++) {
35              for (int j = 1; j <= 4 ; j++) {
36                  System.out.print(j + " * " + i + " = " + (j * i) + "\t");
37              }
38          }
39          System.out.println();
40          for (int i = 5; i <= 5; i++) {
41              for (int j = 1; j <= 5 ; j++) {
42                  System.out.print(j + " * " + i + " = " + (j * i) + "\t");
43              }
44          }
45      }
46  }
```

3.9.2 代码优化

```
//1×1=1
for (int i = 1; i <= 1 ; i++) {
    for (int j = 1; j <= i ; j++) {
        System.out.print(j + " * " + i + " = " + (j * i));
    }
}
System.out.println();
//1×2=2  2×2=4
for (int i = 2; i <= 2 ; i++) {
    for (int j = 1; j <= i ; j++) {
        System.out.print(j + " * " + i + " = " + (j * i) + "\t");
    }
}
System.out.println();
//1×3=3  2×3=6  3×3=9
for (int i = 3; i <= 3; i++) {
    for (int j = 1; j <= i ; j++) {
        System.out.print(j + " * " + i + " = " + (j * i) + "\t");
    }
}
```

代码块

```
1  package com.powernode.for07;
2
3  public class Test02 {
4      public static void main(String[] args) {
```

```
5           /**
6            * 九九乘法表
7            * 1×1=1
8            * 1×2=2  2×2=4
9            * 1×3=3  2×3=6  3×3=9
10           * 1×4=4  2×4=8  3×4=12 4×4=16
11           * 1×5=5  2×5=10 3×5=15 4×5=20 5×5=25
12           */
13          //1×1=1
14          for (int i = 1; i <= 9 ; i++) {
15              for (int j = 1; j <= i ; j++) {
16                  System.out.print(j + " * " + i + " = " + (j * i) + "\t");
17              }
18              System.out.println();
19          }
20
21
22      }
23  }
```

4. 三种循环的区别

1. 通常情况下，明确循环次数时，使用for循环
2. 不明确循环次数时，使用while或者do-while
    a. do-while：至少执行一次
    b. while：至少执行0次

作业

1. 打印10行5列*矩形
    a. while
    b. do-while
    c. for
2. 使用do-while循环，打印1-100之间13的倍数。
3. 使用switch传统语法

4. 打印1-100之间13的倍数，使用continue关键字（for）
5. 打印100-200之间的质数（for）
6. 打印九九乘法表（for）
7. 把上课所有的代码敲一遍