第02天 java基础知识

今日内容介绍

* 运算符
* 键盘录入
* 选择流程控制语句

# 运算符

## 算术运算符的概述和用法

* 运算符

对常量和变量进行操作的符号称为运算符

* 表达式

用运算符把常量或者变量连接起来符号java语法的式子就可以称为表达式。不同运算符连接的式子体现的是不同类型的表达式。

定义两个int类型的变量a,b，做加法(a + b)

* 常用运算符

算术运算符

赋值运算符

关系运算符

逻辑运算符

三元运算符

### 算数运算符概述

A:什么是运算符

就是对常量和变量进行操作的符号。

B:算数运算符有哪些

\* +,-,\*,/,%,++,--

### 代码案例一

**package** com.itheima\_01;

/\*

\* 常用的运算符：

\* 算术运算符

\* 自增自减运算符

\* 赋值运算符

\* 关系运算符

\* 逻辑运算符

\* 三元运算符

\*

\* 算术运算符：

\* +,-,\*,/,%

\*

\* /和%的区别

\* /：获取两个数据相除的商

\* %：获取两个数据相除的余数

\*

\* 注意：

\* 整数相除只能得到整数。要想得到小数，就必须有浮点数参与运算。

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义两个int类型的变量

**int** a = 5;

**int** b = 3;

System.***out***.println(a+b);

System.***out***.println(a-b);

System.***out***.println(a\*b);

System.***out***.println(a/b);

System.***out***.println(a%b);

System.***out***.println("------------");

System.***out***.println(5/4);

System.***out***.println(5.0/4);

System.***out***.println(5/4.0);

}

}

**算数运算符取余和除法的区别：**

%：取余运算符。得到的是两个相除数据的余数。

/:除法运算符。得到是两个相除数据的商。

使用场景： %：判断两个数据是否整除。

### 字符和字符串参与加法运算

字符参与运算

其实是拿该字符对应的数值来操作

‘a’ 97

‘A’ 65

‘0’ 48

字符串参与运算

这里其实做的不是加法运算，而是字符串拼接。

字符串和其他类型的数据做拼接，结果是字符串类型的。

### 代码案例二

**package** com.itheima\_01;

/\*

\* 字符参与加法运算：其实是拿该字符在计算机中存储所表示的数据值来运算的。

\* 'a' 97

\* 'A' 65

\* '0' 48

\*

\* 字符串参与加法运算：其实这里不是加法，而是字符串的拼接。

\*/

**public** **class** OperatorDemo2 {

**public** **static** **void** main(String[] args) {

//定义两个变量，一个int类型，一个char类型

**int** a = 10;

**char** ch = 'a';

System.***out***.println(a + ch);

System.***out***.println("----------------");

//字符串做加法

System.***out***.println("hello"+"world");

System.***out***.println("hello"+10);

System.***out***.println("hello"+10+20);

System.***out***.println(10+20+"hello");

}

}

### 自增自减运算符的概述和用法

++,--运算符：对变量做加1或者减1的操作。

++或者--既可以放在变量的后面，也可以放在变量的前面。

单独使用的时候，++或者--无论是放在变量的前面还是后面，结果是一样的。

参与操作的时候：

如果++或者--在变量的后面，先拿变量参与操作，后变量做++或者--

如果++或者--在变量的前面，先变量做++或者--，后拿变量参与操作

### 代码案例三

**package** com.itheima\_02;

/\*

\* 自增自减运算符：++,--

\*

\* 作用：就是自己+1或者-1

\*

\* ++和--可以放在变量的前面，也可以放在变量的后面。

\* 单独使用一个变量的时候，放在变量的前面或者后面，效果一样。

\* 参与其他操作的时候：

\* ++在变量的后面，先把变量做操作，然后变量再++

\* ++在变量的前面，先变量++，然后在做操作

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义一个int类型的变量

**int** a = 10;

System.***out***.println("a:"+a);

//a++;

//a--;

//++a;

//--a;

//System.out.println("a:"+a);

//int b = a++;

**int** b = ++a;

System.***out***.println("a:"+a);

System.***out***.println("b:"+b);

}

}

## 赋值运算符的概述和用法

### 赋值运算符分类

基本的赋值运算符：=

扩展的赋值运算符：+=,-=,\*=,/=,%=

+=: a+=20;相当于a = (a的数据类型)(a + 20);

### 代码案例四

**package** com.itheima\_03;

/\*

\* 赋值运算符：

\* 基本的赋值运算符：=

\* 扩展的赋值运算符：+=,-=,...

