



第三讲 选择与循环

潘建瑜@MATH.ECNU



1

关系运算与逻辑运算

2

选择结构

3

循环结构

1

关系运算与逻辑运算

- 关系运算（比较大小）
- 逻辑运算
- 条件运算符

关系运算：比较大小

>	大于	<	小于	==	等于
>=	大于等于	<=	小于等于	!=	不等于

- ▶ 比较大小，结论是 真 则返回 `true`，否则返回 `false`
- ▶ C++ 中用 `1` 表示 `true`，`0` 表示 `false`
- ▶ `bool` 型变量，赋值 `0` 时表示 `false`，其他它值都表示 `true`

```
bool x1 = 1.5;    // x1 = true
bool x2 = 0;      // x2 = false
bool x3 = -11;    // x3 = true
```

ex03_bool_02.cpp

注意 `==` 与 `=` 的区别；对浮点数进行比较运算时尽量不要使用 `==`

逻辑运算

&&	逻辑与		逻辑或	!	逻辑非
----	-----	--	-----	---	-----

运算法则

运算对象		与	或	非
A	B	A&&B	A B	!A
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

逻辑运算：两点注意

(表达式1) && (表达式2)

先计算 表达式1 的值，若是 **true**，则计算 表达式2 的值；
若 表达式1 的值是 **false**，则不再计算 表达式2 的值。

(表达式1) || (表达式2)

先计算 表达式1 的值，若是 **false**，则计算 表达式2 的值；
若 表达式1 的值是 **true**，则不再计算 表达式2 的值

```
i = 3; (i > 3) && (i++);
```

➡ i = 3

```
i = 3; (i > 3) || (i++);
```

➡ i = ?

条件运算符

条件表达式 ? 表达式1 : 表达式2

- ▶ 先计算 条件表达式 的值，若是 true，则用 表达式1 作为整个表达式的值，否则就用 表达式2 的值作为整个表达式的值

```
z = x > 0 ? 1 : -1;
```

```
z = x > y ? x : y;
```

条件运算符是 C++ 中唯一的 三目 运算符

举例

```
#include <iostream>
using namespace std;
int main()
{
    int x1 = 1, x2 = 2, x3 = 3;
    bool b1, b2, b3;

    b1 = (x1 < x2) && (x2 < x3);    // 正确写法
    cout << "b1 = " << b1 << "\n";

    b2 = x1 < x2 < x3;              // 有没有问题?
    cout << "b2 = " << b2 << "\n";

    b3 = x3 > x2 > x1;              // 有没有问题?
    cout << "b3 = " << b3 << "\n";

    return 0;
}
```

2

选择结构

- IF 语句：单分支、双分支、多分支、嵌套
- SWITCH 语句

IF 语句：单分支

用法一：单分支

if (表达式) 语句块

```
if (x > y) cout << x;
```

这里的语句块可以是单个语句或 **复合语句** (即多个语句用 **大括号** 括起来)

```
if (x > y)
{
    t = x;
    x = y;
    y = t;
}
```

```
if (x > y)
    t = x;
    x = y;
    y = t;
```

ex03_if_01.cpp

IF 语句：双分支

用法二：双分支

```
if (表达式)  
    语句块  
  
else  
    语句块
```

ex03_if_02.cpp

```
if (x > y)  
    cout << x;  
else  
    cout << y;
```

注意：条件判断表达式两边的 **小括号** 不能省略！

IF 语句：多分支

用法三：多分支

```
if (表达式)
    语句块
else if (表达式)
    语句块
... // else if 可以有多个
else if (表达式)
    语句块
else
    语句 // else 语句可以没有
```

ex03_if_03.cpp

```
if (x > 0)
    f=1;
else if (x < 0)
    f=-1;
else
    f=0;
```

IF 语句：嵌套

if (表达式)

if (表达式)

语句块

else

语句块

else

if (表达式)

语句块

else

语句块

ex03_if_04.cpp

每一层的 if 要与 else 配套；

如果没有 else，必须将该层的 if 语句用 {} 括起来

SWITCH 语句

```
switch (表达式)
```

```
{
```

```
    case 常量表达式1: 语句块
```

```
    case 常量表达式2: 语句块
```

```
        ... ..
```

```
    case 常量表达式n: 语句块
```

```
    default: 语句块
```

```
}
```

