第二章 C++简单程序设计（2）

清华大学　郑 莉

教材：C++语言程序设计（第4版） 郑莉　清华大学出版社
答疑：每周一17:30—19:00，东主楼8区310

问题与解决

- 程序的执行流程不总是顺序的，因此
  - 程序要能够对执行流程进行选择（选择语句、开关语句）
  - 程序要能够反复用同一算法依次处理大批量数据（循环语句）
- 基本数据类型能够表示的数据种类很有限，因此
  - 需要能够在程序中自定义类型（第4章讲类）
  - 枚举类型就是通过列出所有可取值，来定义一种新类型

目录

例2_2：输入一个年份，判断是否闰年

```
//2_2.cpp
#include <iostream>
using namespace std;
int main() {
    int year;
    bool isLeapYear;
    cout << "Enter the year: ";
    cin >> year;
    isLeapYear = ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0));
    if (isLeapYear)
        cout << year << " is a leap year" << endl;
    else
        cout << year << " is not a leap year" << endl;
    return 0;
}
```

If语句的语法形式

if (表达式) 语句

例：if (x > y) cout << x;

if (表达式) 语句1 else 语句2

例：if (x > y) cout << x;

else cout << y;

if (表达式1) 语句1

else if (表达式2) 语句2

else if (表达式3) 语句3

…

else 语句 n

例2_3：输入两个整数，比较两个数的大小。

```
//2_3.cpp
#include<iostream>
using namespace std;
int main() {
   int x, y;
   cout << "Enter x and y:";
   cin >> x >> y;
   if (x != y)
      if (x > y)
         cout << "x > y" << endl;
      else
         cout << "x < y" << endl;
   else
      cout << "x = y" << endl;
   return 0;
}
```

运行结果1：
Enter x and y:5 8
x < y
运行结果2：
Enter x and y:8 8
x = y
运行结果3：
Enter x and y:12 8
x > y

补充2_11：计算两整数和，判断第1个数能否被第2个数整除

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int ival1, ival2;
    cin>> ival1>> ival2;
    if (ival2>=0)
        cout<<ival1<<"+"<<ival2<<"="<<ival1+ival2<<endl;
    else
        cout <<ival1<<"+("<<ival2<<")"<<"="<<ival1+ival2<<endl;
    if (ival1%ival2==0)
    {
        if(ival1!=0)
            cout<<ival1<<"%"<<ival2<<"==0"<<endl;
    }
    else
       cout<<ival1<<"%"<<ival2<<"!=0"<<endl;
    return 0;
}
```

嵌套的if结构（续）

- 语法形式

```
if(  )
    if(  ) 语句 1
    else 语句 2
else
    if(  ) 语句 3
    else 语句 4
```

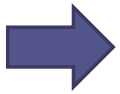
- 注意

  语句 1、2、3、4 可以是复合语句，每层的 if 与 else 配对，或用 { } 来确定层次关系。

补充2_12：将百分制转化为五分制

```
#include <iostream>
using namespace std;
int main(){
    int oldScore,newScore;
    cout << "请输入百分制的成绩:";
    cin >> oldScore;
    newScore = 0;
    if(oldScore > 80 )
        newScore = 5;
    else if( oldScore >60 )
        newScore = 4;
    else if( oldScore > 40 )
        newScore = 3;
    else if( oldScore > 20 )
        newScore = 2;
    else if( oldScore > 0 )
        newScore = 1;
    else
        cout <<"请输入0~100范围内的成绩！"<<endl;
    cout << "转化为五分制分数为：" << newScore << endl;
    return 0;
}
```

```
if (表达式1) 语句1
else
    if (表达式2) 语句2
    else
        if (表达式3) 语句3
        …
            else 语句 n
```



```
if (表达式1) 语句1
else if (表达式2) 语句2
else if (表达式3) 语句3
        …
else 语句 n
```

例2_4：输入一个0～6的整数，转换成星期输出

