高级语言C++程序设计
Lecture 4 控制语句

李雨森
南开大学 计算机学院
2021

条件分支语句

有时，并非所有的程序语句都要被顺序执行到，会希望满足某种条件就执行这部分语句，满足另一条件就执行另一部分语句。这就需要“条件分支语句”

if 语句

```
if (表达式1) {
      语句组1;
}
```

表达式1如果是true，执行花括号中的语句组，否则花括号中的语句都不执行

```
#include<iostream>
using namespace std;
int main(){
      int x = 1; y = 2;
      if(x > y) {
            cout<<x<<endl;
      }
      return 0;
}
```

如果x>y成立，输出x的值

if 语句

```
if (表达式1) {
      语句组1;
} else {
      语句组2;
}
```

表达式1如果是true，执行语句组1，否则执行语句组2

如果x>y成立，输出x的值，否则输出y的值

```
#include<iostream>
using namespace std;
int main(){
      int x = 1; y = 2;
      if(x > y) {
            cout<<x<<endl;
      } else {
            cout<<y<<endl;
      }
      return 0;
}
```

if 语句

```
if (表达式1) {
        语句组1;
} else if (表达式2) {
        语句组2;
}
.... //可以有多个else if
else if (表达式n-1) {
        语句组n-1;
}  else {
        语句组n;
}
```

依次计算表达式1，表达式2...，只要碰到一个表达式i为true，则执行语句组i（前面为false的表达式对应的语句组不会被执行），后面的表达式不再计算，若所有表达式都为false，则执行语句组n

if 语句

```
if (表达式1) {
        语句组1;
} else if (表达式2) {
        语句组2;
}
.... //可以有多个else if
else if (表达式n-1) {
        语句组n-1;
}  else {
        语句组n;
}
```

```
#include<iostream>
using namespace std;
int main(){
      int x = 1; y = 2;
      if(x > y) {
          cout<<x<<endl;
      } else if(x < y) {
          cout<<y<<endl;
      } else {
          cout<<"x==y";
      }
      return 0;
}
```

if 语句

```
#include<iostream>
using namespace std;
int main(){
      int x = 1; y = 2;
      if(x > y)
            cout<<x<<endl;
      cout<<y<<endl;
      return 0;
}
```

程序输出：2

如果省略了花括号，说明语句组只有一条语句，即cout<<x;而cout<<y;这条语句不属于语句组

if 语句

【例】如果输入字符为字母，则输出“YES”

```
#include<iostream>
using namespace std;
int main(){
      char ch;
      cin>>ch;
      if(ch>='a' && ch<='z'||ch>='A' && ch<='Z')
            cout<<"YES"<<endl;
      return 0;
}
```

if 语句

【例】输入一个年份，判断是否为闰年

- 输入年份
  - 可以限定范围，如0至9999年
- 判断是否为闰年的条件
  - 条件1
    - 能够被4整除
    - 不能够被100整除
  - 条件2
    - 能够被400整除
  - 上述两个条件满足其一即可

if 语句

```
#include<iostream>
using namespace std;
int main(){
      int year;
      cout<<"输入年份:"<<endl;
      cin>>year;
      if(year%4==0&&year%100!=0||year%400==0){
            cout<<year<<"年是闰年"<<endl;
      }
      else{
            cout<< year<<"年不是闰年"<<endl;
      }
      return 0;
}
```

if语句的嵌套

在一条if语句的某个分支（语句组）里，还可以再写if语句

```
if(n%3 == 0)
    if(n%5 == 0)
        cout<<n<<"是15的倍数"<<endl;
else //这个else和哪个if配对?
    cout<<n<<"不是3的倍数"<<endl;
```

n等于9时，输出n不是3的倍数，逻辑错误！

配对原则：else总是和其前面最近的尚未配对的if进行配对

if语句的嵌套

在一条if语句的某个分支（语句组）里，还可以再写if语句

```
if(n%3 == 0) {
    if(n%5 == 0)
        cout<<n<<"是15的倍数"<<endl;
} else //这个else和哪个if配对?
    cout<<n<<"不是3的倍数"<<endl;
```

