作业程序测试大多不充分

- 有了分支结构（尤其今天讲了循环后）
  - 所有路径、边界条件都要覆盖
  - 提交的结果截图要体现这一点
- 如数字逆序
  - 输入abc对不对？
  - 输入230能返回032吗？
- 如时间制转换
  - 输入3:03行不行？
  - 输入15：00行不行？
  - 输入12：00行不行？
- 如纸牌
  - 对规则的理解
  - 如果有多个A怎么办？
- 一些共性问题（作业示例）
  - **还有人在提交doc（除了orz，我别无他选。。。）**
  - **个别同学截图不完整（似乎想隐藏什么。。。）**
  - **个别同学的截图跟上次作业不一样（窗口风格、目录等，重装系统？？？）**

关于教材与参考书的错误预警

- 书出错是正常的
  - 尤其是翻译的书
    - 有可能是高考还不如大家的人翻译的
  - 不正常的是我们发现了还强迫自己相信
  - 如教科书：129页（中英文均有）
- 发现错误了怎么办
  - 如果肯定是错误，并能证明，就嘲笑作者
  - 如果不肯定，怀疑，就相互讨论，一起嘲笑
  - 如果讨论了还不确定，告诉我，我也加入嘲笑
  - 总结：因为年青，你（wo）们对的概率很高

School of Economics ang Management, Beihang University

Introduction to C Programming

Jichang Zhao

jichang@buaa.edu.cn

Repetition

Objectives

- Basic Loop Structures
- The `while` Statement
- The `for` Statement
- Nested Loops
- The `do-while` Statement
- 阅读5.8编程错误与编译器错误
  - p205

Introduction

- A section of code that is repeated is called a **loop**, because after the last statement in the code is executed, the program branches, or loops, back to the first statement and starts another repetition through the code
- Each repetition is also called an **iteration** or **pass through the loop**

Basic Loop Structures

- Constructing a repeating section of code requires that four elements be present
  - **Repetition statement**
    - `while` statement
    - `for` statement
    - `do-while` statement
  - **Condition**
  - **A statement that initially sets the condition being tested**
  - **A statement within the repeating section of code that alters the condition so that it eventually becomes false**