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义变量

**int** a = 10; //把10赋值给int类型的变量a

System.***out***.println("a:"+a);

//扩展的赋值运算符：+=

//把运算符左边的数据和右边的数据进行运算，然后把结果赋值给左边

//a = a + 20;

a += 20;

System.***out***.println("a:"+a);

//short s = 1;

//s = s + 1;

//扩展的赋值运算符隐含了强制类型转换。

//a+=20

//等价于

//a =(a的数据类型)(a+20);

**short** s = 1;

s += 1;

System.***out***.println("s:"+s);

}

}

## 关系运算符的概述和使用

### 基本使用及注意事项

关系运算符包含以下内容:

==,!=,>,>=,<,<=

关系运算符的结果都是boolean型，也就是要么是true，要么是false。

注意事项:

关系运算符“==”不能误写成“=”。

### 代码案例五

**package** com.itheima\_04;

/\*

\* 关系运算符：

\* ==,!=,>,>=,<,<=

\* 关系运算符操作完毕的结果是boolean类型。

\*

\* 注意事项：

\* 千万不要把==写成了=

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义三个变量

**int** a = 10;

**int** b = 20;

**int** c = 10;

//==

System.***out***.println(a == b);

System.***out***.println(a == c);

System.***out***.println("------------");

//!=

System.***out***.println(a != b);

System.***out***.println(a != c);

System.***out***.println("------------");

//>

System.***out***.println(a > b);

System.***out***.println(a > c);

System.***out***.println("------------");

//>=

System.***out***.println(a >= b);

System.***out***.println(a >= c);

System.***out***.println("------------");

System.***out***.println(a == b);

System.***out***.println(a = b);//20,把b的值赋值给了a，并把a作为结果输出了

}

}

## 逻辑运算符的概述和使用

### 逻辑运算符概述

逻辑运算符用于连接关系表达式，在Java中不可以写成3<x<6，应该写成x>3 && x<6 。

A:逻辑运算符有哪些

&&,|| ！

B:结论：

&&:有false则false。

||:有true则true。

!:非false则true，非true则false。

特点：偶数个不改变本身。

### 代码案例六

**package** com.itheima\_05;

/\*

\* 逻辑运算符：

\* &&:有false则false

\* ||:有true则true

\* !:true则false,false则true

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义变量

**int** a = 3;

**int** b = 4;

**int** c = 5;

//&&逻辑与

System.***out***.println((a>b) && (a>c)); //false && false

System.***out***.println((a<b) && (a>c)); //true && false

System.***out***.println((a>b) && (a<c)); //false && true

System.***out***.println((a<b) && (a<c)); //true && true

System.***out***.println("------------");

//||逻辑或

System.***out***.println((a>b) || (a>c)); //false || false

System.***out***.println((a<b) || (a>c)); //true || false

System.***out***.println((a>b) || (a<c)); //false || true

System.***out***.println((a<b) || (a<c)); //true || true

System.***out***.println("------------");

//!逻辑非

System.***out***.println((a>b));

System.***out***.println(!(a>b));

System.***out***.println(!!(a>b));

}

}

## 三元运算符的概述和使用

### 三元运算符概述

A:格式

(关系表达式)?表达式1：表达式2；

如果条件为true，运算后的结果是表达式1；

如果条件为false，运算后的结果是表达式2；

### 代码案例七

**package** com.itheima\_06;

/\*

\* 三元运算符：

\* 关系表达式?表达式1:表达式2;

\*

\* 执行流程：

\* A:计算关系表达式的值，看结果是true还是false

\* B:如果是true，表达式1就是结果

\* 如果是false，表达式2就是结果

\*/

**public** **class** OperatorDemo {

**public** **static** **void** main(String[] args) {

//定义两个变量

**int** a = 10;

**int** b = 20;

**int** c = (a>b)?a:b;

System.***out***.println("c:"+c);

