第3章 流程控制

3.1 本章语法

1. 流程控制

流程控制是计算机运算领域的用语，意指在程序运行时，个别的指令(或是陈述、子程序)运行或求值的顺序。流程控制分为三种：

1) 顺序结构

顺序结构是指语句从上到下顺序执行。

2) 选择结构

选择结构是依指定变量或表达式的结果，决定后续运行的程序，选择结构通过 if 语句，分 3 种形式：

- 单分支：如果条件正确就执行一个单向 if 语句。当且仅当条件为 true 时，一个单向 if 语句执行一个动作。注意：if 块中的语句都要在 if 语句之后缩进。格式为：

```
if 判断条件:
    执行语句块
```

- 双分支：双向 if-else 语句根据条件是真还是假来决定要执行哪一个动作。如果条件是 True，那么 if 语句执行第一个动作，但当条件是 False 使用双向 if-else 来执行第二个动作。格式为：

```
if 判断条件:
    执行语句块 1
else:
    执行语句块 2
```

- 多分支：Python 不支持 switch 语句，常用的多分支是 if … elif … else 判断语句。格式为：

```
if 判断条件 1:
    执行语句块 1
elif 判断条件 2:
    执行语句块 2
```

```
elif 判断条件 3:
    执行语句块 3
else:
    执行语句块 4
```

3）循环结构

循环结构是指一段在程序中只出现一次，但可能会连续运行多次的代码。循环分 2 种形式：

- for 循环：该循环可指定运行次数。格式为：

```
for 循环变量 in 迭代器:
    执行语句块
```

迭代器有很多类型，例如：range 函数、字符串、列表、元组、字典、集合等。

- while 循环：该循环可指定继续运行条件（或停止条件），比较重视对循环条件的判断语句进行执行循环的动作。格式为：

```
while 判断条件:
    执行语句块
```

绝大多数场景下，for 循环与 while 循环可以互相替代。

2. 条件表达式

选择结构和循环结构中，都要根据条件表达式的值来确定下一步的执行流程。在条件表达式中会经常用到关系运算符（==、!=、<、>、>=、<=）和逻辑运算符（and、or、no）。

使用各种运算符可以构建不同的条件表达式，例如：假设有整数 x，如果满足 x % 2 == 0，则表示 x 为一个偶数；满足 x % 3 == 0 and x % 10 == 5，则表示 x 是 3 的倍数且个位上数字为 5。假设三角形的 3 条边分别为 a、b、c，如果满足（a+b>c）and（b+c>a）and（a+c>b）and（a>0）and（b>0）and（c>0），则表示 a、b、c 能构成一个三角形。

3. range 函数

range 函数是 Python 中的一个内置函数，调用这个函数就能产生一个迭代序列，因此适合放在 for 语句的头部，range 函数有 3 种不同的调用方式：

- range(n)：得到的迭代序列为：0,1,2,3,…,n−1。例如：range(100)表示序列 0,1,2,3,…,99。当 n 小于等于 0 的时候序列为空。
- range(m, n)：得到的迭代序列为：m,m + 1, m + 2,…,n−1。例如：range(11,16)表示序列 11,12,13,14,15。当 m>=n 的时候序列为空。
- range(m, n, d)：得到的迭代序列为：m,m +d, m + 2d,…，按步长值 d 递增，如果 d 为负则递减，直至那个最接近但不包括 n 的等差值。因此 range(11,16,2)表示序列 11,13,15；而 range(15,4,−3)表示的序列为：15,12,9,6。

4. break 和 continue 语句

for 语句和 while 语句都是通过头部控制循环的执行，一旦进入循环体，就会完整地执行一遍其中的语句，然后再重复。实际中，也会遇到一些只执行循环体中的部分语句就结束循环或者立刻转去做下一次循环的情况，那么就需要用到循环控制语句 break 和 continue。

break 语句的作用是立刻结束整个 for 循环，continue 语句的作用是结束这一轮的循环，程序跳转到循环头部，根据头部的要求继续。

循环中的 else 用法：碰到 break、return，打破整个循环，不执行 else；碰到 continue，只是跳出单次循环，整个循环完毕还是会执行 else。

