

集训课-04：多分支语句

课程目标

01

三目运算符

02

逻辑运算符

03

多分支语句

04

switch语句

05

总结回顾



Part 1

三目运算符

三目运算符

输入两个整数，输出两个整数中最大的数

三目运算符

两个数中的最大值

```
int a, b, maxn;  
cin >> a >> b;  
if (a > b)  
{  
    maxn = a;  
}  
else  
{  
    maxn = b;  
}  
cout << maxn;
```

三目运算符

```
int a, b, maxn;  
cin >> a >> b;  
if (a > b)  
{  
    maxn = a;  
}  
else  
{  
    maxn = b;  
}  
cout << maxn;
```



```
int a, b, maxn;  
cin >> a >> b;  
maxn = a > b ? a : b;  
cout << maxn;
```

三目运算符

三目运算符：

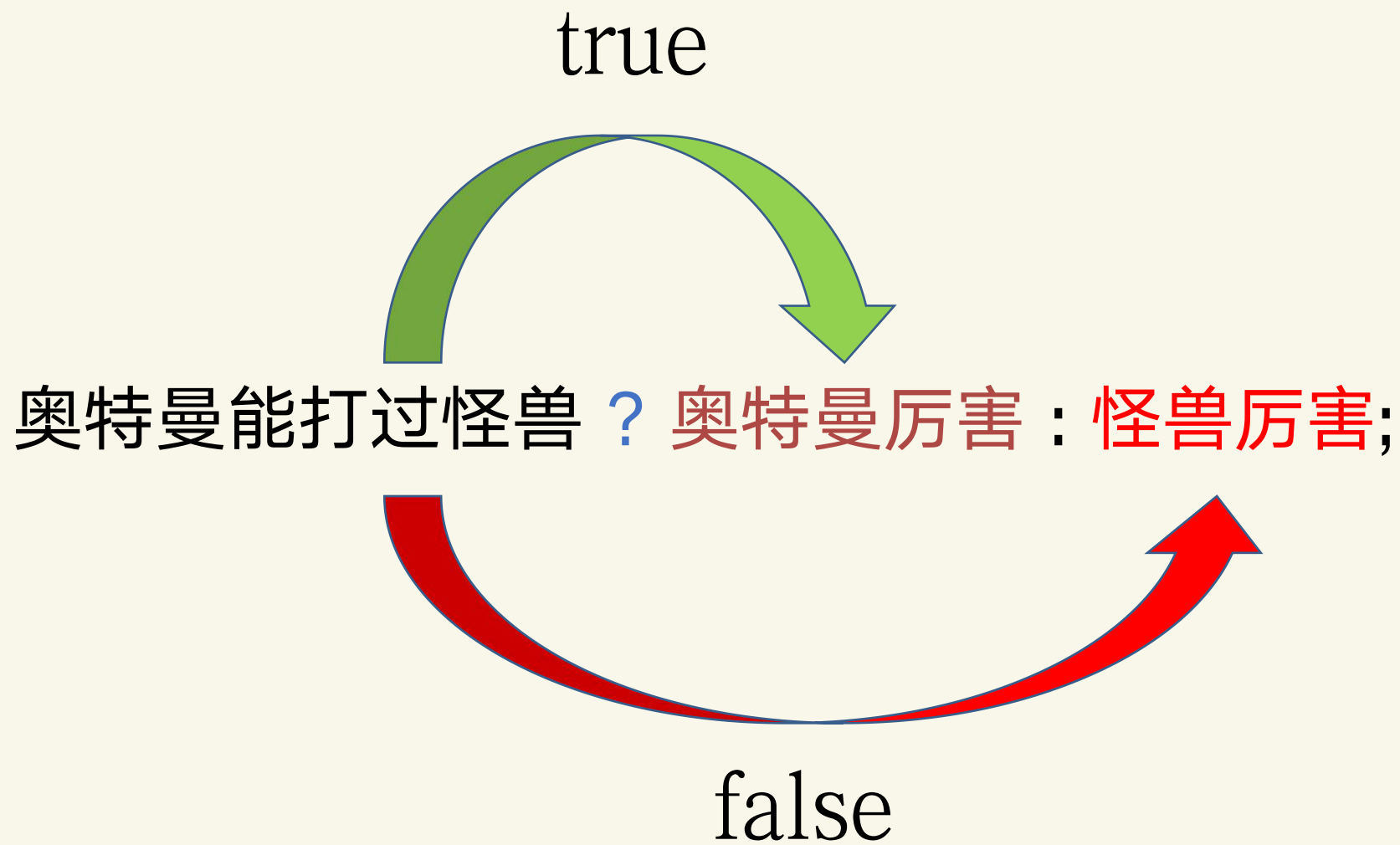
C++中唯一——一个需要三个操作数的操作符

三目运算符

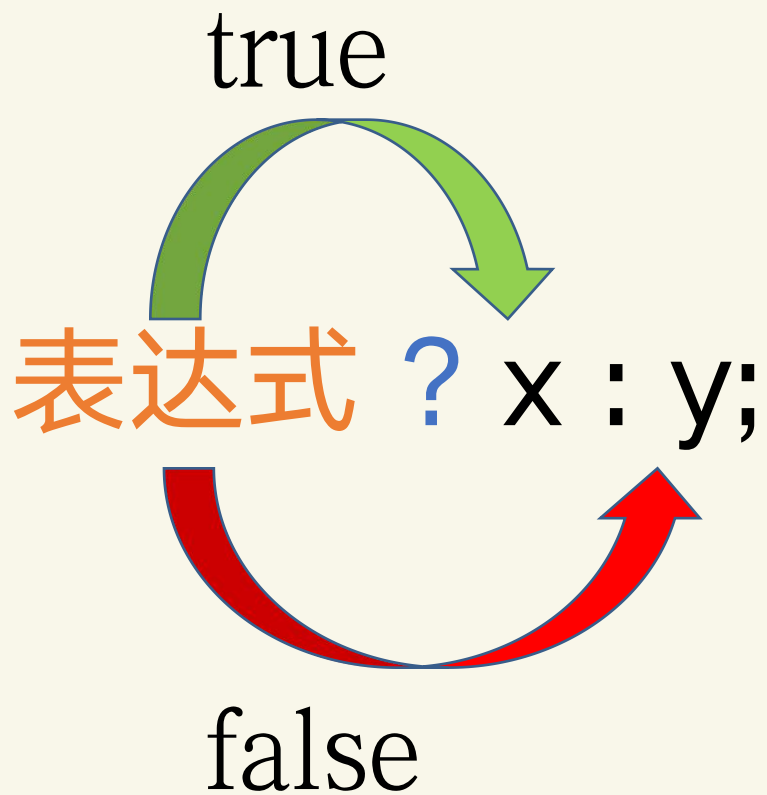
表达式 ? x : y;

例如：8 > 2 ? 8 : 2;