Pretest and Posttest Loops

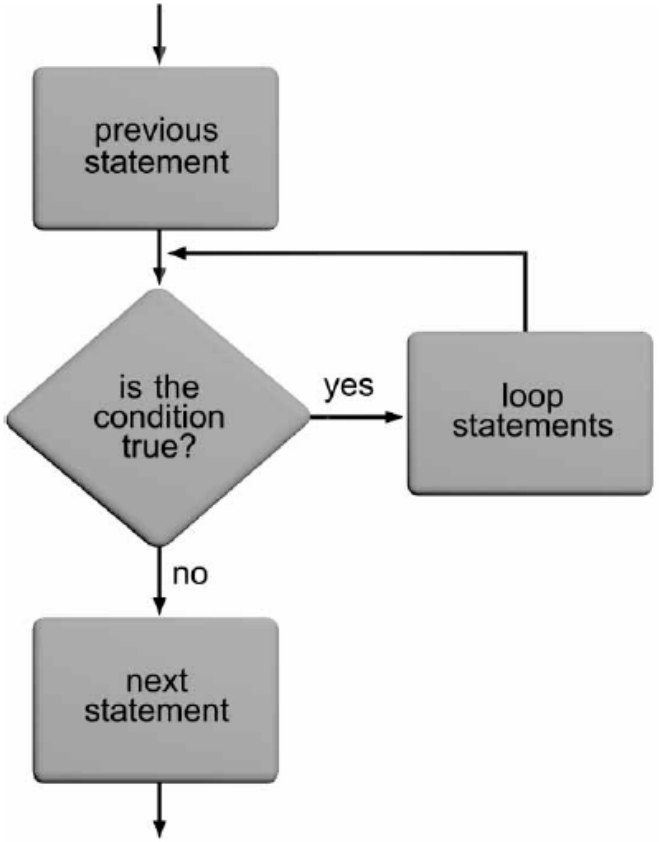

Figure 5.1 A pretest (entrance-controlled) loop

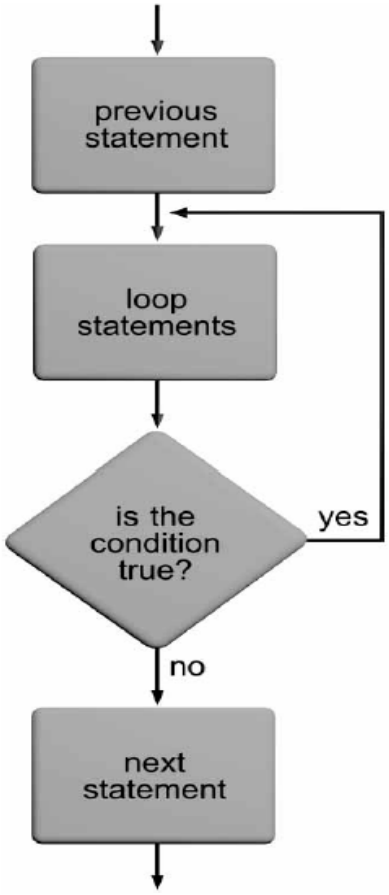

Figure 5.2 A posttest (exit-controlled) loop

Counter-Controlled and Condition-Controlled Loops

- **Counter-controlled loop**
  - the condition is used **to keep track of the number of repetitions**
  - Also known as a **fixed-count loop**
- **Condition-controlled loop**
  - the tested condition does not depend on a count being achieved, **but rather on a specific value being encountered**

The `while` Statement

- The general form of the `while` statement is

```
while (expression)
   statement;
```

- entry condition
- The transfer of control back to the start of a `while` statement to reevaluate the expression is known as a **program loop**
- *The following is a valid but infinite loop:*

```
while (count <= 10)
   printf("%d ",count);
```

condition altering is necessary

The break and continue Statements

- A break forces an ***immediate exit*** from while, switch, for, and do-while statements only

```
while(count <= 10)
{
  printf("Enter a number: ");
  scanf("%f", &num);
  if (num > 76)
  {
    printf("You lose!");
    break; /* break out of the loop */
  }
  else
    printf("Keep on truckin!");
}
/* break jumps to here */
```

提前发现已经达到目的，无需继续循环

The break and continue Statements

- The continue applies to loops **only**
- when a continue statement is encountered in a loop, ***the next iteration of*** the loop **begins immediately**

```
while (count < 30)
{
  printf("Enter a grade: ");
  scanf("%f", &grade);
  if(grade < 0 || grade > 100)
    continue;
  total = total + grade;
  count = count + 1;
}
```

提前发现不满足条件，需要中止并重新开始

The Null Statement

- A semicolon with nothing preceding it is also a valid statement, called the **null statement**
  - **我们在`if`语句里已经提及**
- Use the null statement where a statement is syntactically required, but no action is needed
- Null statements typically are used either with `while` or `for` statements
  - 本质上条件语句里有某种“运算”或“修改”

The `for` Statement

- The `for` statement combines all four elements required to easily produce a loop **on the same line**

```
for (initializing list; tested expression; altering list)
   statement;
```

- This statement does not require that any of the items in parentheses be present or that they actually be used **for initializing or altering the values** in the expression statements
  - However, the two semicolons must be present
    - **`for ( ; count <= 20;)` is valid**
    - **`for ( ; ; )` is also valid**
    - Omitting tested expression results in infinite loop (if there is no break)
    - **`for ( int i=0; i<100; i++)`** **`//c99标准，部分编译器默认不支持，且该变量仅能在for循环中使用`**
    - **`for (i=0,j=0,k=0; i<n && j<n && k<n; i++,j+=1,k++)`**
    - **`//可以这么写，但不是好的风格`**