表达式结果可以是整型、字符型

将 表达式 的值与 case 后的 常量表达式 的值进行匹配，如果一致，则执行其后面的语句，包括该case后面所有case和default后的语句

- ▶ 每个 case 分支最后一般需要加 break 语句
- ▶ 每个 case 后面的常量表达式的值不能相同
- ▶ 每个 case 后面可以是复合语句（可以不加 {}）

ex03_switch.cpp

3

循环结构

- while 循环
- do while 循环
- for 循环
- 循环的终止

WHILE 循环

while (条件表达式)
循环体语句块

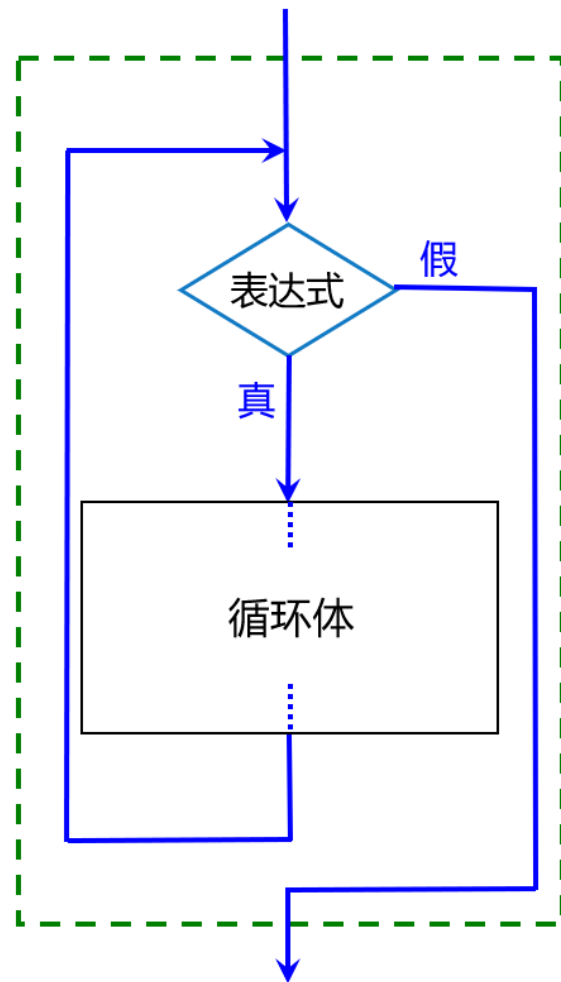
❑ 执行过程

- (1) 判断条件表达式的值
- (2) 如果是“真”，则执行循环体语句，否则退出循环
- (3) 返回第一步

```
int i = 1, s = 0;  
while (i <= 10)  
    s += i;  
    i++;
```

ex03_while.cpp

如果循环体语句是 **复合语句**，别忘了 **大括号**！



WHILE 循环

DO WHILE 循环

do

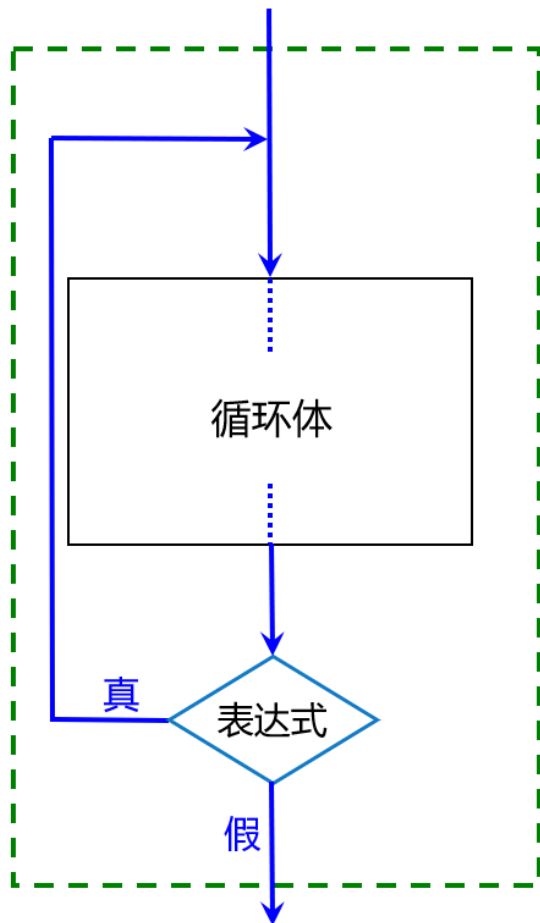
循环体语句块 (通常是复合语句)

while (条件表达式);