三目运算符



三目运算符



如果表达式结果为true

那么结果为x

否则结果为y

三目运算符

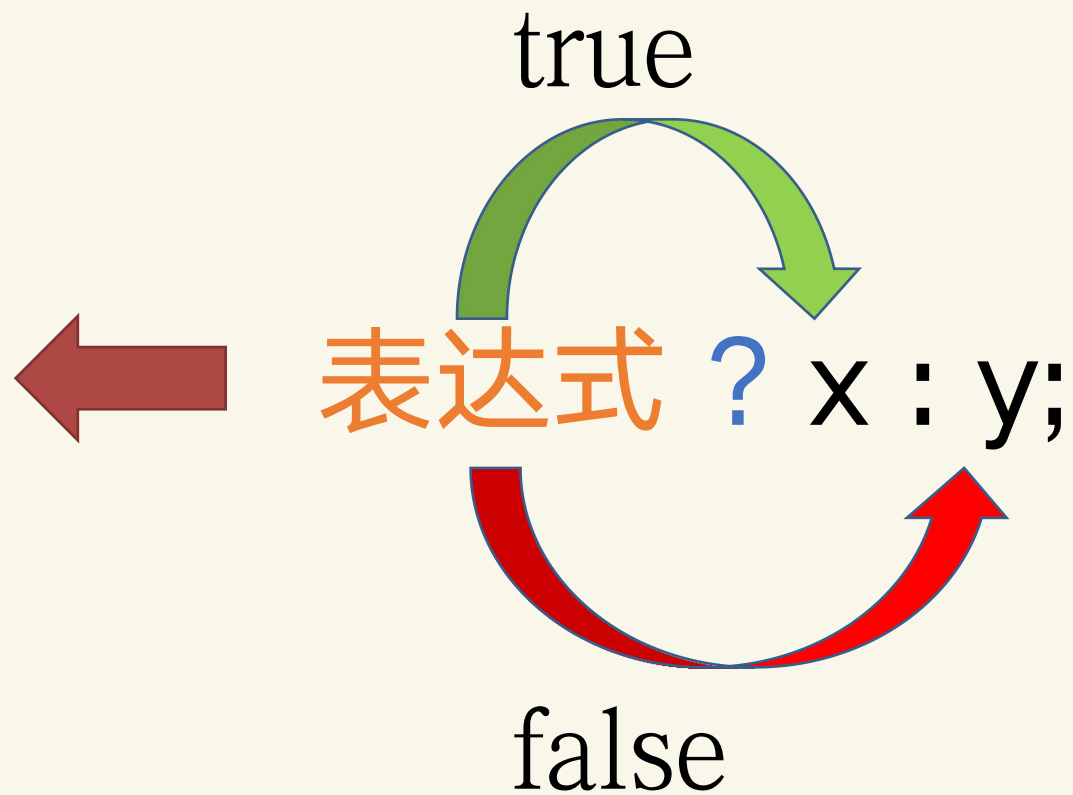
$5 > 3$

$4 \leq 2$

$12 \% 6 == 0$

$10 \% 5 != 0$

.....



三目运算符

想一想1

三目运算符 $5 > 10 ? 2022 : 2023$; 的结果为

A、 2022

B、 5

C、 2023



Part 2

逻辑运算符

逻辑运算符

如何判断一个数在14到16之间？

逻辑运算符

判断一个数在14到16之间

```
if (time >= 14)
{
    if (time <= 16)
    {
        cout << 1;
    }
}
```

逻辑运算符

★ 逻辑与运算：判断两个表达式是否都成立

(表达式1 && 表达式2)

运算符：&&

逻辑运算符

```
if (time >= 14 && time <= 16)
{
    cout << 1;
}
```

逻辑运算符

★ 1、逻辑与运算的结果为bool型

★ 2、&&真真为真，一假为假

逻辑运算符

想一想2

表达式1 && 表达式2		
表达式1	表达式2	结果
1 == 1	2 == 2	
1 == 1	2 == 3	
1 == 2	2 == 2	
1 == 2	2 == 3	

逻辑运算符

★ 逻辑或运算，判断表达式是否至少有一个成立

(表达式1 || 表达式2)

运算符： ||

逻辑运算符

★ 1、逻辑或运算的结果为bool型

★ 2、|| 有真为真，假假为假。

逻辑运算符

3或者7

```
if (a == 3 || a == 7)
{
    cout << 1;
}
```

逻辑运算符

想一想3

表达式1 表达式2		
表达式1	表达式2	结果
1 == 1	2 == 2	
1 == 1	2 == 3	
1 == 2	2 == 2	
1 == 2	2 == 3	

逻辑运算符

★ 逻辑非运算：与当前取反

!(表达式)

运算符：!