```
//2_4.cpp
#include <iostream>
using namespace std;
int main() {
    int day;
    cin >> day;
    switch (day) {
    case 0: cout << "Sunday" << endl; break;
    case 1: cout << "Monday" << endl; break;
    case 2: cout << "Tuesday" << endl; break;
    case 3: cout << "Wednesday" << endl; break;
    case 4: cout << "Thursday" << endl; break;
    case 5: cout << "Friday" << endl; break;
    case 6: cout << "Saturday" << endl; break;
    default:
       cout<<"Day out of range Sunday .. Saturday"<< endl;   break;
    }
    return 0;
}
```

switch语句的语法

- 一般形式

```
switch (表达式)
  { case   常量表达式 1：语句1
    case   常量表达式 2：语句2
             ┆
    case   常量表达式 n：语句n
    default :          语句n+1
  }
```

- case分支可包含多个语句，且不用{ }。
- 表达式、判断值都是int型或char型。
- 若干分支执行内容相同可共用一组语句。

- 执行顺序

以case中的常量表达式值为入口标号，由此开始顺序执行。因此，每个case分支最后应该加break语句。

例2_5：求自然数1～10之和

```
//2_5.cpp
#include <iostream>
using namespace std;
int main() {
  int i = 1, sum = 0;
  while (i <= 10) {
      sum += i;  //相当于sum = sum + i;
      i++;
  }
  cout << "sum = " << sum << endl;
   return 0;
}
```

分析：本题需要用累加算法，累加过程是一个循环过程，可以用while语句实现。

运行结果：
sum = 55

while语句的语法

- 形式

  while（表达式）语句

  可以是复合语句，其中必须含有改变条件表达式值的语句。

- 执行顺序

  先判断表达式的值，若为 true 时，执行语句。

例2_6：输入一个整数，将各位数字翻转后输出

```
//2_6.cpp
#include <iostream>
using namespace std;
int main() {
   int n, right_digit;
   cout << "Enter the number: ";
   cin >> n;
   cout << "The number in reverse order is  ";
   do {
         right_digit = n % 10;
         cout << right_digit;
         n /= 10;  //相当于n=n/10
   } while (n != 0);
    cout << endl;
   return 0;
}
```

运行结果：

```
Enter the number: 365
The number in reverse order is  563
```

do-while 语句的语法形式

- 一般形式

  do　语句

  while (表达式)

  可以是复合语句，其中必须含有改变条件表达式值的语句。

- 执行顺序

  先执行循环体语句，后判断条件。
  表达式为 true 时，继续执行循环体

- 与while语句的比较：

  while 语句执行顺序
  先判断表达式的值，为true时，再执行语句

例2_7：用do-while语句，求自然数1~10之和

```
//2_7.cpp
#include <iostream>
using namespace std;
int main() {
   int i = 1, sum = 0;
   do {
         sum += i;
         i++;
   } while (i <= 10);
   cout << "sum = " << sum << endl;
   return 0;
}
```

对比下面的程序

程序1：

```
#include <iostream>
using namespace std;
int main() {
  int i, sum = 0;
  cin >> i;
  while (i <= 10) {
    sum += i;
    i++;
  }
  cout<< "sum= " << sum
      << endl;
  return 0;
}
```

程序2：

```
#include <iostream>
using namespace std;
int main() {
  int i, sum = 0;
  cin >> i;
  do {
    sum += i;
    i++;
  } while (i <= 10) ;
  cout << "sum=" << sum
       << endl;
  return 0;
}
```

例2_8：输入一个整数，求出它的所有因子

```
//2_8.cpp
#include <iostream>
using namespace std;
int main() {
   int n;
   cout << "Enter a positive integer: ";
   cin >> n;
   cout << "Number  " << n << "   Factors  ";
   for (int k = 1; k <= n; k++)  //思考
     if (n % k == 0)
        cout << k << "   ";
  cout << endl;
  return 0;
}
```

运行结果1：

```
Enter a positive integer: 36
Number  36  Factors  1  2  3  4  6  9  12  18  36
```

运行结果2：