5. 嵌套

Python 语言允许选择嵌套和循环嵌套以及选择和循环彼此嵌套。

循环嵌套是指在一个循环体里面嵌入另一个循环，将嵌套在里面的循环当作一个语句来看，相对外面的循环，每次执行完整的内存循环，常用的有两种形式：

- for 循环嵌套 for 循环，格式为：

```
for 循环变量 1 in 迭代器 1:
    for 循环变量 2 in 迭代器 2:
        执行语句块 1
    执行语句块 2
```

- while 里嵌套 while 循环，格式为：

```
while 判断条件 1:
    while 判断条件 2:
        执行语句块 1
    执行语句块 2
```

当然，for 循环里也可以嵌套 while 循环，while 循环里也可以嵌套 for 循环，因为比较容易出错，不建议混用。

选择和循环彼此嵌套是指在一个选择里面嵌入一个循环，或者一个循环里嵌套一个选择，同样，将嵌套在里面的选择或循环当作一个语句来看，常用的有两种形式：

- 选择里嵌套循环，格式为：

```
if 判断条件:
    for 循环变量 in 迭代器:
        执行语句块 1
    执行语句块 2
else:
    执行语句块 3
```

- 循环里嵌套选择：常见的应用是穷举法，如鸡兔同笼问题的求解。格式为：

```
for 循环变量 in 迭代器:
    if 判断条件:
        执行语句块 1
        break
else:
    执行语句块 2
```

3.2 本章示例

【例 3.1】 用不同的流程控制方法实现：输入三角形三条边的边长，计算三角形的面积。

方法 1：

```
import math
a = input()
b = input()
c = input()
h = float(a+b+c)/2
area = math.sqrt(h*(h-a)*(h-b)*(h-c))
print "%.2f"%area
```

用例输入：

```
3
4
5
```

用例输出：

```
area=6.00
```

用例输入：

```
22.3
3.5
5.4
```

则无用例输出，程序运行报错。

代码解析：采用顺序结构实现，不存在对三角形三边数据是否合法的判断，当输入的三角形三边不能构成三角形时，无法计算三角形面积，所以无用例输出，运行后程序报错，提示错误原因是第 2 行代码语法无效，从根源上推测原因是输入有误！如图 3.1 所示。

方法 2：

```
import math
a = input()
```

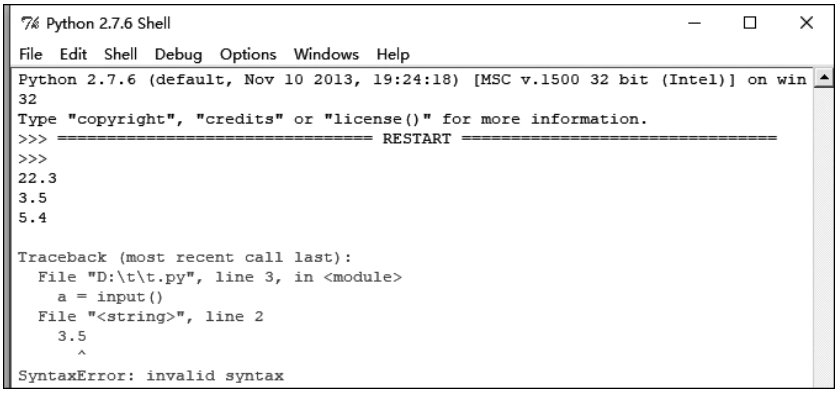

图 3.1　输入三角形三边运行出错界面

```
b = input()
c = input()
if (a+b>c) and (a+c>b) and (b+c>a) and (a>0) and (b>0) and (c>0):
    h = float(a+b+c)/2
    area = math.sqrt(h * (h-a) * (h-b) * (h-c))
    print "area=%.2f"%area
else:
  print " Three sides are illegal, can't form a triangle. "
```

用例输入：

```
3
4
5
```

用例输出：

```
area=6.00
```

用例输入：

```
22.3
3.5
5.4
```

用例输出：

```
Three sides are illegal, can't form a triangle.
```

代码解析：采用选择控制，对输入的三个边，不同的情况做出不同的处理，程序出现了分支，如果输入的三角形三边边长为合法数据时，程序才会求面积值，否则告诉用户数据有误，不能构成三角形。