逻辑运算符

与当前条件取反

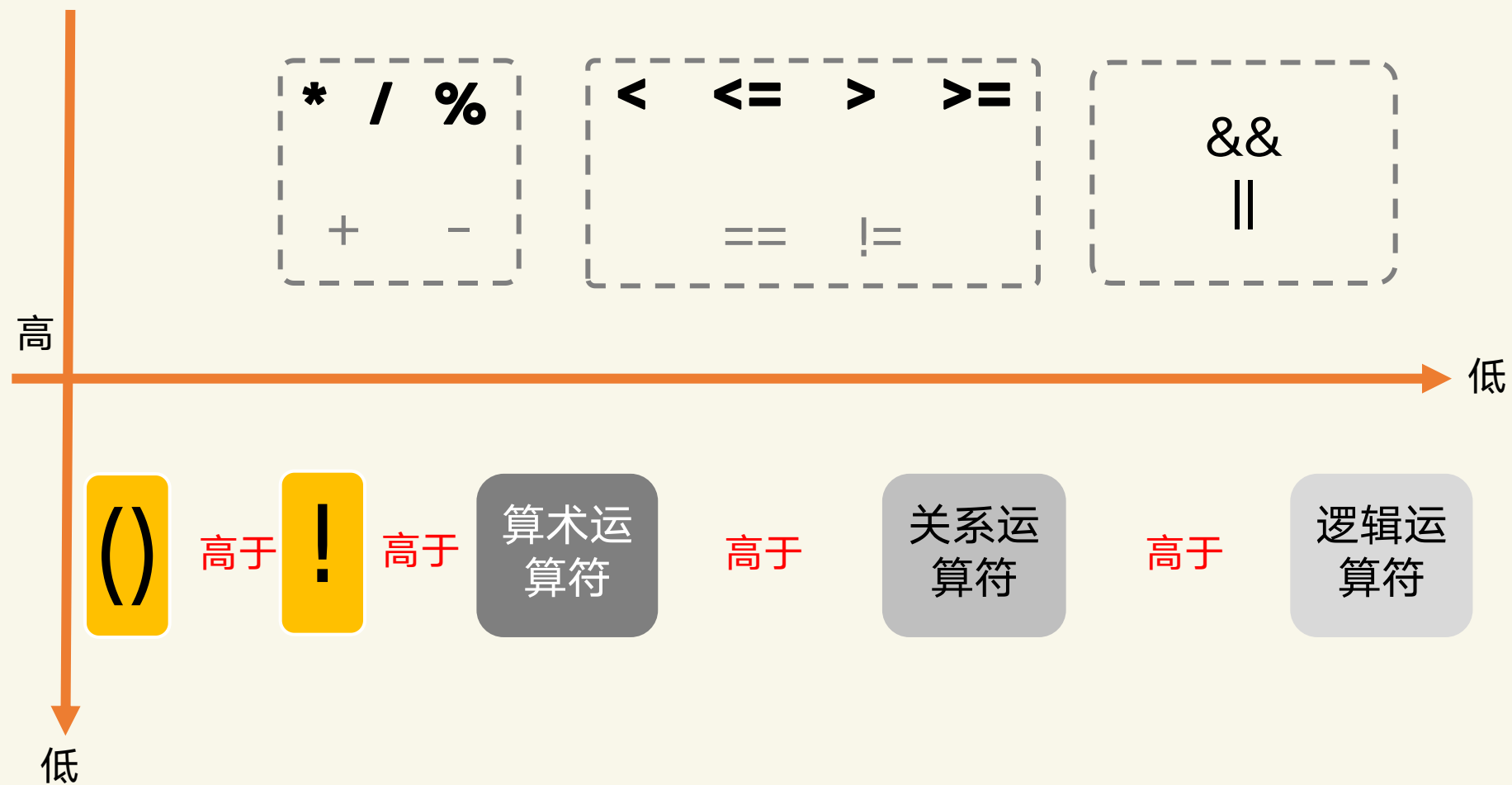
```
cout << !(1 > 2);
```

逻辑运算符

逻辑运算符优先级

! 高于 && 高于 ||

逻辑运算符



逻辑运算符

想一想4

```
if (10 > 5 || 3 < 4 && 8 <= 7)
{
    cout<< "嘻嘻";
}
else
{
    cout<< "哈哈";
}
```