```
Enter a positive integer: 7
Number  7   Factors  1  7
```

for语句（续）

语法形式

for（初始语句；表达式1；表达式2）语句

for语句还有另一种更加简洁的写法，称为范围for语句，语法形式为：

for（声明：表达式）

语句

这种形式的for语句主要用于遍历一个容器中的序列，将在第6、10章详细介绍

循环结构与选择结构的嵌套

例2_10：输入一系列整数，统计出正整数个数i和负整数个数j,读入0则结束。

- 分析：
  - 需要读入一系列整数，但是整数个数不定，要在每次读入之后进行判断，因此使用while循环最为合适。循环控制条件应该是n!=0。由于要判断数的正负并分别进行统计，所以需要在循环内部嵌入选择结构。

例2-10

```
//2_10.cpp
#include <iostream>
using namespace std;

int main() {
   int i = 0, j = 0, n;
   cout <<"Enter some integers please (enter 0 to quit):" << endl;
   cin >> n;
   while (n != 0) {
      if (n > 0) i += 1;  //
      if (n < 0) j += 1;  //可以如何修改？
      cin >> n;
   }
   cout << "Count of positive integers: " << i << endl;
   cout << "Count of negative integers: " << j << endl;
   return 0;
}
```

其他控制语句

- break语句
  使程序从循环体和switch语句内跳出，继续执行逻辑上的下一条语句。不宜用在别处。
- continue 语句
  结束本次循环，接着判断是否执行下一次循环。
- goto 语句
  goto语句的作用是使程序的执行流程跳转到语句标号所指定的语句。

枚举类型

- 只要将需要的变量值一一列举出来，便构成了一个枚举类型。
- C++包含两种枚举类型：不限定作用域的枚举类型和限定作用域的枚举类。
- 不限定作用域枚举类型声明形式如下：

  enum 枚举类型名 {变量值列表};
  - 例如：

    enum Weekday
    {SUN, MON, TUE, WED, THU, FRI, SAT};
- 关于限定作用域的enum类将在第4章和第5章详细介绍。

不限定作用域枚举类型说明

- 对枚举元素按常量处理，不能对它们赋值。例如，不能写：SUN = 0;
- 枚举元素具有默认值，它们依次为： 0,1,2,……。
- 也可以在声明时另行指定枚举元素的值，如：

  enum Weekday{SUN=7,MON=1,TUE,WED, THU,FRI,SAT};
- 枚举值可以进行关系运算。
- 整数值不能直接赋给枚举变量，如需要将整数赋值给枚举变量，应进行强制类型转换。

例2-11

- 设某次体育比赛的结果有四种可能：胜（WIN）、负（LOSE）、平局（TIE）、比赛取消（CANCEL），编写程序顺序输出这四种情况。
  - 分析：由于比赛结果只有四种可能，所以可以声明一个枚举类型，声明一个枚举类型的变量来存放比赛结果。

例2-11

```
#include <iostream>
using namespace std;
enum GameResult {WIN, LOSE, TIE, CANCEL};
int main() {
    GameResult result;
    enum GameResult omit = CANCEL;
    for (int count = WIN; count <= CANCEL; count++) {
      result = GameResult(count);
      if (result == omit)
        cout << "The game was cancelled" << endl;
      else {
        cout << "The game was played ";
        if (result == WIN)      cout << "and we won!";
        if (result == LOSE)       cout << "and we lost.";
        cout << endl;
      }
    }
    return 0;
}
```

运行结果

```
The game was played and we won!
The game was played and we lost.
The game was played
The game was cancelled
```

小结

- 主要内容
  - 程序流程的选择和循环结构if、while、do...while、for语句适用的场合、算法，几种流程控制语句的语法。
  - 枚举类型。
  - 综合案例。
- 达到的目标
  - 能够设计简单的选择结构、循环结构算法，能够运用流程控制语句实现简单算法。简单了解枚举类型。