}

}

### 三元运算符练习之比较两个整数是否相同

**package** com.itheima\_06;

/\*

\* 需求：比较两个整数是否相同

\*/

**public** **class** OperatorTest {

**public** **static** **void** main(String[] args) {

//定义两个int类型的变量

**int** a = 10;

//int b = 20;

**int** b = 10;

**boolean** flag = (a==b)?**true**:**false**;

System.***out***.println(flag);

}

}

# 键盘录入

## 键盘录入的基本步骤

### 键盘录入数据概述

我们目前在写程序的时候，数据值都是固定的，但是实际开发中，数据值肯定是变化的，所以，把数据改进为键盘录入，提高程序的灵活性。

键盘录入数据的步骤:

A:导包(位置放到class定义的上面)

import java.util.Scanner;

B:创建对象

Scanner sc = new Scanner(System.in);

C:接收数据

int x = sc.nextInt();

### 代码案例八

**package** com.itheima;

**import** java.util.Scanner;

/\*

\* 为了提高程序的灵活性，我们把数据改进为键盘录入。

\* 如何实现键盘录入数据呢?目前使用JDK提供的类Scanner。

\* 如何使用Scanner来获取数据呢?目前大家记住使用步骤即可。

\*

\* 使用步骤：

\* A:导包

\* import java.util.Scanner;

\* 注意：在一个类中，有这样的顺序关系

\* package > import > class

\* B:创建键盘录入对象

\* Scanner sc = new Scanner(System.in);

\* C:获取数据

\* int i = sc.nextInt();

\*/

**public** **class** ScannerDemo {

**public** **static** **void** main(String[] args) {

//创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

//给出提示

System.***out***.println("请输入一个整数：");

//获取数据

**int** i = sc.nextInt();

//把获取的数据输出

System.***out***.println("i:"+i);

}

}

## Scanner录入数据练习之求两个整数的和

键盘录入两个数据，并对这两个数据求和，输出其结果

键盘录入：

A:导包

B:创建对象

C:接收数据

**package** com.itheima;

**import** java.util.Scanner;

/\*

\* 需求：键盘录入两个数据，并对这两个数据求和，输出其结果

\*/

**public** **class** ScannerTest {

**public** **static** **void** main(String[] args) {

//创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

//给出提示

System.***out***.println("请输入第一个整数：");

**int** a = sc.nextInt();

System.***out***.println("请输入第二个整数：");

**int** b = sc.nextInt();

//对数据求和

**int** sum = a + b;

//输出结果

System.***out***.println("sum:"+sum);

}

}

# 选择流程控制语句

## 选择流程控制语句概述

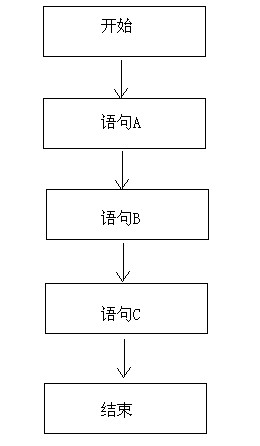
* 在一个程序执行的过程中，各条语句的执行顺序对程序的结果是有直接影响的。也就是说程序的流程对运行结果有直接的影响。所以，我们必须清楚每条语句的执行流程。而且，很多时候我们要通过控制语句的执行顺序来实现我们要完成的功能。
* 流程控制语句分类
  + 顺序结构
  + 选择结构
  + 循环结构

## 流程控制语句之顺序结构概述和使用

### 顺序结构概述

是程序中最简单最基本的流程控制，没有特定的语法结构，按照代码的先后顺序，依次执行，程序中大多数的代码都是这样执行的

### 顺序流程流程图



### 代码案例九

**package** com.itheima\_01;

/\*

\* 顺序结构：从上往下，依次执行

\*/

**public** **class** OrderDemo {

**public** **static** **void** main(String[] args) {

System.***out***.println("开始");

System.***out***.println("语句A");

System.***out***.println("语句B");

System.***out***.println("语句C");

System.***out***.println("结束");