左侧程序的运行结果是

A、嘻嘻

B、哈哈



Part 3

多分支语句

多分支语句



多分支结构

```
if ( 条件表达式1 )
{
    语句组1;
}
else if ( 条件表达式2 )
{
    语句组2;
}
else if ( 条件表达式n-1 )
{
    语句组n-1;
}
else
{
    语句组n;
}
```

多分支语句

★ 适用情况

多分支结构适用于一个问题有多种情况，
只有一种情况成立

多分支语句

★ 执行逻辑

- 1、从上到下执行
- 2、某个条件成立之后结束多分支
- 3、前面条件都不成立，执行else

多分支语句

```
int n;  
cin >> n;  
if (n >= 1 && n <= 40)  
{  
    cout << 1;  
}  
else if (n >= 41 && n <= 80)  
{  
    cout << 2;  
}  
else  
{  
    cout << 3;  
}
```

多分支语句

想一想5

```
int n;  
cin >> n;  
if (n >= 1 && n <= 40)  
{  
    cout << 1;  
}  
else if (n >= 41 && n <= 80)  
{  
    cout << 2;  
}  
else  
{  
    cout << 3;  
}
```

当输入75时，左侧程序的执行结果为

A、 1

B、 2

C、 3



Part 4

switch语句

switch语句



生活中

```
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int n;
7      cin >> n;
8      switch (n)
9      {
10         case 1: cout << "n"; break;
11         case 2: cout << "y"; break;
12         case 3: cout << "n"; break;
13         case 4: cout << "y"; break;
14         case 5: cout << "n"; break;
15         case 6: cout << "y"; break;
16         case 7: cout << "y"; break;
17         default: cout << "error";
18     }
19     return 0;
20 }
```

程序中

switch语句

switch语句

专门用来处理多分支结构的条件语句

又称开关语句

switch语句

switch (参数表)

{

case 常量1 : 语句组1; **break**;

case 常量2 : 语句组2; **break**;

.....

case 常量n : 语句组n; **break**;

default : 默认语句组n+1;

}

switch语句

```
switch (参数表)
{
    case 常量1 : 语句组1; break;
    case 常量2 : 语句组2; break;
    .....
    case 常量n : 语句组n; break;
    default : 默认语句组n+1;
}
```

参数表匹配到哪个常量，就执行对应后面的语句组

如果都不匹配，执行default后面的语句组

switch语句

想一想6

```
int n;  
cin >> n;  
switch (n)  
{  
    case 1 : cout << "一星"; break;  
    case 2 : cout << "二星"; break;  
    case 3 : cout << "三星"; break;  
    case 4 : cout << "四星"; break;  
    case 5 : cout << "五星"; break;  
    default : cout << "输入有误";  
}
```

程序输入5的时候，运行结果为

A、一星

B、五星

C、输入有误

switch语句

```
switch (参数表)
{
    case 常量1 : 语句组1;
    case 常量2 : 语句组2;
    .....
    case 常量n : 语句组n; break;
    default : 默认语句组n+1;
}
```

如果匹配成功的case后面没有遇到break

则会继续执行后续的case，直到遇到break结束

switch语句

```
switch (参数表)
{
    case 常量1:
    case 常量2:
    .....
    case 常量n: 语句组n; break;
    default: 默认语句组n+1;
}
```

常量后面的语句可以根据题目情况进行省略

switch语句

```
switch (参数表)
{
    case 常量1:
    case 常量2:
    .....
    case 常量n: 语句组n; break;
    default: 默认语句组n+1;
}
```

case后的常量根据情况可以不按照顺序

switch语句

想一想7

```
int n;  
cin >> n;  
switch (n)  
{  
    case 1: cout << "差评";  
    case 2: cout << "差评"; break;  
    case 3: cout << "中评";  
    case 4: cout << "中评"; break;  
    case 5: cout << "好评";  
    case 6: cout << "好评"; break;  
    default: cout << "输入有误";  
}
```