解决方案：每一层的if都要加花括号，可以防止嵌套引起的逻辑错误！

if语句常见错误

赋值号误用为等号

```
int a = 0;
if(a = 0) { //a赋值为0，表达式为false
    cout<<"hello";
if(a = 5) //a赋值为5，表达式为true
    cout<<"Hi";
```

输出：Hi

if语句常见错误

在互相矛盾的多个条件，如果确实只希望执行一个分支，应该用if和多个else if，而不要写多个if

```
int a = 0;
if(a >= 0 && a < 5)
    a = 8;
else if(a >=5 && a < 10)
    cout<<"hello"
else
    .....
```

正确写法

不会输出hello

if语句常见错误

在互相矛盾的多个条件，如果确实只希望执行一个分支，应该用if和多个else if，而不要写多个if

```
int a = 0;
if(a >= 0 && a < 5)
    a = 8;
if(a >=5 && a < 10)
    cout<<"hello"
if(a >= 20)
    .....
```

错误写法

会输出hello

switch语句

```
switch (表达式) {   //表达式的值必须是整数类型(int, char ……)
        case 常量表达式1:  //常量表达式必须是整数类型的常量(int, char ……)
                语句组1;
                break;
        case 常量表达式2:
                语句组2;
                break;
        …
        case 常量表达式n:
                 语句组n
                 break;
        default:
                语句组n+1
}
```

常量表达式里不能包含变量！

“表达式”的值等于哪个“常量表达式”，就执行相应的语句组；都不相等，则执行default的语句组，default分支可以省略

switch语句

例子：输入1个整数，输出整数所对应的是星期几，如果整数不在1~7之内，输出错误

```
#include<iostream>
using namespace std;
int main(){
      int day;
      cin>>day;
      switch(day) {
         case 1:
             cout<<"Monday";
             break;
         case 2:
             cout<<"Tuesday";
             break;
         case 3:
             cout<<"Wednesday";
             break;
```

switch语句

```
            case 4:
                cout<<"Thursday";
                break;
            case 5:
                cout<<"Friday";
                break;
            case 6:
                cout<<"Saturday";
                break;
            case 7:
                cout<<"Sunday";
                break;
            default:
                cout<<"Illegal";
    }
    return 0;
}
```

switch语句

```
int x = 2;
switch(x) {
    case 1:
        cout<<"1";
        break;
    case 2:
    case 3:
        cout<<"2 or 3";
        break;
    default:
}
```

程序输出：2 or 3

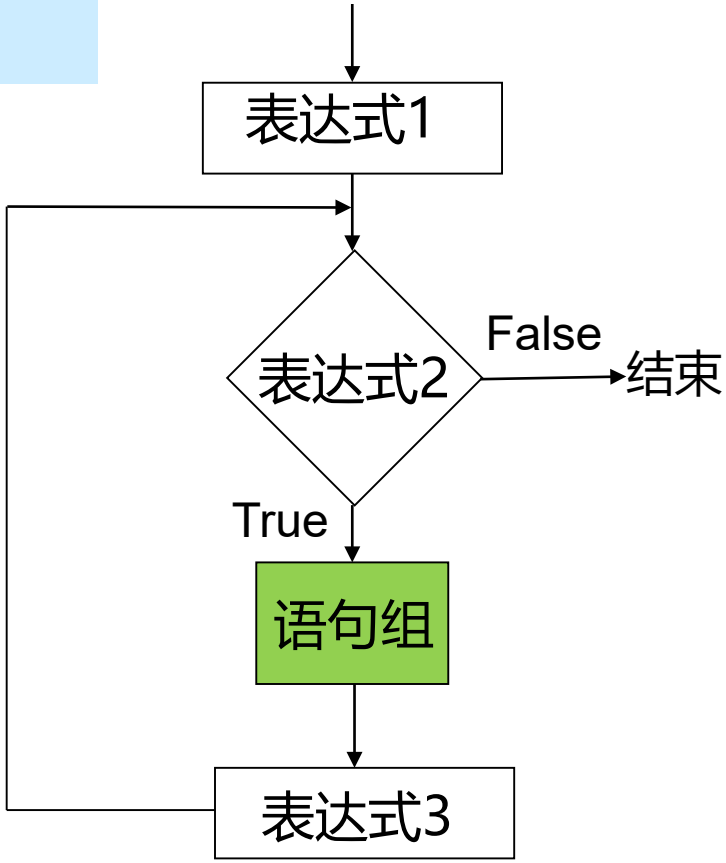
语句组中的break可以没有，执行语句组n时，如果没有break，将继续执行语句组n+1，直到碰到break为止或直到switch语句结束