【例 3.2】 用不同的选择结构方法实现：从键盘输入两个数 a 和 b，比较 a 和 b 的大小，保证降序输出。

方法 1：

```
a = input()
b = input()
if a<b:
    a,b=b,a
print a,b
```

用例输入：

```
3,4
```

用例输出：

```
4 3
```

代码解析：采用单分支选择结构，即如果 a＜b 则进行两数交换保证 a 大于 b。如果 a＞＝b，满足降序要求，则无需作处理。

方法 2：

```
a = input()
b = input()
if a<b:
    print b,a
else:
    print a,b
```

用例输入：

```
3,4
```

用例输出：

```
4 3
```

代码解析：采用双分支选择结构，即如果 a＜b，输出 ba，否则输出 ab。

【例 3.3】 用不同的选择结构方法实现：根据用户的身高和体重，计算用户的 BMI 值，并给出相应的健康建议。提示：BMI，即身体质量指数，是用体重（千克）除以身高（米）的平方得出的数字（BMI＝体重（kg）÷身高^2（m）），是目前国际上常用的衡量人体胖瘦程度以及是否健康的一个标准。先来看看成人的 BMI 数值：过轻：低于 18.5；正常：18.5-23.9；过重：24-27.9；肥胖：28-32；过于肥胖：32 以上。

方法 1：

```
#coding=GBK
height = input()
weight = input()
BMI = 1.0*weight/height/height
print "BMI=%.1f"%BMI
```

```
if BMI <18.5: print "偏瘦"
elif 18.5 <= BMI <24: print "正常"
elif 24 <= BMI <28: print "偏胖"
elif 28 <= BMI <32: print "肥胖!"
else: print "过胖"
```

用例输入：

```
1.62
60
```

用例输出：

```
BMI=22.9
正常
```

代码解析：采用 if-elif-else 多分支选择结构，但是没有考虑 elif 和 else 的隐含意义，所写的条件表达式比较烦琐。

方法 2：

```
#coding=GBK
height = input()
weight = input()
BMI = 1.0 * weight/height/height
print "BMI=%.1f"%BMI
if BMI <18.5: print "偏瘦"
elif  BMI <24: print "正常"
elif  BMI <28: print "偏胖"
elif  BMI <32: print "肥胖"
else: print "过胖"
```

用例输入：

```
1.62
60
```

用例输出：

```
BMI=22.9
正常
```

代码解析：采用 if-elif-else 多分支选择结构，巧妙利用了 elif 和 else 的隐含意义，所写的条件表达式简洁，推荐用这个方法来写多分支结构。

【例 3.4】 用不同的分离百位、十位、个位的方法实现：从键盘输入三位的正整数，输出其中的最大的一位数字是多少。例如：输入 386，输出 8。解题思路：首先需要从输入的三位数中分离出百位、十位和个位分别是多少，然后从百位，十位和个位中找出最大的一个数字。

方法 1：

```
#coding=utf-8
num = input()
a = num//100 #取 num 的百位数字
b = num//10%10#取 num 的十位数字
c = num%10#取 num 的个位数字
print a,b,c
if a>b and a>c: m = a
if b>a and b>c: m = b
if c>a and c>b: m = c
print m
```

用例输入：

```
673
```

用例输出：

```
6 7 3
7
```

代码解析：采用了整除“//”和求余“%”运算符来分离整数的每一位。

方法 2：

```
#coding=utf-8
num = input()
a = str(num)[0] #取 num 的百位数字
b = str(num)[1]#取 num 的十位数字
c = str(num)[2]#取 num 的个位数字
print a,b,c
if a>b and a>c: m = a
if b>a and b>c: m = b
if c>a and c>b: m = c
print m
```

用例输入：

```
673
```

用例输出：

```
6 7 3
7
```

代码解析：分离一个三位整数，利用字符串的索引取元素，因此，在索引取元素之前需要用 str 函数将输入的数据从数值类型转换为字符串类型。

【例 3.5】 用不同的循环语句实现：求 1 至 100 中所有整数的和。

方法 1：

```
s=0
for i in range(1,101):
    s=s+i
print s
```

无用例输入

用例输出：

```
5050
```

代码解析：采用 for 循环的方法，利用了 range 函数作迭代器。

方法 2：

```
s=0
i=1
while i<=100:
    s=s+i
    i=i+1
print s
```

无用例输入

用例输出：

```
5050
```

代码解析：采用 while 循环的方法，指定 i＜＝100 作继续运行条件(或停止条件)。

【例 3.6】 用循环方法实现：求 1 至 100 中奇数和偶数的和分别是多少。

```
sum_odd=0
sum_even=0
for i in range(1,101):
    if i%2==1:
        sum_odd=sum_odd+i
    else:
        sum_even=sum_even+i
print "sum_odd=",sum_odd
print "sum_even=",sum_even
```

