**Java 条件语句 - if...else**

一个 if 语句包含一个布尔表达式和一条或多条语句。

**语法**

if 语句的语法如下：

if(布尔表达式) { //如果布尔表达式为true将执行的语句 }

如果布尔表达式的值为 true，则执行 if 语句中的代码块，否则执行 if 语句块后面的代码。

**Test.java 文件代码：**

public class Test { public static void main(String args[]){ int x = 10; if( x < 20 ){ System.out.print("这是 if 语句"); } } }

以上代码编译运行结果如下：

这是 if 语句

**if...else语句**

if 语句后面可以跟 else 语句，当 if 语句的布尔表达式值为 false 时，else 语句块会被执行。

**语法**

if…else 的用法如下：

if(布尔表达式){ //如果布尔表达式的值为true }else{ //如果布尔表达式的值为false }

**实例**

**Test.java 文件代码：**

public class Test { public static void main(String args[]){ int x = 30; if( x < 20 ){ System.out.print("这是 if 语句"); }else{ System.out.print("这是 else 语句"); } } }

以上代码编译运行结果如下：

这是 else 语句

**if...else if...else 语句**

if 语句后面可以跟 else if…else 语句，这种语句可以检测到多种可能的情况。

使用 if，else if，else 语句的时候，需要注意下面几点：

* if 语句至多有 1 个 else 语句，else 语句在所有的 else if 语句之后。
* if 语句可以有若干个 else if 语句，它们必须在 else 语句之前。
* 一旦其中一个 else if 语句检测为 true，其他的 else if 以及 else 语句都将跳过执行。

**语法**

if...else 语法格式如下:

if(布尔表达式 1){ //如果布尔表达式 1的值为true执行代码 }else if(布尔表达式 2){ //如果布尔表达式 2的值为true执行代码 }else if(布尔表达式 3){ //如果布尔表达式 3的值为true执行代码 }else { //如果以上布尔表达式都不为true执行代码 }

**实例**

**Test.java 文件代码：**

public class Test { public static void main(String args[]){ int x = 30; if( x == 10 ){ System.out.print("Value of X is 10"); }else if( x == 20 ){ System.out.print("Value of X is 20"); }else if( x == 30 ){ System.out.print("Value of X is 30"); }else{ System.out.print("这是 else 语句"); } } }

以上代码编译运行结果如下：

Value of X is 30

**嵌套的 if…else 语句**

使用嵌套的 if…else 语句是合法的。也就是说你可以在另一个 if 或者 else if 语句中使用 if 或者 else if 语句。

**语法**

嵌套的 if…else 语法格式如下：

if(布尔表达式 1){ ////如果布尔表达式 1的值为true执行代码 if(布尔表达式 2){ ////如果布尔表达式 2的值为true执行代码 } }

你可以像 if 语句一样嵌套 else if...else。

**实例**

**Test.java 文件代码：**

public class Test { public static void main(String args[]){ int x = 30; int y = 10; if( x == 30 ){ if( y == 10 ){ System.out.print("X = 30 and Y = 10"); } } } }

## break 关键字

break 主要用在循环语句或者 switch 语句中，用来跳出整个语句块。

break 跳出最里层的循环，并且继续执行该循环下面的语句。

### 语法

break 的用法很简单，就是循环结构中的一条语句：

break;

# Java switch case 语句

switch case 语句判断一个变量与一系列值中某个值是否相等，每个值称为一个分支。

### 语法

switch case 语句语法格式如下：

switch(expression){ case value : //语句 break; //可选 case value : //语句 break; //可选 //你可以有任意数量的case语句 default : //可选 //语句 }

switch case 语句有如下规则：

* switch 语句中的变量类型可以是： byte、short、int 或者 char。从 Java SE 7 开始，switch 支持字符串 String 类型了，同时 case 标签必须为字符串常量或字面量。
* switch 语句可以拥有多个 case 语句。每个 case 后面跟一个要比较的值和冒号。
* case 语句中的值的数据类型必须与变量的数据类型相同，而且只能是常量或者字面常量。
* 当变量的值与 case 语句的值相等时，那么 case 语句之后的语句开始执行，直到 break 语句出现才会跳出 switch 语句。
* 当遇到 break 语句时，switch 语句终止。程序跳转到 switch 语句后面的语句执行。case 语句不必须要包含 break 语句。如果没有 break 语句出现，程序会继续执行下一条 case 语句，直到出现 break 语句。
* switch 语句可以包含一个 default 分支，该分支一般是 switch 语句的最后一个分支（可以在任何位置，但建议在最后一个）。default 在没有 case 语句的值和变量值相等的时候执行。default 分支不需要 break 语句。

**switch case 执行时，一定会先进行匹配，匹配成功返回当前 case 的值，再根据是否有 break，判断是否继续输出，或是跳出判断。**

### 实例

## Test.java 文件代码：

public class Test { public static void main(String args[]){ //char grade = args[0].charAt(0); char grade = 'C'; switch(grade) { case 'A' : System.out.println("优秀"); break; case 'B' : case 'C' : System.out.println("良好"); break; case 'D' : System.out.println("及格"); break; case 'F' : System.out.println("你需要再努力努力"); break; default : System.out.println("未知等级"); } System.out.println("你的等级是 " + grade); } }

以上代码编译运行结果如下：

良好

你的等级是 C

如果 case 语句块中没有 break 语句时，JVM 并不会顺序输出每一个 case 对应的返回值，而是继续匹配，匹配不成功则返回默认 case。

## Test.java 文件代码：

public class Test { public static void main(String args[]){ int i = 5; switch(i){ case 0: System.out.println("0"); case 1: System.out.println("1"); case 2: System.out.println("2"); default: System.out.println("default"); } } }

以上代码编译运行结果如下：

default

如果 case 语句块中没有 break 语句时，匹配成功后，从当前 case 开始，后续所有 case 的值都会输出。

## Test. 以上代码编译运行结果如下：