第 3 章

程序的控制结构

学习目标

- 掌握传统流程图或伪代码描述算法的方法。
- 掌握顺序结构、选择结构和循环结构的用法。

扫一扫

视频讲解

3.1 程序设计的基本结构

3.1.1 算法的描述

从广义上讲，算法是解决一个问题所采取的方法与步骤。常用自然语言、传统流程图和伪代码表示算法。例如，怎样求一个三角形的面积？可以将这个问题分成三个步骤：输入、处理过程和输出，即 IPO（Input，Process，Output）模式，如图 3.1 所示。

输入：三角形的底和高
处理过程：三角形面积=底*高/2
输出：三角形的面积

图 3.1 求三角形面积的步骤

IPO 模式是系统分析和软件工程中广泛使用的方法，用于描述信息处理程序或其他过程的结构。

（1）输入（Input）：主要包括从文件读入、控制台输入、随机数据输入、交互界面输入、网络输入和程序内部参数输入等几种方式。

（2）处理过程（Process）：对输入数据的处理方法也称为“算法”，如求三角形的面积，可以用底乘高，再除以 2，也可以用“海伦公式”，这是不同的算法。

（3）输出（Output）：包括控制台输出、写出到文件中、网络输出以及操作系统内部变量输出等方式。

上面求三角形面积的例子是采用自然语言的方式描述算法的，但是对于复杂的问题，用自然语言描述时，如果程序发生多次跳转，则会降低程序的可读性。以下面三道例题为例，讲解描述算法的不同方法。

【例 3.1】 将变量 a 和 b 的值交换（a 和 b 的初值为 a=8，b=5）。

【例 3.2】 计算 z 的值，z=|a−b|（输入 a、b 的值）。

【例 3.3】 求 1～5 的累加和。

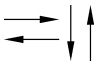
1. 自然语言

例 3.1	例 3.2	例 3.3
S1：a＝8，b＝5 S2：a＝5，b＝8 S3：输出 a 和 b 的值。	S1：输入 a 和 b 的值。 S2：判断 a＞b? 是：S21：z＝a－b， 转到 S3 步。 否：S22：z＝b－a， 转到 S3 步。 S3：输出 z 的值。	S1：i＝1，s＝0 S2：当 i≤5 时，执行 S21 和 S22，否则执行 S3。 S21：s＝s＋i S22：i＝i＋1，执行 S2。 S3：输出 s 的值。

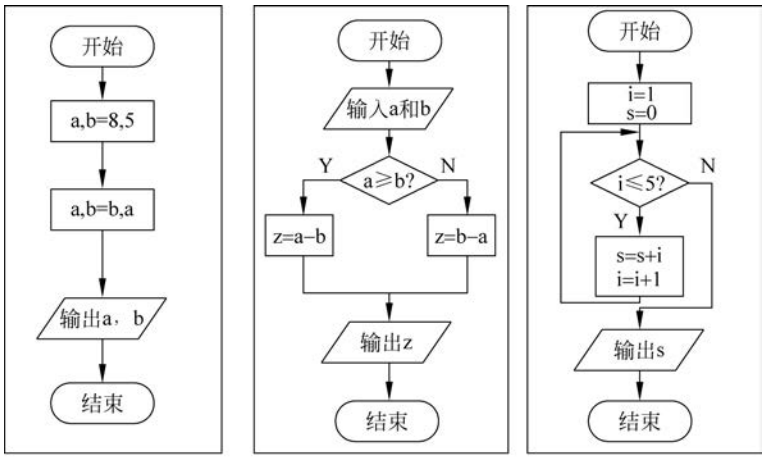
2. 传统流程图

常用的传统流程图的符号如表 3.1 所示。

表 3.1　传统流程图的符号

图　形	含　义	图　形	含　义
	起止框		注释框
	判断框		流向线
	处理框		连接点
	输入/输出框		

用传统流程图表示例题结果如图 3.2 所示。



(a) 例3.1　(b) 例3.2　(c) 例3.3

图 3.2　传统流程图

3. 伪代码

例 3.1	例 3.2	例 3.3

a=8　b=5
a=5　b=8
print(a,b)
	input(a,b)
	if　a>=b then
	　z=a-b
	else
	　z=b-a
	endif
	print(z)
		i=1　s=0
		while　i<=5
		　s=s+i
		　i=i+1
		end
		print(s)