无用例输入

用例输出：

```
sum_odd= 2500
sum_even= 2550
```

代码解析：与例 3.5 的方法 1 代码相比，实际上就是在 for 循环里嵌套一个双分支 if 语句判断奇偶。

【例 3.7】 用不同的循环控制语句实现：求 1 至 10 中能被 3 整除的数，并输出。

方法 1：

```
for i in range(1, 10+1):
    if i % 3 == 0:
        break
    print i,
```

无用例输入

用例输出：

```
1 2
```

代码解析：采用 break 控制循环，当 i 是 3 的倍数的时候，执行 break 语句。break 语句的作用是立刻结束整个 for 循环，因此输出只有 1 和 2 两个数字。

方法 2：

例 3.13：

```
for i in range(1, 10+1):
    if i % 3 == 0:
        continue
    print i,
```

无用例输入

用例输出：

```
1  2  4  5  7  8  10
```

代码解析：采用 continue 控制循环，当 i 是 3 的倍数时，执行 continue 语句。continue 语句的作用是结束这一轮的循环，程序跳转到循环头部，根据头部要求继续，因此输出不是 3 的倍数的所有数字。

【例 3.8】 用不同的循环控制语句实现：输出 0 至 4 中的数。

方法 1：

```
for i in range(5):
    print i,
else:
    print "for over"
```

无用例输入

用例输出：

```
0  1  2  3  4  for over
```

代码解析：没有 break 循环控制，因此遍历结束而正常退出循环，将会执行 else 中的语句。

方法 2：

```
for i in range(5):
```

```
    if i >= 3: break
    print i,
else:
    print "for over"
```

无用例输入

用例输出：

```
0  1  2
```

代码解析：有 break 循环控制，因此满足 if 条件，执行 break 语句，提前退出循环，将会导致 else 中语句不被执行。思考：如果将 break 换成 continue，结果是什么？

【例 3.9】 判断一个正整数 $n(n \geqslant 2)$是否为素数。素数又称质数。一个大于 1 的自然数，除了 1 和它自身外，不能被其他整数整除的数叫作素数；否则称为合数。

```
n = input()
for i in range(2, n):
 if n % i == 0:
  print "no"
  break
else:
  print "yes"
```

用例输入：

```
3
```

用例输出：

```
yes
```

用例输入：

```
8
```

用例输出：

```
no
```

代码解析：采用嵌套的语法实现，巧用循环中的 else 用法，即素数是从来没有满足 if 条件，当循环执行到最后，是正常退出时，才执行 else 子句。

【例 3.10】 采用不同嵌套方法实现：斐波拉契数列 1,1,2,3,5,8,……的前 n 项输出。

方法 1：

```
n=input()
a,b=1,1
for i in range(1,n+1):
    if i>1:
        a,b=b,a+b
```

```
    print a,
```

用例输入：

```
6
```

用例输出：

```
1 1 2 3 5 8
```

代码解析：输出时，输出数的后面跟着一个逗号，可实现多个数以空格分隔。

方法 2：

```
n=input()
a,b=1,1
for i in range(1,n+1):
    if i>1:
        a,b=b,a+b
    print a,
    if i%4==0:
        print
```

用例输入：

```
6
```

用例输出：

```
1 1 2 3
5 8
```

代码解析：输出时，除了输出数的后面跟着一个逗号，实现多个数以空格分隔之外，还增加了一个 if 单分支选择结构，实现输出时每隔 4 个换 1 行。

【例 3.11】 采用不同嵌套方法实现：输出 300 以内的素数。

方法 1：

```
for n in range(2,300):
    for i in range(2, n):
        if n % i == 0:
            break
    else:
        print n,
```

