逻辑分支

引入：我们在日常生活中，有很多事情是需要做判断的，比如说，去服装城买衣服，你看中一件衣服，老板要300元，你立马就会想，这个价格是否贵了，判断的结果只有两种可能，一是贵，二是不贵，贵了你就不买了，不贵你就买了。再比如去网吧，网管也要判断你是否满18岁，结果也只有两种，是和否，是就上网，不是就看别人上网，再比如，学校根据考试成绩对每个人进行评级，如果成绩大于60就合格，否则就不合格，如果成绩大于90，就优秀等等。。。在咱们的代码中，也会有很多判断，比如咱们做的练习，小红满足条件了，就能嫁人了，不满足条件就不能嫁人。咱们昨天只是能看到一个布尔值，并没有进行下一步的操作，通过今天的学习就可以进行下一步的操作了，咱们今天学习的主要内容就是判断，也叫做逻辑分支。

判断也会有很多种，比如：

- 考试成绩大于60，及格
- 考试成绩大于60，及格，否则，不及格
- 考试成绩如果大于60并且小于80，及格，如果大于80并且小于90，良好，如果大于90，优秀

根据上述几种情况，我们把判断分为三种，根据结果只做一个件事情的，叫单分支，做两件事情的，叫做双分支，做多件事情的，叫多分支。

单分支

语法：

```
if(条件表达式){
    当条件表达式的结果为true的时候要执行的代码
}
```

例：

```
var age;
if(age>18){
   alert("已经成年了");
}
```

双分支

语法：

```
if(条件表达式){
    当条件表示式的结果为true的时候要执行的代码
}else{
    当条件表达式的结果为false的时候要执行的代码
}
```

例：

```
if(age>18){
   alert("成年了");
}else{
    alert("未成年");
}
```

案例：输入年份，判断是否是闰年

多分支

语法：

```
if(条件表达式1){
    当条件表达式1的结果为true的时候，要执行的代码
}else if(条件表达式2){
    当条件表达式2的结果为true的时候，要执行的代码
}else if(条件表达式3){
    当条件表达式3的结果为true的时候，要执行的代码
}
。。。

}else{

}
# 多分支可以有若干个else if，else根据需要可以有也可以没有
```

例：

```
var score = Number(prompt("请输入考试成绩："));
if(isNaN(score)){
    alert("输入不正确");
}else{
    if(score>=0 && score<60){
       alert("不及格");
    }else if(score>=60 && score<70){
       alert("及格了");
    }else if(score>=70 && score<80){
        alert("丙");
    }else if(score>=80 && score<90){
        alert("乙");
    }else if(score>=90 && score<=100){
        alert("甲");
    }else{
        alert("输入不正确");
    }
}
```

分支结构的简写方式

如果单分支或双分支以及多分支的大括号中只有一行代码的时候，大括号可以省略。

例：

```
// 单分支的简写
var age;
if(age>18)
   alert("已经成年了");
// 双分支的简写
if(age>18)
   alert("成年了");
else
    alert("未成年");
// 多分支的简写
var score = Number(prompt("请输入考试成绩："));
var score = Number(prompt("请输入考试成绩："));
if(isNaN(score))
    alert("输入不正确");
else{
    if(score>=0 && score<60)
       alert("不及格");
    else if(score>=60 && score<70)
       alert("及格了");
    else if(score>=70 && score<80)
        alert("丙");
    else if(score>=80 && score<90)
        alert("乙");
    else if(score>=90 && score<=100)
        alert("甲");
    else
        alert("输入不正确");
}
```

分支结构的嵌套

例：定义三个变量，求出三个值中的最大值。

```
var a,b,c;
if(a>b){
    if(a>c){
       alert("变量a最大");
    }else{
        alert("变量c最大");
    }
}else{
    if(b>c){
       alert("变量b最大");
    }else{
        alert("变量c最大");
    }
}
```

switch多路判断

语法：

```
switch(变量){
    case 值1:
        执行的代码块
    break;
    case 值2:
        执行的代码块
    break;
    。。。
    default:
        执行代码块
}
```

使用说明：

1. break表示当前分支执行后就结束switch的运行，后续代码不再运行
2. default可以理解为判断语句中的else
3. case理解为if来判断这个变量是否全等于某个值

例：

```
var day = Number(prompt("请输入一个0~7之间数字："));
if(isNaN(day) || day<=0 || day>7){
    alert("输入错误");
}else{
    switch(day){
        case 1:
            alert("今天星期一");
        break;
        case 2:
            alert("今天星期二");
        break;
        case 3:
            alert("今天星期三");
        break;
        case 4:
            alert("今天星期四");
        break;
        case 5:
            alert("今天星期五");
        break;
        default:
            alert("输入错误");
    }
}
```

注意：使用多路判断的时候，能使用switch进行判断，就尽量使用switch，因为switch的判断都是确定的值，条件比较简单，所以效率高。

使用场景：

- 如果判断的条件比较复杂，使用if
- 条件是确定的值，分两种情况：
  - 如果判断的分支小于等于3个，就使用if，这时候的效率和switch一样
  - 如果大于3个，就使用switch。
- switch判断使用的是**全等于(===)**

switch 穿透写法

如果case后面不写break，那当前case执行后，会继续执行后面的case中的代码

输出1~5：

```
var day;
switch(day){
    case 1:
        alert("今天星期一");
    break;
    case 2:
        alert("今天星期二");
    break;
    case 3:
        alert("今天星期三");
    case 4:
        alert("今天星期四");
    break;
    case 5:
        alert("今天星期五");
    break;
    default:
        alert("输入错误");
}
```

当day的值为2的时候，不会执行后面的代码，当day的值为3的时候，执行完3的代码，还会执行4的代码，因为3的代码后面没有break。

这时候可以利用switch的这个特性，简写一些代码：

输入一个月份，判断并输出这个月有多少天？

```
var month = Number("请输入一个月份：");
if(isNaN(month) || month<=0 || month>12){
    alert("输入错误");
}else{
    switch(day){
        case 1:
        case 3:
```

```
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:
            alert("这个月有31天");
        break;
        case 4:
        case 6:
        case 9:
        case 11:
            alert("这个月有30天");
        break;
        case 2:
            alert("这个月有29天或28天");
        break;
        default:
            alert("输入错误");
    }
}
```

if条件中的细节

if条件的结果是布尔值，所以可以将布尔值当做条件放入if的小括号中

例：

```
if(true){
   alert("真的");
}else{
    alert("假的");
}
```

如果将别的非布尔值的表达式或数据放入条件小括号中，会发生隐形的类型转换

例：

```
if(2){ // 这里的2被转换成了布尔型
        console.log(2);
}
```

三元运算

if双分支有一种简写方式：

```
条件?条件成立时运行的代码段:条件不成立时运行的代码段;
```

例：

```
a>b?console.log(a):console.log(b);
```

这种表达双分支的方式叫做三元运算，也叫做三元表达式。

三元运算有个特点：可以将整个表达式当做一个值，可以赋值给一个变量，也可以输出

例：

```
var max = a>b?a:b; // 将a与b中较大的数赋值给变量max
console.log(a>b?a:b); // 将a与b中较大的数输出
```