The `for` Statement

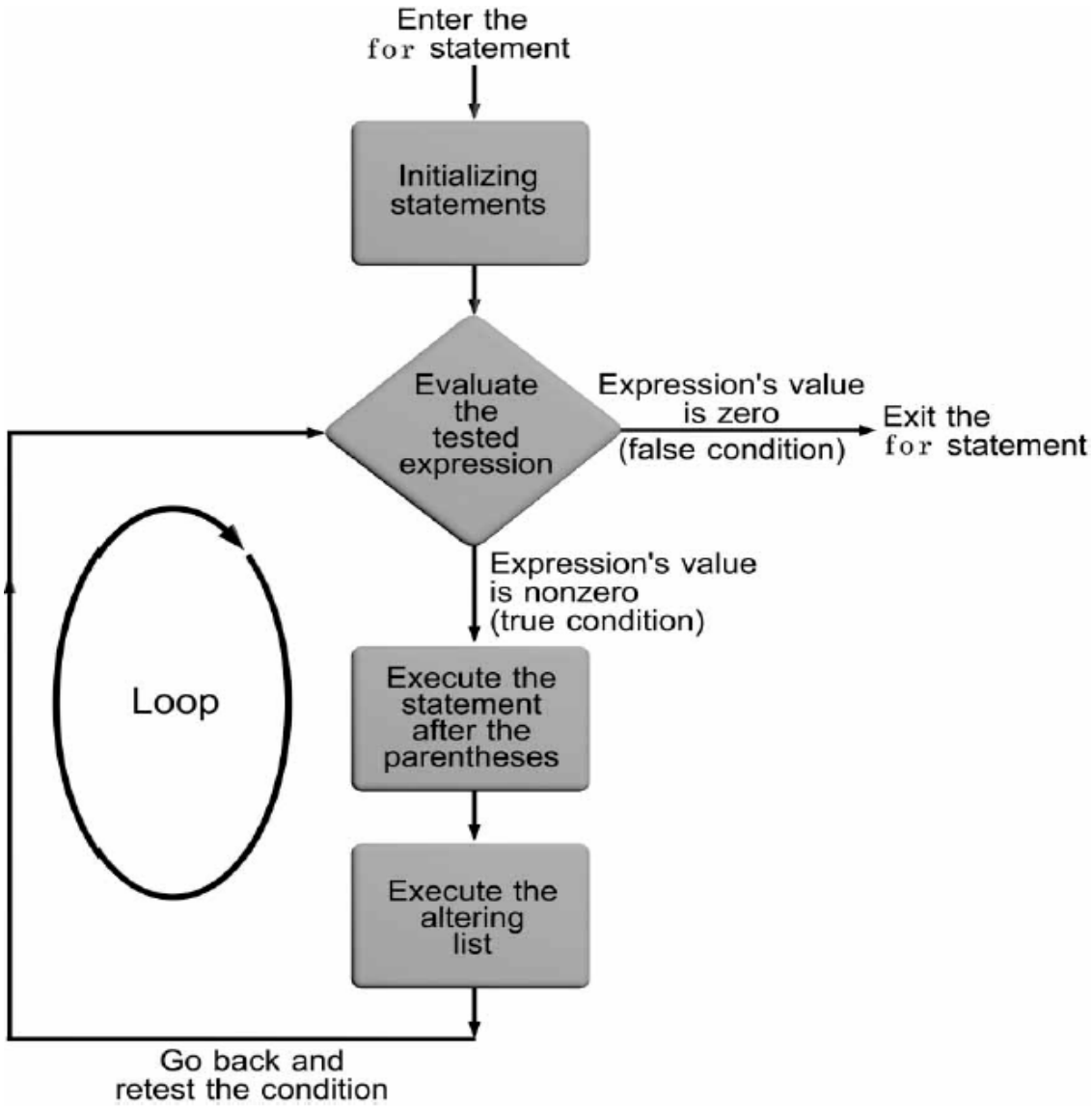

Figure 5.8 `for` statement flow of control

补充：计数的形式

- 从0到n-1
  - `for(i=0; i<n; i++)`
  - 计算机语言里一般序号从0开始（数组中）
- 从1到n
  - `for(i=1; i<=n; i++)`
- 从n-1到0
  - `for(i=n-1; i>=0; i--)`
- 从n到1
  - `for(i=n; i>0; i--)`

Nested loops

- `int i;`
- `int j;`
- `for(i=1;i<=100;i++)`
  - `for(j=1;j<=4;j++)`
  - `{`
    - ….
  - `}`
- `complexity!`