无用例输入

用例输出：

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239
241 251 257 263 269 271 277 281 283 293
```

代码解析：实现 300 以内素数的输出。

方法 2：

```
count=0
for n in range(2,300):
    for i in range(2, n):
        if n % i == 0:
            break
    else:
        count=count+1
print count
```

无用例输入

用例输出：

```
62
```

代码解析：实现统计 300 以内的素数个数。

【例 3.12】 采用穷举法实现：鸡兔同笼，这是中国古代著名的数学题之一。大约在 1500 年前，《孙子算经》中就记载了这个有趣的问题。书中是这样叙述的："今有雉兔同笼，上有三十五头，下有九十四足，问雉兔各几何？"这四句话的意思是：有若干只鸡和兔同在一个笼子里，从上面数，有 35 个头；从下面数，有 94 只脚。问笼中各有几只鸡和兔？

```
m,n=input()
for i in range(m+1):
    j=m-i
    if j>=0 and i*4+2*j==n:
        print j,i
        break
else:
    print "no result"
```

用例输入：

```
35,94
```

用例输出：

```
23 12
```

代码解析：穷举法思路是利用 for 循环穷举所有可能，用 if 进行条件验证，即 for 循环里嵌套 if 选择结构。

3.3 本章练习题

本章的练习题可以分为 3 次完成，第 1 次练习侧重选择结构语法掌握，第 2 次练习测重循环结构语法掌握，第 3 次练习侧重相对复杂的循环嵌套、循环选择嵌套等用法。

流程控制练习 1

【练习 3.1】 三角形面积计算。

题目描述：输入三角形三条边的边长，判断能否构成三角形，如果能构成三角形，则计算三角形的面积并输出(保留 2 位小数)，否则输出“不能构成三角形”。

用例输入：

```
22.3,3.5,5.4
```

用例输出：

```
不能构成三角形
```

用例输入：

```
3.12,4.3,5.789
```

用例输出：

```
6.58
```

【练习 3.2】 两数降序排序。

题目描述：从键盘输入两个数，要求降序(按由大到小的顺序)排序输出。

用例输入：

```
3
5
```

用例输出：

```
5 3
```

【练习 3.3】 成绩通过判断。

题目描述：从键盘输入百分制成绩，如果大于或等于 60，则输出"Congratulation! Pass."，否则输出"Sorry! Fail."。

用例输入：

```
55
```

用例输出：

```
Sorry!Fail.
```

用例输入：

```
98
```

用例输出：

```
Congratulation!Pass.
```

【练习 3.4】 水的状态。

题目描述：从键盘上输入一个温度值，输出所对应的水的状态。假设约定，0 度和 100 度为液态。

用例输入：

```
37
```

用例输出：

```
liquid
```

用例输入：

```
-5
```

用例输出：

```
solid
```

用例输入：

```
123
```

用例输出：

```
gas
```

【练习 3.5】 奇偶判断。

题目描述：从键盘输入一个正整数，判断是否奇数还是偶数。

用例输入：

```
21
```

用例输出：

```
odd
```

用例输入：

```
56
```

用例输出：

```
even
```

【练习 3.6】 船够坐吗？

题目描述：班级春游，有 40 人要过河，租 m 条小船（每条小船限乘 4 人）和 n 条大船（每条大船限乘 6 人），m 和 n 从键盘输入，请输出够坐还是不够坐？

用例输入：

```
8
1
```

用例输出：

```
no
```

用例输入：

```
6
3
```

用例输出：

```
yes
```

【练习 3.7】 成绩合法校验。

题目描述：从键盘输入成绩，判断是否合法。[0-100]为合法，其余为非法。

用例输入：

```
88
```

用例输出：

```
valid
```

用例输入：

```
120
```

用例输出：

```
invalid
```

【练习 3.8】 PM 指数。

题目描述：根据输入的雾霾 PM 指数值，判断空气质量，若 PM＜100 为“较好”；100≤PM≤150 为“轻度”；150＜PM≤200 为“中度”，PM 大于 200 为“重度”。

用例输入：

```
80
```

用例输出：

```
较好
```

【练习 3.9】 BMI 判断。

题目描述：从键盘输入用户的身高和体重，如果 BMI 属于 18.5～23.9，输出 yes(正常)，否则输出 no(异常)。

提示：BMI 即身体质量指数，是用体重(千克)除以身高(米)的平方得出的数字，即 BMI＝体重(kg)÷身高^2(m)。

用例输入：