循环语句

for语句

```
for (表达式1；表达式2; 表达式3) {
        语句组;
}
```

1) 计算“表达式1”
2) 计算“表达式2”，若其值为true, 则执行{}中的语句组，然后转到 3); 若为false，则不再执行{}中的语句组，转到 5)
3) 计算“表达式3”
4) 转到 2)
5) for循环结束



for语句

例子：连续输出26个字母

```
int i; //循环控制变量
for(i = 0; i < 26;  i++) {
     cout<<char('a' + i);//'a'+i 强制转换成char类型
}
```

循环控制变量也可以在表达式1里面定义

```
for(int i = 0; i < 26;  i++)
     cout<<char('a' + i);
```

//语句组里只有一条语句可以不写{}

for语句

循环控制变量定义for循环内部，则其只在for语句内部起作用

```
int i = 5; //外部定义的循环控制变量
for(int i = 0; i < 26;  i++) {
     cout<<char('a' + i);
}
for(int i = 0; i < 26;  i+=2){
     cout<<char('A' + i);
}
cout<<endl;
cout<<i; //此处的 i 和for循环里面的 i 无关
```

for循环内部定义的变量，只在for循环内部起作用，与外部无关

程序输出：
abcdefghijklmnopqrstuvwxyz
ACEGIKMOQSUWY
5

for语句

for循环结构里的“表达式1”和“表达式3”都可以是用逗号连接的若干表达式

```
for(int i=15, j=0; i > j; i-=2, j+=3)
     cout<<i<<" , "<<j<<endl;
}
```

程序输出：
15, 0
13, 3
11, 6

for语句

例子：写一个程序，输入一个正整数n，从小到大输出它的所有因子

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin>>n;
    for(int i = 1; i <=n; i ++)
        if(n % i == 0)
            cout<<i<<endl;
    return 0;
}
```

输入：
15
输出：
1
3
5
15

for语句

for循环可以嵌套，形成多重循环：

```
for(int i = 0; i < n; i ++) {
    ......
    for(int j = 0; j < m; j ++) {
        ...... //内重循环一共执行n*m次
    }
}
```

for语句

例子：给定正整数n和m，在1至n这n个数中，取出两个不同的数，使得其和是m的因子，问有多少种不同的取法。

思路：穷举

第一个数取1，第二个数分别取2, 3, ...n
第一个数取2，第二个数分别取3, 4, ...n
...
第一个数取n-2，第二个数分别取n-1, n
第一个数取n-1，第二个数取n

for语句