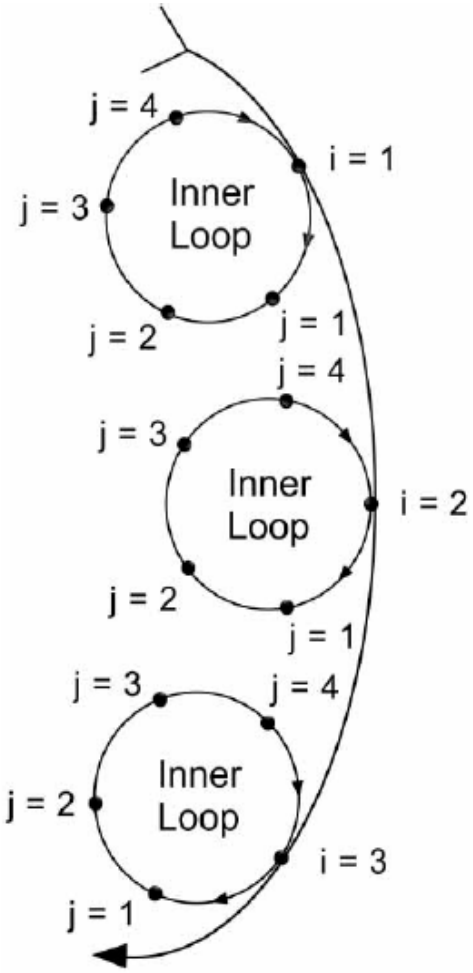

Figure 5.9 `j` loops once for each `i`

The `do-while` statement

- `while` statement might require some necessary statements (but duplicated) before the loop
  - e.g. prompt and scanf both before the inside the loop
- The `do-while` statement allows us to do some statements before an expression is evaluated
  - It can be used to eliminate the duplication
- `do`
  - `statement;`
- `while (expression);`
- exit condition