■ 执行过程

ex03_do_while.cpp

- (1) 执行循环体语句
- (2) 判断条件表达式的值
- (3) 如果是“真”，返回第一步，否则退出循环



DO WHILE 循环

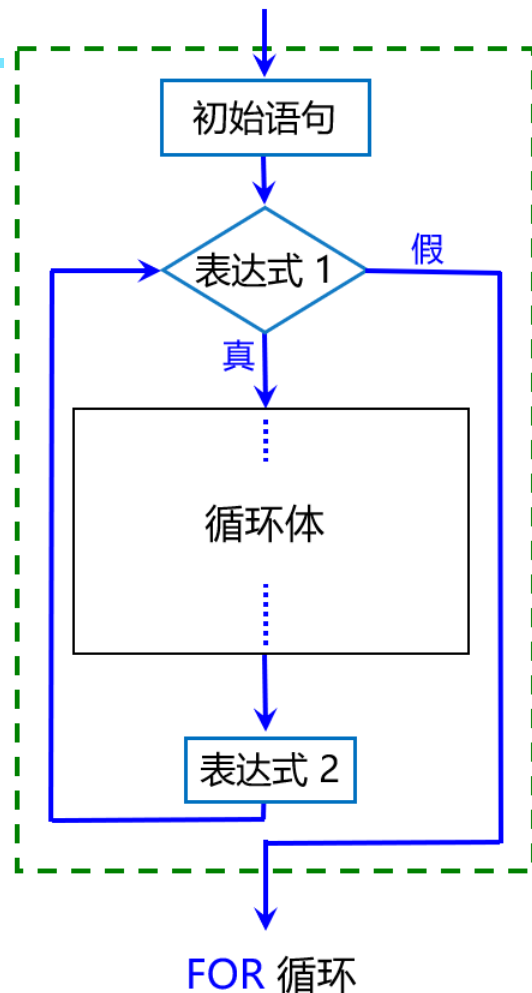
与 while 循环的区别：无论条件是否成立，循环体语句 **至少执行一次**！

FOR 循环

```
for (初始语句; 表达式1; 表达式2)  
    循环体语句
```

□ 执行过程

- (1) 执行 初始语句
- (2) 计算 表达式1 的值,
如果是“真”, 则执行循环体语句, 否则退出循环
- (3) 执行表达式2, 返回第二步;



FOR 循环

几点注记

- ▶ 初始语句, 表达式1, 表达式2 均可省略, 但分号不能省
- ▶ 表达式1 是循环控制语句, 如果省略的话就构成 **无限循环**
- ▶ 循环体可以是单个语句, 也可以是 **复合语句** (大括号!)
- ▶ 初始语句 与 表达式2 可以是逗号表达式

```
s = 0;  
for (i = 1; i <= 10; i++)  
    s += i;
```

```
for (s = 0, i = 1; i <= 10; i++)  
    s += i;
```

□ 代码中的变量 **i** 有时也称为**循环变量**

for (循环变量赋初值; 循环条件; 循环变量增量)
 循环体语句块

FOR 循环

若省略初始语句和表达式2，只有表达式1，则等同于 while 循环，故 while 可以用 for 替代

```
i = 1; s = 0;
while (i <= 10)
{
    s += i;  i++;
}
```



```
i = 1; s = 0;
for ( ; i <= 10; )
{
    s += i;  i++;
}
```

例：输出一个正整数的所有因子。

ex03_for_factor.cpp

FOR 循环

循环变量可以在初始语句中声明，此时循环变量只在该循环内有效，循环结束后，循环变量即被释放。

```
s = 0;  
for (auto i = 1; i <= 10; i++)  
    s += i;  
cout << i << endl; // ERROR
```

关于变量的 作用域 和 生存期，详见后面的讲义。

FOR 循环：基于范围 (C++11)

for (声明: 取值范围)
 循环体语句

□ 执行过程

按顺序遍历所有取值，每取一次值就执行一次循环体。

```
s = 0;  
for (auto x: {1,3,5,9})  
    s += x;  
cout << s << endl;
```

[ex03_for_range.cpp](#)