```
1.65,60
```

用例输出：

```
yes
```

用例输入：

```
1.55,77.6
```

用例输出：

```
no
```

流程控制练习 2

【练习 3.10】 区间求和。

题目描述：计算区间[m,n)内的自然数的和并输出。m、n 从键盘输入。

用例输入：

```
1
5
```

用例输出：

```
10
```

【练习 3.11】 多项式求和。

题目描述：计算多项式 1 + 4 + 7 + 10 + 13 + 16 +…… 前 n 项和，n 从键盘输入。

用例输入：

```
5
```

用例输出：

```
35
```

【练习 3.12】 平均成绩。

题目描述：从键盘输入 10 个学生的成绩，要求计算平均成绩并输出(保留 1 位小数)。

用例输入：

```
21
42
13
54
75
96
77
98
29
```

```
100
```

用例输出：

```
60.5
```

【练习 3.13】 区间输出。

题目描述：将区间[m,n)内的数输出。m、n 从键盘输入。

提示：在区间求和题的基础上，将求和改为数的输出。

用例输入：

```
1
5
```

用例输出：

```
1 2 3 4
```

【练习 3.14】 区间偶数和。

题目描述：计算区间[m,n)内的自然数的偶数和并输出。m、n 从键盘输入。

提示：在区间求和题基础上，求和时加入偶数判断。

用例输入：

```
1
5
```

用例输出：

```
6
```

【练习 3.15】 计算偶数和。

题目描述：从键盘上输入 10 个数，输出其中偶数之和。

提示：在平均成绩题基础上，求和时加入偶数判断。

用例输入：

```
21
42
13
54
75
96
77
98
29
100
```

用例输出：

```
390
```

【练习 3.16】 区间偶数输出。

题目描述：将区间[m,n)内的偶数输出。m、n 从键盘输入。

提示：在区间输出题基础上，数的输出时加入偶数判断。

用例输入：

```
1
5
```

用例输出：

```
2 4
```

【练习 3.17】 多项式求和。

题目描述：计算多项式：1/1 + 1/4 + 1/7 + 1/10 + 1/13 + 1/16 +……前 n 项和，n 从键盘输入。

提示：1/4、1/7、1/10、1/13……按整数除都是 0？如何得到小数，1.0/分母或者干脆分母定义为实数类型。

用例输入：

```
5
```

用例输出：

```
1.5698
```

【练习 3.18】 多项式求和。

题目描述：计算多项式：1/1－1/4＋1/7－1/10＋1/13－1/16＋……前 n 项和，n 从键盘输入。

提示：可以设置一个标志 f＝1，在循环体内每次利用 f＝-f；实现正负符号切换。也可以不需要设标志 f，直接在循环体内，利用项数的奇偶特性，奇数项加，偶数项减。

用例输入：

```
5
```

用例输出：

```
0.8698
```

【练习 3.19】 斐波拉契数列第 n 项。

题目描述：斐波拉契数列：1，1，2，3，5，8，13，21，…，求这个数列的第 n 项并输出，n 从键盘上输入。

提示：该数列除了第 1、2 项是 1 之外，后面的项＝前两项之和。

用例输入：

```
4
```

用例输出：

```
3
```

【练习 3.20】 统计及格人数。

题目描述：从键盘输入 10 个学生的成绩，要求统计及格人数并输出。

提示：在平均成绩题基础上，将求和变成统计个数。

用例输入：

```
21
42
13
54
75
96
77
98
29
100
```

用例输出：

```
5
```

流程控制练习 3

【练习 3.21】 素数判断。

题目描述：从键盘上输入一个大于或等于 2 的正整数 m，判断它是否素数。m 从键盘输入。

用例输入：

```
3
```

用例输出：

```
yes
```

用例输入：

```
8
```

用例输出：

```
no
```

【练习 3.22】 又一个素数判断。

题目描述：输入一个整数，并判断是否是素数，如果是将输出它本身，否则输出其真因子的个数。

提示：真因子是指除它本身和 1 外没有其他约数。

用例输入：

```
3
```