自然语言描述算法虽然简单，但当循环较多时，不易直观地表述清楚；传统流程图比较直观，但当算法复杂时，传统流程图占篇幅较多，而且当程序转向较多时，难以阅读与修改。伪代码介于自然语言与计算机语言之间，用文字和符号描述算法，易于修改，其表达近似于编程语言，便于向程序过渡。

3.1.2 三种基本结构

在图 3.2 中，三个流程图分别对应程序设计中的三种基本结构：顺序结构、选择结构和循环结构。三道例题分别用 Python 实现的代码如下。

例 3.1	例 3.2	例 3.3

a,b=8,5
a,b=b,a
print(a,b)
	a = input('a=')
	b = input('b=')
	if a>=b:
	　print(a)
	else:
	　print(b)
		i=1
		s=0
		while　i<=5:
		　s=s+i
		　i=i+1
		print(s)

扫一扫

视频讲解

3.2 顺序结构

顺序结构是按照语句的先后顺序执行，几乎每个程序中都包含顺序结构。

【例 3.4】 将输入的摄氏温度转化为华氏温度。

```
c = float(input("请输入摄氏温度:"))
f = 9/5 * c + 32
print("华氏温度为:",f)

#运行结果
请输入摄氏温度:30
华氏温度为: 86.0
```

input()函数的返回值是字符型，通过 float()函数转换为实数类型并赋值给变量 c。根据摄氏温度与华氏温度之间的转换公式计算对应的华氏温度并输出结果。

扫一扫

视频讲解

3.3　选择结构

3.3.1　单分支结构

单分支结构格式：

```
if <表达式>:
    <语句块>
```

表达式可为算术表达式、关系表达式和逻辑表达式。当表达式成立时，执行语句块；当表达式不成立时，则不执行语句块。

【例 3.5】 比较 x 和 y 的值，如果 x＞y，则输出 x 的值。

```
x,y = 4,3
if x > y:
    print(x)
```

3.3.2　双分支结构

双分支结构类似于当走到有两条分支的岔路口时，只能选择其中一条岔路前进的情景，如图 3.3 所示。

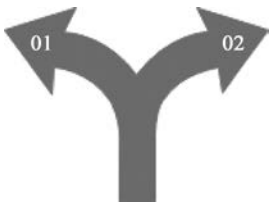


图 3.3　双分支结构

双分支结构的格式：

```
if <表达式>:
    <语句块 1>
else:
    <语句块 2>
```

当表达式成立时，执行语句块 1；当表达式不成立时，执行语句块 2。语句块中的语句由一条或多条语句组成，要注意缩进格式，如当语句块 1 由多条语句组成时，所有的语句同时缩进。

【例 3.6】 根据输入 AQI 的值，判断是否适宜户外运动。

```
AQI = eval(input("请输入 PM2.5 的值: "))
if AQI >= 75:
    print("不适宜户外运动")
else:
    print("适宜户外运动")

#运行结果
请输入 PM2.5 的值: 80
不适宜运动
```

其中 input()函数的返回值是字符型。eval() 函数用于执行一个字符串表达式，并返回表达式的值。简单地理解为，eval()函数将 input()的结果转化为数值赋给变量 AQI。