循环的终止

❑ break 语句

- ▶ **跳出循环体**，可用在循环语句和 switch 语句中
- ▶ break 只能跳出一层循环

```
break;
```

❑ continue 语句

- ▶ **结束本轮循环**，执行下一轮循环，一般只用在循环语句中

```
continue;
```

❑ goto 语句

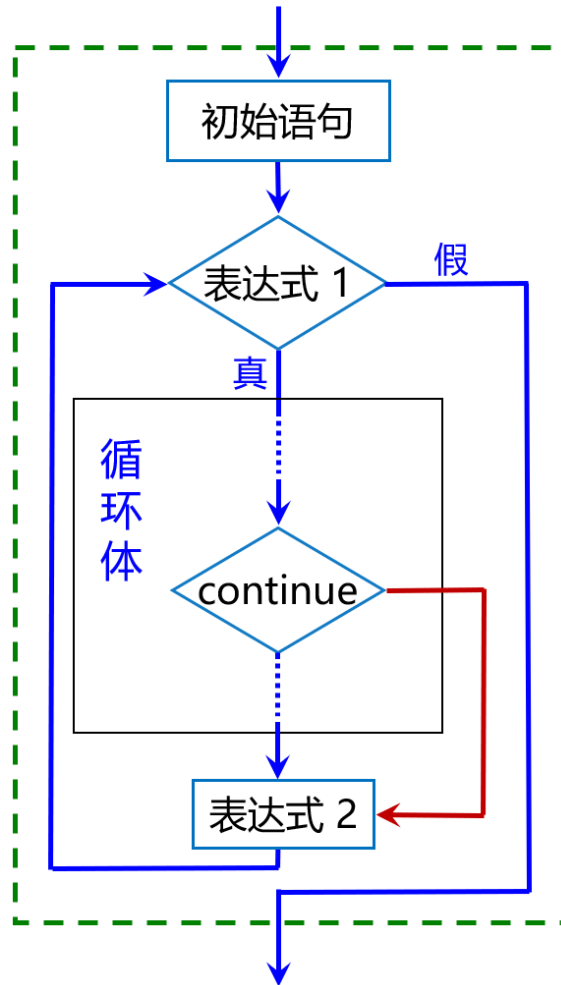
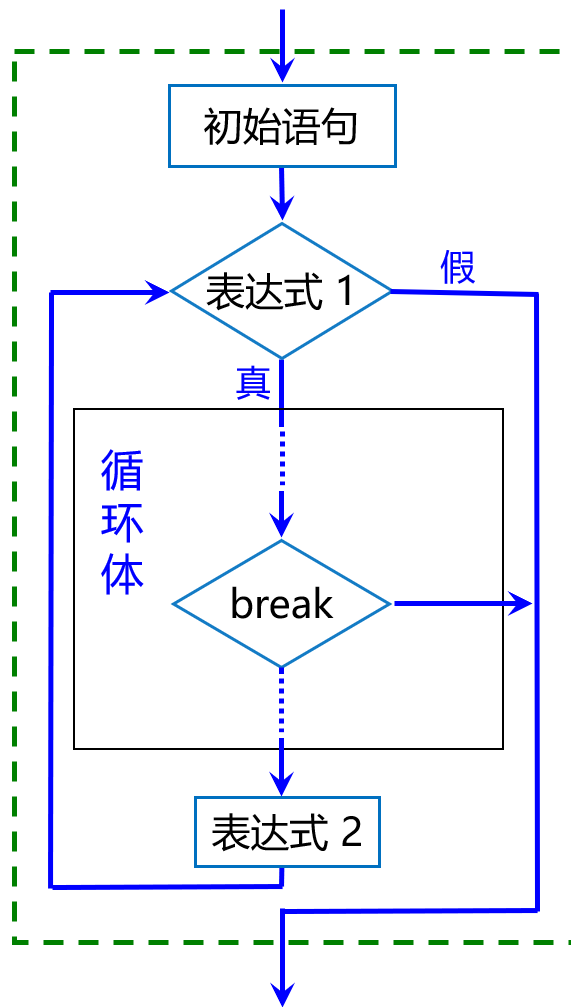
ex03_goto.cpp

- ▶ 跳转到由语句标号所指定的语句
- ▶ **语句标号**为标记符后面跟冒号，即 **标记符: 语句;**
- ▶ goto 语句会破坏程序的结构性，**不建议使用**

```
goto 语句标号;
```

break, continue 和 goto
通常与 if 语句配合使用

BREAK 和 CONTINUE (以 FOR 循环为例)



循环的嵌套

□ 循环可以嵌套使用

例：输出九九乘法表

[ex03_for_99.cpp](#)