```
#include <iostream>
using namespace std;
int main() {
    int n, m;
    int total = 0; //取法总数
    cin>>n>>m;                                    //取第一个数，共
    for(int i = 1; i < n; i ++) {                 n-1种取法
        for(int j = i+1; j <= n; j ++) {
            if(m % (i+j) == 0)                    //第二个数要比第一个
                total ++;                         数大，以免取法重复
    }
    cout<<total;
    return 0;
}
```

for语句

for语句括号里面的“表达式1”，“表达式2”，“表达式3”任何一个都可以不写，但“;”必须保留

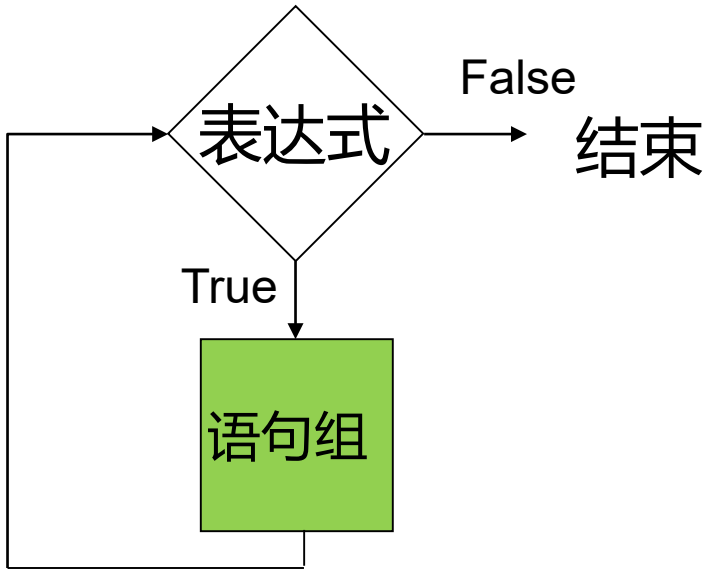
```
for( ; i < 100; i ++) {//假设i在for前已经有合理值
    cout<<i;


for( ; ; ) {
    cout<<“hello”<<endl; //永远不停输出hello
```

while循环

```
while (表达式) {
        语句组;
}
```

1) 判断“表达式”是否为true，如果不为true，转4)
2) 执行”语句组”
3) 转1)
4) while循环结束



while循环

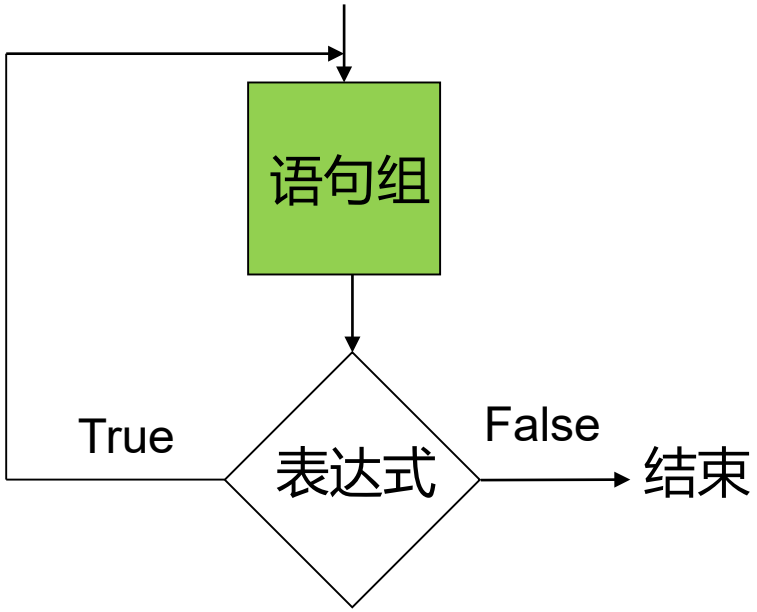
例子：求1+2+3+…+100的值

```
#include <iostream>
using namespace std;
const int n = 100;
int main() {
    int i = 1, sum=0;
    while(i <= n){
        sum += i;
        i ++;
    }
    cout<<"sum="<<sum<<endl;
    return 0;
}
```

do...while循环

```
do {
        语句组
} while (表达式) ;
```

1) 执行”语句组”
2) 判断“表达式”是否为true，如果不为true，转4)
3) 转1)
4) 循环结束



do...while循环

例子：输出1到1000以内所有的2的整数次幂

```
#include <iostream>
using namespace std;

int main() {
    int n = 1;
    do {
        cout<<n<<endl;
        n *= 2;
    } while (n < 10000);
    return 0;
}
```

三类循环语句之间的等价变换

```
int i=1,sum=0;
//循环初始条件
while(i<=4){
      sum+=i;
      i++;   //修改循环条件
}
```

```
int i=1,sum=0;
//循环初始条件
do{
    sum+=i;
    i++;//修改循环条件
 } while(i<=4);
```

```
int i,sum=0;
for( i=1; i<=4; i++ ){
   sum+=i;
}
/*习惯上：表达式1：循环初始条件；表达式2：循环
   终止条件；表达式3：修改循环条件*/
```

转向语句

break语句

break语句出现在循环体(for、while、do...while)中，作用是跳出循环