3.3.3 多分支结构

多分支结构类似于当面前有多条岔路时，只能选择其中一条岔路前进，如图 3.4 所示。

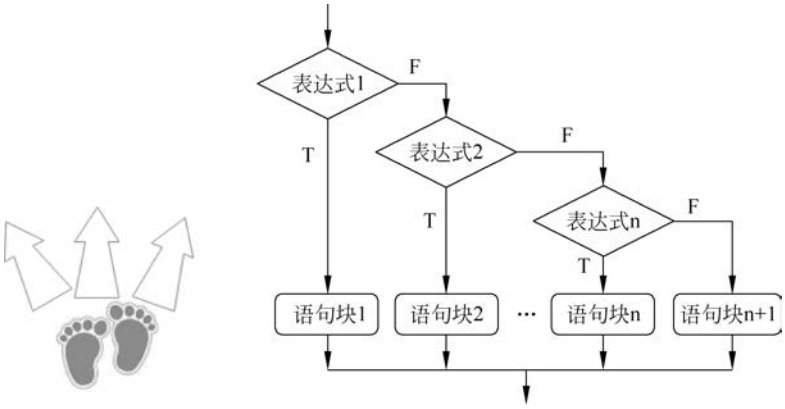


图 3.4 多分支结构

多分支结构的格式：

```
if <表达式 1>:
    <语句块 1>
elif <表达式 2>:
    <语句块 2>
elif <表达式 n>:
    <语句块 n>
        ⋮
else:
    <语句块 n+1>
```

（1）首先判断表达式 1 是否成立？如果成立，则执行语句块 1。

（2）当表达式 1 不成立时，判断表达式 2 是否成立？如果成立，则执行语句块 2。同理，判断表达式 n 是否成立？如果成立，则执行语句块 n。

（3）如果前面的表达式都不成立，则执行语句块 n+1。

【例 3.7】 求解分段函数。

$$y=\begin{cases}1, & x>0\\ 0, & x=0\\ -1, & x<0\end{cases}$$

```
x = int(input("请输入一个数:"))
if x > 0:
    y = 1
elif x == 0:
    y = 0
else:
    y = -1
print("y = ",y)
```

这个分段函数有三种可能，分别是 x 小于 0、x 等于 0、x 大于 0，在使用 if…elif…else 结构时，注意以下几点。

（1）每个保留字后面有冒号“：”。

（2）else 与 if 的配对原则：else 与离它最近的未配对的 if 配对。

（3）每个保留字后边的一条或多条语句的缩进层次相同。

3.3.4　紧凑结构

```
if <表达式>:
    <语句块 1>
else:
    <语句块 2>
```

分支选择的 if…else…形式，可以写成等价的紧凑形式：

```
<语句块 1> if <表达式> else <语句块 2>
```

紧凑形式的含义为，判断表达式是否成立，如果成立，则执行语句块 1，否则执行语句块 2。

【例 3.8】 if…else…紧凑形式示例一。

```
a,b = 5,4
c = a if a > b else b
print(c)
#等价于
a,b = 5,4
if a > b:
    c = a
else:
    c = b
print(c)
```

【例 3.9】 if…else…紧凑形式示例二。

```
count = 2
"存在" if count!= 0 else "不存在"#如果 count!= 0 成立,则显示"存在",否则显示"不存在"

#运行结果
'存在'
count = 0
"存在" if count!= 0 else "不存在"

#运行结果
'不存在'
```

扫一扫

视频讲解

3.4　循环结构

循环结构类似于小朋友跳绳，将一个动作反复执行。Python 中的循环语句主要包括 for 语句和 while 语句，同时常结合 break 语句和 continue 语句一起使用。

3.4.1 for 语句

for 语句常与保留字 in 一起使用，同时用 range()函数控制循环次数。

1. range()函数用法一

range(start, stop[, step])，range()函数可以创建一个整数列表，一般用在 for 循环中。参数 start，stop，step 分别表示起始值、终止值、步长。range()函数的参数含义如表 3.2 所示。

表 3.2 range()函数的参数含义

参　数	含　义
start	起始值，缺省值为 0
stop	终止值，循环变量不取 stop 的值
step	步长，缺省值是 1

【例 3.10】 range()的用法示例。

在语句 for i in range(5,10)中，5 是起始值，10 是终止值，默认步长是 1。i 的取值范围是 5～9，不包括 10。end＝''为控制数据输出时，以空格分隔，并显示在一行。

```
for i in range(5,10):
print(i,end = ' ')
#运行结果
5 6 7 8 9

for i in range(0, 10, 3):
print(i,end = ' ')
#运行结果
0 3 6 9

for i in range( - 10, - 100, - 30):
print(i,end = ' ')
#运行结果
- 10 - 40 - 70
```

range(0,10,3)表示取值范围为 0～10，以步长为 3 取值，得到一个等差数列。

2. range()函数用法二

格式：

```
for 循环变量 range (循环次数):
    语句块
```

【例 3.11】 累加求和。

for i in range(1,10)，i 的取值范围为 1～9，不包括 10，即循环 9 次。for i in range(10)，i 的取值范围为 0～9，即循环 10 次，对 i 累加求和。

```
sum = 0
for i in range(1,10):
    sum = sum + i
print(sum)
#运行结果
45
```

```
sum = 0
for i in range(10):
    sum = sum + i
print(sum)
#运行结果
45
```