}

}

## If语句的格式

### if语句格式1的概述和使用

if语句第一种格式：

if(关系表达式) {

语句体

}

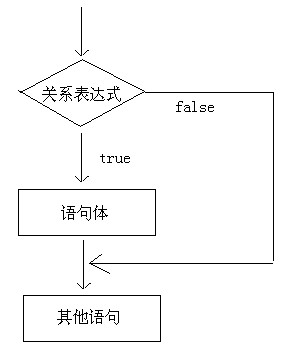
执行流程:

首先判断关系表达式看其结果是true还是false

如果是true就执行语句体

如果是false就不执行语句体

#### if语句格式1执行流程图



#### 代码案例十

**package** com.itheima;

/\*

\* if语句有三种格式。

\*

\* 格式1：

\* if(关系表达式) {

\* 语句体;

\* }

\*

\* 执行流程：

\* A:首先计算关系表达式的值，看是true还是false

\* B:如果是true，就执行语句体

\* C:如果是false，就不执行语句体

\*/

**public** **class** IfDemo {

**public** **static** **void** main(String[] args) {

System.***out***.println("开始");

//定义两个int类型的变量

**int** a = 10;

**int** b = 20;

//判断两个变量是否相等

**if**(a == b) {

System.***out***.println("a等于b");

}

//定义变量

**int** c = 10;

**if**(a == c) {

System.***out***.println("a等于c");

}

System.***out***.println("结束");

}

}

### if语句格式2的概述和使用

if语句第二种格式：

if(关系表达式) {

语句体1;

}else {

语句体2;

}

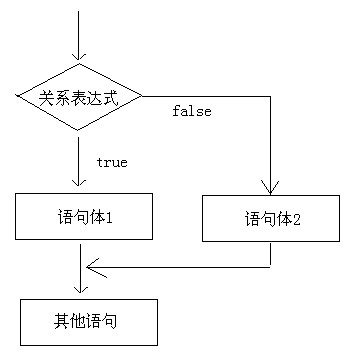
执行流程

首先判断关系表达式看其结果是true还是false

如果是true就执行语句体1

如果是false就执行语句体2

#### if语句格式2执行流程图



#### 代码案例十一

**package** com.itheima;

/\*

\* 格式2：

\* if(关系表达式) {

\* 语句体1;

\* }else {

\* 语句体2;

\* }

\*

\* 执行流程：

\* A:首先计算关系表达式的值，看结果是true还是false

\* B:如果是true，就执行语句体1

\* C:如果是false，就执行语句体2

\*/

**public** **class** IfDemo2 {

**public** **static** **void** main(String[] args) {

System.***out***.println("开始");

//判断一个数据是奇数还是偶数

//思路：如果一个数据对2取余的结果是0，说明该数是偶数

//定义一个变量

**int** a = 100;

//重新给a赋值

a = 99;

**if**(a%2 == 0) {

System.***out***.println("a是偶数");

}**else** {

System.***out***.println("a是奇数");

}

System.***out***.println("结束");

}

}

### if语句格式3的概述和使用

if语句第三种格式：

if(关系表达式1) {

语句体1;

}else if (关系表达式2) {

语句体2;

}

…

else {

语句体n+1;

}

执行流程

首先判断关系表达式1看其结果是true还是false

如果是true就执行语句体1

如果是false就继续判断关系表达式2看其结果是true还是false

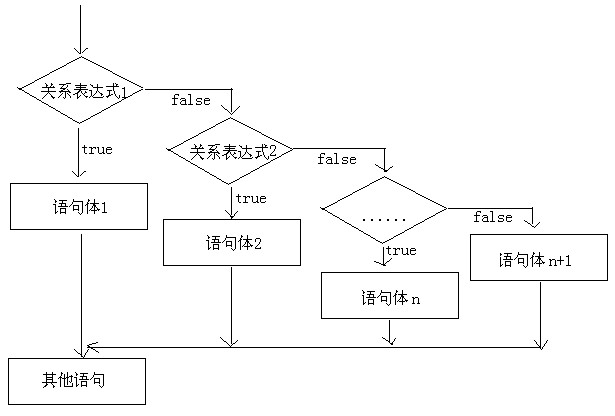
如果是true就执行语句体2

如果是false就继续判断关系表达式…看其结果是true还是false

…

如果没有任何关系表达式为true，就执行语句体n+1

#### if语句格式3执行流程图



#### 代码案例十二

**package** com.itheima;

/\*

\* 格式3：

\* if(关系表达式1) {

\* 语句体1;

\* }else if(关系表达式2) {

\* 语句体2;

\* }

\* ...