左侧程序输入5的时候，运行结果为

A、好评

B、好评好评

C、输入有误

switch语句

想一想8

```
int n;  
cin >> n;  
switch (n)  
{  
    case 1 :  
    case 2 : cout << "差评"; break;  
    case 3 :  
    case 4 : cout << "中评"; break;  
    case 5 :  
    case 6 : cout << "好评"; break;  
    default : cout << "输入有误";  
}
```

左侧程序输入5的时候，运行结果为

A、好评

B、好评好评

C、输入有误

switch语句

参数表不能是浮点数类型

case后面的常量不可重复



Part 5

总结回顾

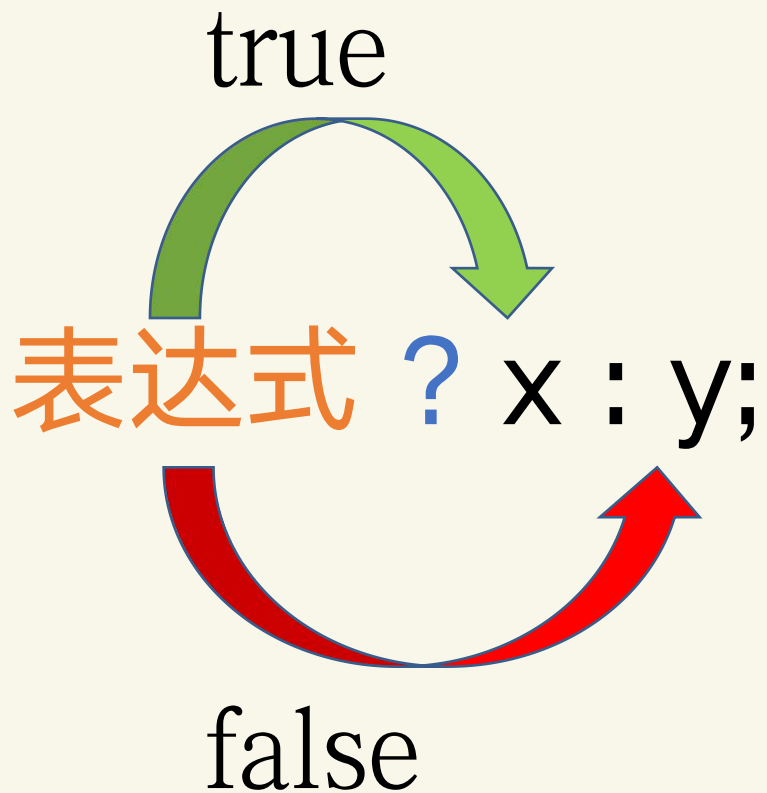
总结回顾

符号	逻辑运算
&&	逻辑与 真真为真，一假为假
	逻辑或 有真为真，假假为假
!	非、取反

优先级： ! 高于 && 高于 ||

总结回顾

三目运算符



如果表达式结果为true
那么结果为x
否则结果为y

总结回顾

多分支结构

```
if ( 条件表达式1 )
{
    语句组1;
}
else if ( 条件表达式2 )
{
    语句组2;
}
else if ( 条件表达式n-1 )
{
    语句组n-1;
}
else
{
    语句组n;
}
```

```
switch (参数表)
{
    case 常量1 : 语句组1; break;
    case 常量2 : 语句组2; break;
    .....
    case 常量n : 语句组n; break;
    default : 默认语句组n+1;
}
```

总结回顾

switch语句执行逻辑

参数表匹配到哪个常量，就执行对应后面的语句组，如果都不匹配，执行default后面的语句组

如果匹配成功的case后面没有遇到break，则会继续执行后续的case，直到遇到break结束

常量后面的语句可以根据题目情况进行省略

case后的常量根据情况可以不按照顺序

A horizontal yellow banner with blue triangular ends pointing outwards, centered on a solid blue background.

Class is over