3.4.2　while 语句

格式：

```
while <表达式>:
    <语句块 1>
else:
    <语句块 2>
```

while 语句的表达式为真时，则执行语句块 1，否则执行语句块 2。While 循环语句中，一般包含使循环趋于结束的语句，如 i=i+1。

【例 3.12】 用 while 语句计算 1～10 的累加和。

```
i = 1
sum = 0
while i < 11:
    sum = sum + i
    i = i + 1
print(sum)
#运行结果
55
```

思考：修改程序，求 1～10 的奇数和、偶数和。

【例 3.13】 读程序，观察 while…else 中 i 值的变化。

```
i = 0
while i < 5:
   print(i, " is less than 5")
   i = i + 1
else:
   print(i, " is not less than 5")
#运行结果
0  is  less than 5
1  is  less than 5
2  is  less than 5
3  is  less than 5
4  is  less than 5
5  is not less than 5
```

一般情况下，for 语句与 while 语句是通用的。当循环次数固定时，用 for 语句比较容易清晰地表示，例如，计算 100～200 的素数；当循环次数不固定时，常用 while 语句，如计算 1－1/3＋1/5－1/7＋…，直到某一项的绝对值小于 10^{-6} 为止（该项不累加）。

3.4.3 循环嵌套

当输出九九乘法表或画图案时，如钻石星形，用一个循环变量难以清晰地同时表示行和列，这时需要循环嵌套来解决问题。

【例 3.14】 输出 4×4 的乘法表的值。

```
for i in range(1,5):
    for j in range(1,5):
        print("{:<4d}".format(i*j),end='')
    print('')
#运行结果
1   2   3   4
2   4   6   8
3   6   9   12
4   8   12  16
```

i 控制行，j 控制列，i 和 j 的取值范围均是 1～4。控制输出数据的输出格式，“<”表示左对齐，输出的数据宽度为 4。end=''表示数据间以空格分隔。

【例 3.15】 修改例 3.14，只输出例 3.14 输出结果的下三角。

```
for i in range(1,5):
    for j in range(1,i+1):
        print(" {:d}".format(i*j),end='')
    print('')
#运行结果
1
2  4
3  6  9
4  8 12  16
```

3.4.4 break 与 continue 语句

break 和 continue 语句一般与循环语句结合使用。break 语句表示跳出当前循环；continue 语句表示结束本次循环，进行下一次判断。

【例 3.16】 计算下列程序段的输出结果。

```
sum=0
for i in range(1,11):
    if i>5:
        break
    sum=sum+i
print(sum)
#运行结果
15

sum=0
for i in range(1,11):
```

```
    if i<=5:
        continue
    sum=sum+i
print(sum)
#运行结果
40
```

在例 3.16 中，第一个程序 i 取值范围为 1～10，在 i 取值 1～5 时，执行累加。当 i>5 时，则跳出 for 循环。程序执行了 5 次循环，运行结果为 15。第二个程序中，当 i<=5 时，不执行累加求和语句“sum=sum+i”，当 i 取值 6～10 时，执行累加求和语句“sum=sum+i”，运行结果为 40。

【例 3.17】 字符串输出控制。

```
for ch in "Happy New Year!":
    if ch=="N":
        continue
    print(ch, end="")
#运行结果
Happy ew Year!

for ch in "Happy New Year!":
    if ch=="e":
        break
    print(ch, end="")
#运行结果
Happy N
```

例 3.17 中，第一个程序中的变量 ch 遍历字符串 Happy New Year!，当 ch==“N”时，结束本次循环，然后 ch 取 e 值，进入下一次循环，直至循环结束，因此运行结果中不包含字符 N。第二个程序中的变量 ch 遍历字符串 Happy New Year!，当 ch==“e”时，跳出循环，因此运行结果为 Happy N。