The do-while statement

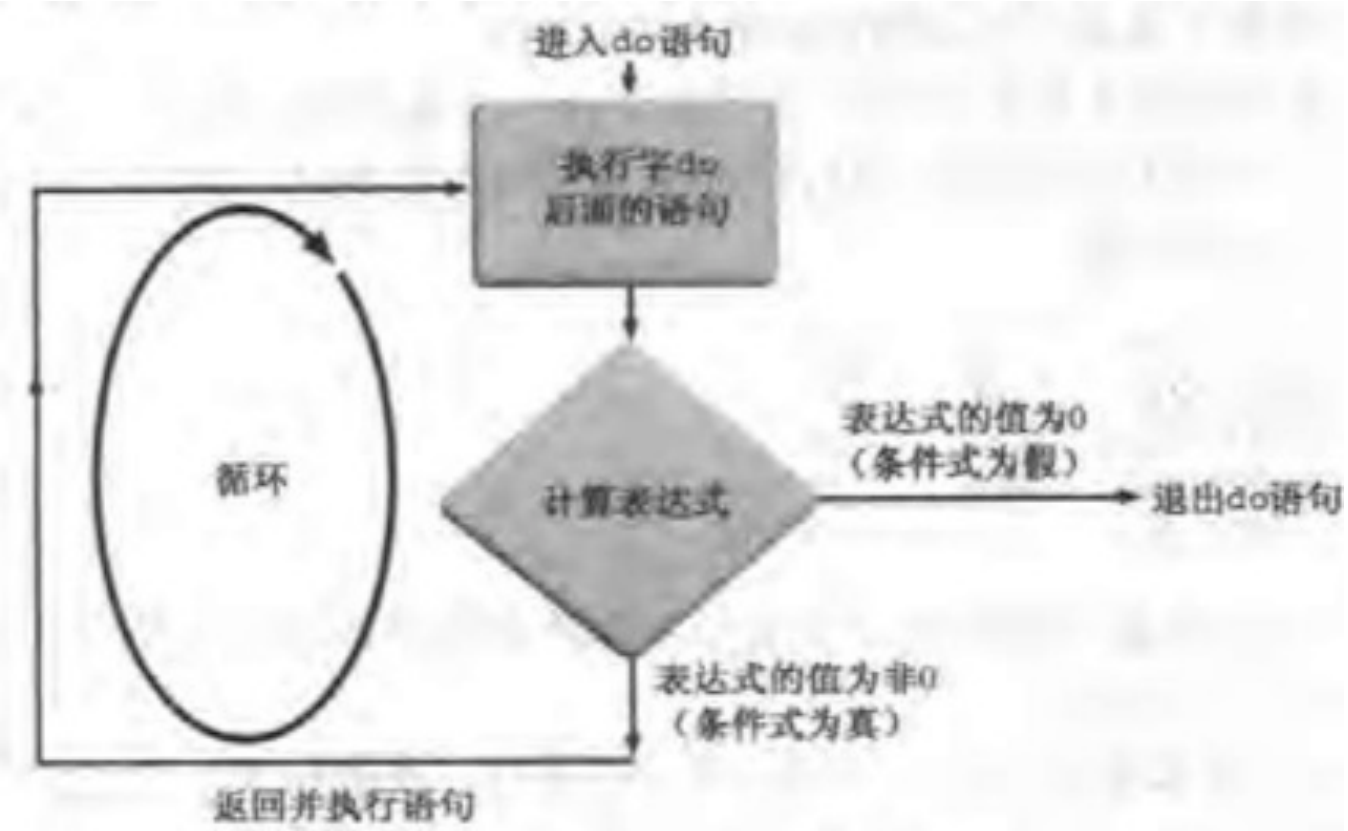

图 5.10　do-while语句流程控制

补充：符号常量的使用

- 在条件中使用符号常量
  - 程序的可读性更好
  - 程序的可维护性更好
  - 好的编程风格
- `#define STUDENT_NUMBER 198`
- …

- `int i;`
- `for(i=0;i<STUDENT_NUMBER;i++)`
- `{`
  - `scanf…`
- `}`

Homework

- 会给大家两周时间，充分测试，截图完整，要体现测试
- 1. P183, 2
- 2. P191, 7
- 3. P198, 11
- 4. P202, 5
- 5. P205, 3
- 6. 编写程序，找出用户输入一串数中的最大数和最小数。程序需要提示用户一个一个地输入数，当用户输入0或负数时，程序停止输入，并显示已输入的最大非负数和最小非负数。注意：输入的数不一定是整数。
- 7. 编写程序，要求用户输入一个分数，然后将其约为最简形式。如输入6/12，输出1/2；输入12/6，输出2。提示：先计算出分子、分母的最大公约数。

Case study

- 写一个按秒的倒计时。

Case study

- 计算一个输入整数的位数
  - while
  - do while

Case study

- f(n)= (1+1/n)^n，输出不同n下，f(n)的值，观察是否收敛

Case study

- 用c语言求函数f(x)=-xlnx在何处取大值
  - x : (0,1)
- 解析解与数值解
  - Analytical
  - Numerical

Case Study

- How many prime numbers in the range[2,n], where n should be read through scanf.

Case study

- The Multiplication Table

```
1×1=1
2×1=2 2×2=4
3×1=3 3×2=6 3×3=9
4×1=4 4×2=8 4×3=12 4×4=16
5×1=5 5×2=10 5×3=15 5×4=20 5×5=25
6×1=6 6×2=12 6×3=18 6×4=24 6×5=30 6×6=36
7×1=7 7×2=14 7×3=21 7×4=28 7×5=35 7×6=42 7×7=49
8×1=8 8×2=16 8×3=24 8×4=32 8×5=40 8×6=48 8×7=56 8×8=64
9×1=9 9×2=18 9×3=27 9×4=36 9×5=45 9×6=54 9×7=63 9×8=72 9×9=81
```

Case study

写程序，当输入某个大写字母时，逐个降序并换行打印，直至A

如输入F时，打印如下：

F
FE
FED
FEDC
FEDCB
FEDCBA

Case study

- 写程序，输入某大写字母，打印出如下字母塔
- 如输入E，输出如下：
- A
- ABA
- ABCBA
- ABCDCBA
- ABCDEDCBA

- 思考：把塔倒过来怎么打印？