\* else {

\* 语句体n+1;

\* }

\*

\* 执行流程：

\* A:计算关系表达式1的值，看是true还是false

\* B:如果是true，就执行语句体1

\* C:如果是false，就继续计算关系表达式2的值，看是true还是false

\* D:如果是true，就执行语句体2

\* E:如果是false，就继续计算...

\* F:所有的关系表达式的结果都是false，执行语句体n+1

\*/

**public** **class** IfDemo3 {

**public** **static** **void** main(String[] args) {

System.***out***.println("开始");

//假如x和y满足如下的关系：

//x>=3 y=2x+1

//-1<=x<3 y=2x

//x<-1 y=2x-1

//根据给定的x的值，计算出y的值

**int** x = 5;

x = 0;

x = -5;

//定义变量y

**int** y;

**if**(x >= 3) {

y = 2\*x+1;

}**else** **if**(x>=-1 && x<3) {

y = 2\*x;

}**else** **if**(x<-1) {

y = 2\*x-1;

}**else** {

y = 0;

System.***out***.println("不存在这样的x的值");

}

System.***out***.println("y:"+y);

System.***out***.println("结束");

}

}

## If语句的练习

### if语句练习之获取两个整数的较大值

**package** com.itheima;

**import** java.util.Scanner;

/\*

\* 需求：键盘录入两个数据，获取这两个数据的较大值

\*

\* 分析：

\* A:看到了键盘录入，我们就应该想到键盘录入的三个步骤

\* 导包，创建键盘录入对象，获取数据

\* B:获取两个数据的较大值，用if语句的格式2就可以实现判断

\* C:把较大的结果输出即可

\*

\* 导包：

\* A:手动导包

\* B:点击鼠标自动生成

\* C:快捷键

\* ctrl+shift+o

\*/

**public** **class** IfTest {

**public** **static** **void** main(String[] args) {

//创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

//给出提示

System.***out***.println("请输入第一个整数：");

**int** a = sc.nextInt();

System.***out***.println("请输入第二个整数：");

**int** b = sc.nextInt();

//if格式2实现判断

/\*

if(a > b) {

System.out.println("较大的值是："+a);

}else {

System.out.println("较大的值是："+b);

}

\*/

//我们获取到较大的值未必是输出，可能还会进行其他的操作，所以我们要改进代码

//定义一个变量用于保存较大的值

**int** max;

**if**(a > b) {

max = a;

}**else** {

max = b;

}

//我可以做

//max += 10;

System.***out***.println("较大的值是："+max);

}

}

### if语句练习之根据学生成绩输出对应级别

**package** com.itheima;

**import** java.util.Scanner;

/\*

\* 键盘录入学生考试成绩，请根据成绩判断该学生属于哪个级别

\* 90-100 优秀

\* 80-90 好

\* 70-80 良

\* 60-70 及格

\* 60以下 不及格

\*

\* 分析：

\* A:键盘录入学生考试成绩,想到键盘录入数据的步骤

\* B:通过简单的分析，我们知道了该使用if语句的格式3进行判断

\* 根据判断直接输出对应的级别

\*

\* 写程序的时候，做数据测试，应该测试这样的几种情况：

\* 正确数据

\* 边界数据

\* 错误数据

\*/

**public** **class** IfTest2 {

**public** **static** **void** main(String[] args) {

//创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

//给个提示

System.***out***.println("请输入学生的考试成绩：");

**int** score = sc.nextInt();

//if语句格式3实现

/\*

if(score>=90 && score<=100) {

System.out.println("优秀");

}else if(score>=80 && score<90) {

System.out.println("好");

}else if(score>=70 && score<80) {

System.out.println("良");

}else if(score>=60 && score<70) {

System.out.println("及格");

}else {

System.out.println("不及格");

}

\*/

//通过数据的测试，我们发现程序不够严谨，未加入非法数据的判断

**if**(score>100 || score<0) {

System.***out***.println("你输入的成绩有误");