循环举例

例：判断一个数是否为素数

`ex03_for_prime.cpp`

例：计算最大公约数。

`ex03_for_gcd.cpp`

`ex03_for_gcd_new.cpp`

循环举例

例：猜生日。 请回答你的生日出现在下面五组数的哪几组中？
这几组数的第一个数字之和就是你的生日

16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

8 9 10 11 12 13 14 15 24 25 26 27 28 29 30 31

4 5 6 7 12 13 14 15 20 21 22 23 28 29 30 31

2 3 6 7 10 11 14 15 18 19 22 23 26 27 30 31

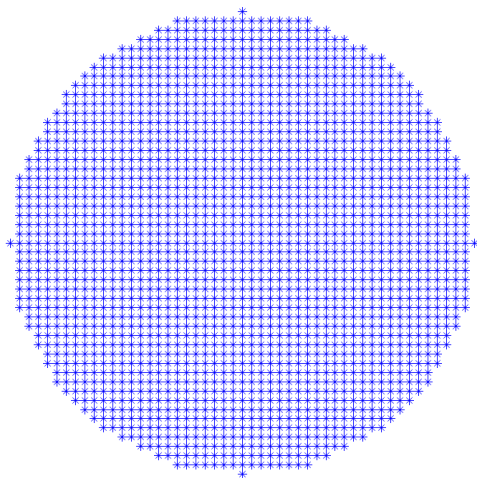
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31

ex03_birthday.cpp

循环举例

例：打印圆周图案

`ex03_plot.cpp`



应用：定积分的数值计算方法

见 “应用举例：定积分数值计算”

第三讲上机作业

1、按下面的要求编写四个程序

学生的成绩有两种表示方法：等级制和百分制。对应关系如下：

等级	A	B	C	D	E
分数	90--100	80--89	70--79	60--69	0--59

请编写程序，实现它们之间的互换，即

(a) 要求从键盘输入一个分数，然后输出其对应的等级。试分别用 `if` 语句和 `switch` 语句实现。

(程序取名 `hw03_if_1.cpp`, `hw03_switch_1.cpp`)

(b) 从键盘输入一个等级，然后输出其对应的分数区间。试分别用 `if` 语句和 `switch` 语句实现。

(程序取名 `hw03_if_2.cpp`, `hw03_switch_2.cpp`)

† 注意：输入用 `cin`，要有提示，如“请输入成绩（百分制）：”

2、(跳出多重循环)

修改程序 `ex03_goto.cpp`，避免使用 `goto` 语句。（程序取名 `hw03_02.cpp`）

第三讲上机作业

3、(最大素数)

从键盘输入一个大于 1 的整数（需要判断输入的合法性，即是否大于 1），求出不超过这个整数的最大素数，然后在屏幕上输出。

（要求使用 **for** 循环和 **if** 语句，程序取名 **hw03_03.cpp**）

4、(消去误差，大数吃小数)

当两个数量级相差很大的数进行加减运算时，由于计算机的舍入误差，小数可能被大数“吃掉”，如 **10000000000+0.0000000001** 的结果是 **10000000000**。因此在做计算时应尽可能避免这种情况发生。例如，计算下面级数时，由右至左计算会比由左至右计算获得更精确的结果：

$$S = 10^{14} + 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n}$$

编写程序，分别从右至左和从左至右计算上面级数的和，取 **n=50000**。

（输出小数点后 6 位，分别用 **float** 和 **double**，看看结果会怎样，程序取名 **hw03_04.cpp**）

第三讲上机作业

5、(常数 e 的计算)

用下面的级数部分和来近似常数 e

$$e = 2.718281828459045235\dots$$

$$e \approx 1 + \frac{1}{1!} + \frac{1}{2!} + \dots + \frac{1}{n!}$$

编写程序，输出当 $n=30$ 时的结果。

(用 `long double`，输出小数点后 18 位，程序取名 `hw03_05.cpp`)

6、(闰年)

编写程序，输出 21 世纪（2000 至 2099 年）中所有的闰年，即满足下面条件之一的年份：

(1) 能被 400 整除；(2) 能被 4 整除但不能被 100 整除。

要求每行输出 6 个，闰年之间用两个空格隔开。（程序取名 `hw03_06.cpp`）

第三讲上机作业

7、(定积分近似计算)

分别用复合梯形法和复合抛物线法计算定积分

$$\int_0^{\pi/2} \sin x \, dx$$

编写程序，输出积分区间 20 等分时的结果。

(程序取名 `hw03_07.cpp`)

8、(最大正整数与最小正实数)

编写程序，找出计算机所能表示的最大 `short` 型正整数和最小的 `float` 型正实数，并在屏幕上输出。

(用指数形式输出最小 `float` 型正实数，保留 8 位有效数字，程序取名 `hw03_08.cpp`)