```
int n = 0;
while(true) {
    if(n > 100)
        break;
    n++;
}
cout<<n;
```

当 if 条件满足时，执行 break，直接跳出while循环

break语句

break语句出现在循环体(for、while、do...while)中，作用是跳出循环

```
for(int i = 0; i < 2; i ++) {
    for(int x = 0; x < 10; x ++) {
        if(x > 8) { //内层循环每次循环到x>8时跳出
            break;  ，但外层循环不受影响
        }
        cout<<x<<endl;
    }
}
```

如果有多重循环，break只跳出所在的那一层循环

continue语句

continue语句出现在循环体(for、while、do...while)中，作用是跳过一次循环

```
int x = 0;
while (x < 10) {
    if(x == 8) {
        continue;
    }
    cout<<x<<endl;
    x ++;
}
```

当 x == 8 时，立即结束本次循环 (continue 之后的语句)，回到循环开头判断是否进行下一次循环

continue语句

continue语句出现在循环体(for、while、do...while)中，作用是跳过一次循环

```
for(int i = 0; i < 3; i ++) {
    cout<<i<<endl;
    if(i == 1) {
        continue;
    }
    for(int j = 0; j < 2; j ++) {
        if(j == 0) {
            continue;
        }
        cout<<j<<endl;
    }
}
```

如果有多重循环，continue只对包含它的那重循环起作用

控制语句练习

例1.乘方计算

给出一个整数a和一个正整数n，求乘方a^n。

输入：
一行，包含两个整数a和n。-1000000 <= a <= 1000000，1<=n<=10000。
输出：
一个整数，即乘方结果。

样例输入
2 3
样例输出
8

例1.乘方计算

给出一个整数a和一个正整数n，求乘方a^n。

```
#include <iostream>
using namespace std;
int main() {
    int a, n;
    cin>>a>>n;
    int result = a;
    for(int i = 0; i <n-1; i ++)
        result *= a;
    cout<<result;
    return 0;
}
```

例2.输入若干个整数求最大值

输入若干个整数（可正可负，不超过int的表示范围），输出最大值

样例输入

-100 -20 20 -2

样例输出

20

例2.输入若干个整数求最大值

```
#include <iostream>
using namespace std;
int main() {
    int a, mx;
    bool first = true; //输入的是否是第1个数
    while(cin >> n)
        if(first) {
            mx = n;
            first = false;
        } else {
            if(n > mx)
                mx = n;
        }
    }
    return 0;
}
```

例3.求阶乘的和

给定正整数n，求不大于n的正整数的阶乘的和（即求1! + 2! + 3! + … + n!）

输入
输入有一行，包含一个正整数n（1<n<12）

输出
阶乘的和

样例输入
5
样例输出
153

例3.求阶乘的和

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin>>n;
    int sum = 0;
    int factorial = 1;
    for(int i = 1; i <= n; i ++) {
        factorial *= i;
        sum += factorial;
    }
    cout<<sum;
    return 0;
}
```

例4.求小于整数n的全部质数

输入正整数n(n>=2)，求不大于n的全部质数

解法1

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin>>n;
    for(int i = 2; i <= n; i ++) { //判断i是否是质数
        int k;
        for(k = 2; k < i; k ++) {
            if(i % k == 0)
                break;
        }
        if(k == i)
            cout<<i<<endl;
    }
    return 0;
}
```

//k == i说明没有执行过break

此解法做了没必要的尝试，k大于i的平方根后就不必再试

例4.求小于整数n的全部质数

输入正整数n(n>=2)，求不大于n的全部质数

解法2

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin>>n;
    cout<<2<<endl;
    for(int i = 3; i <= n; i += 2;) { //判断i是否是质数
        int k;
        for(k = 3; k < i; k += 2) {
            if(i % k == 0 || k*k > i)
                break;
        }
        if(k*k > i)
            cout<<i<<endl;
     }
     return 0;
}
```

END