}**else** **if**(score>=90 && score<=100) {

System.***out***.println("优秀");

}**else** **if**(score>=80 && score<90) {

System.***out***.println("好");

}**else** **if**(score>=70 && score<80) {

System.***out***.println("良");

}**else** **if**(score>=60 && score<70) {

System.***out***.println("及格");

}**else** {

System.***out***.println("不及格");

}

}

}

## switch语句的格式和执行流程概述

### 格式解释:

switch语句格式：

switch(表达式) {

case 值1:

语句体1;

break;

case 值2:

语句体2;

break;

case 值3:

语句体3;

break;

...

default:

语句体n+1;

break;

}

switch表示这是switch语句

表达式的取值：byte,short,int,char

JDK5以后可以是枚举

JDK7以后可以是String

case后面跟的是要和表达式进行比较的值

语句体部分可以是一条或多条语句

break表示中断，结束的意思，可以结束switch语句

default语句表示所有情况都不匹配的时候，就执行该处的内容，和if语句的else相似。

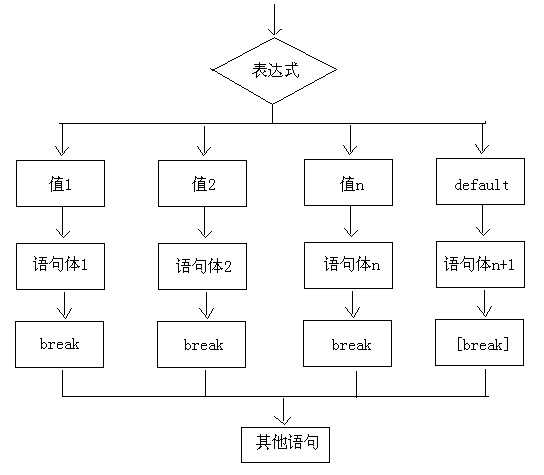
### 执行流程

首先计算出表达式的值

其次，和case依次比较，一旦有对应的值，就会执行相应的语句，在执行的过程中，遇到break就会结束。

最后，如果所有的case都和表达式的值不匹配，就会执行default语句体部分，然后程序结束掉。

### switch语句执行流程图



## switch语句的练习

### switch语句的案例根据数字输出对应星期

**package** com.itheima;

**import** java.util.Scanner;

/\*

\* switch语句格式：

\* switch(表达式) {

\* case 值1:

\* 语句体1;

\* break;

\* case 值2:

\* 语句体2;

\* break;

\* case 值3:

\* 语句体3;

\* break;

\* ...

\* default:

\* 语句体n+1;

\* break;

\* }

\*

\* 格式解释：

\* 表达式：byte,short,int,char

\* JDK5以后可以是枚举,JDK7以后可以是字符串

\* case后面的值：是用来和表达式的值进行匹配的

\* break：表示中断的意思

\* default：所有的值都和表达式不匹配，就执行default对应的内容

\*

\* 执行流程：

\* A:计算表达式的值

\* B:拿着这个值依次和case后面的值进行比对，一旦有匹配的，就执行对应的语句，在执行的过程中，遇到break就结束。

\* C:如果所有的case都不匹配，就执行语句体n+1

\*

\* 案例：

\* 根据键盘录入的数据1-7，输出对应的星期一到星期日

\*

\* 快捷键：对代码进行格式化

\* ctrl+shift+f

\*/

**public** **class** SwitchDemo {

**public** **static** **void** main(String[] args) {

// 创建键盘录入数据

Scanner sc = **new** Scanner(System.***in***);

// 给出提示

System.***out***.println("请输入一个整数(1-7)：");

**int** weekDay = sc.nextInt();

// 用switch语句实现判断

**switch** (weekDay) {

**case** 1:

System.***out***.println("星期一");

**break**;

**case** 2:

System.***out***.println("星期二");

**break**;

**case** 3:

System.***out***.println("星期三");

**break**;

**case** 4:

System.***out***.println("星期四");

**break**;

**case** 5:

System.***out***.println("星期五");

**break**;

**case** 6:

System.***out***.println("星期六");

**break**;

**case** 7:

System.***out***.println("星期日");

**break**;

**default**:

System.***out***.println("你输入的数据有误");

**break**;

}

}

}