扫一扫

视频讲解

3.5　实例

【例 3.18】 “水仙花数”是指一个三位数，其各位数字的立方之和等于该数本身，例如，$153=1^3+5^3+3^3$。判断数字 153 是否为水仙花数。

这是一个典型的数字分离问题，需要把每一位从数中分离出来，百位为 n//100，十位为 n//10%10，个位是 n%10 取余数。

```
n=153
i = n // 100
j = n // 10 % 10
k = n % 10
if n == i*i*i + j*j*j + k*k*k:
    print("{:d}是水仙花数".format(n))
else:
```

```
    print("{:d}不是水仙花数".format(n))
#运行结果
153是水仙花数
```

【例 3.19】 求解分段函数。

有如下分段函数：

$$y=\begin{cases}|x|+1, & x<0\\ x^2+1, & 0<x<1\\ \sqrt{2x}-1, & x>1\end{cases}$$

编写程序，输入自变量 x 的值，计算 y 的值并输出。

```
import math
x = eval(input("请输入数据:"))
if x<0:
    y = abs(x) + 1
elif x>0 and x<1:
    y = x ** 2 + 1
else:
    y = math.sqrt(2 * x) - 1
print(y)
```

【例 3.20】 输入一行字符，分别统计其中字母、空格、数字和其他字符的个数。

isalpha()方法用于检测字符串是否只由字母组成。isspace() 方法用于检测字符串是否只由空格组成。isdigit()方法用于检测字符串是否只由数字组成。

```
import string
s = input('请输入一个字符串:')
letters,space,digit,others = 0,0,0,0
i = 0
while i < len(s): #len( )返回字符串长度
    k = s[i]
    i += 1
    if k.isalpha( ):
        letters += 1
    elif k.isspace( ):
        space += 1
    elif k.isdigit( ):
        digit += 1
    else:
        others += 1
print('letters = %d,space = %d,digit = %d,others = %d' % (letters,space,digit,others) )
```

【例 3.21】 冒泡法排序，将列表[9,6,5,3,1]中的数值从小到大排序。

```
arr = [9,6,5,3,1]
n = len(arr)
for i in range(n-1):
    for j in range(0,n-i-1):
```

```
        if arr[j] > arr[j + 1] :
            arr[j], arr[j + 1] = arr[j + 1], arr[j]
print(arr)

#运行结果
[1, 3, 5, 6, 9]
```

冒泡法解题思路：将相邻的两个数进行比较，小数放前面，大数放后面。经过几轮比较与交换位置，将排序后的结果输出，如图 3.5 所示。

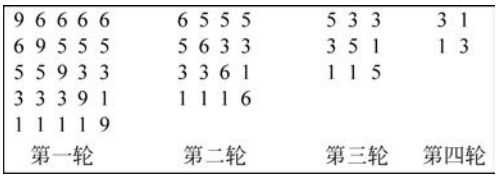


图 3.5　冒泡法排序

例如，有 5 个数 9,6,5,3,1。

第一轮：

(1) 第一次是 9 和 6 比较，根据解题规则(两两比较，小数放前，大数放后)，9＞6,6 放在 9 前面，5 个数的顺序变成 6,9,5,3,1。

(2) 第二次是 9 和 5 比较，9＞5,5 放在 9 前面，5 个数的顺序变成 6,5,9,3,1。

(3) 第三次是 9 和 3 比较，9＞3,3 放在 9 前面，5 个数的顺序变成 6,5,3,9,1。

(4) 第四次是 9 和 1 比较，9＞1,1 放在 9 前面，5 个数的顺序变成 6,5,3,1,9。

这样，在第一轮中，找到了 5 个数中的最大数 9。

第二轮：6,5,3,1，比较方法同第一轮，找到第二大数 6。

第三轮：5,3,1，同理，找到第三大数 5。

第四轮：3,1，同理，找到第四大数 3。

说明：

(1) 轮数：如果用 n 表示数的个数，比较的轮数就是 n－1，如 n＝5，5 个数比较 n－1 轮，即 4 轮。

(2) 每一轮比较次数：n－i－1，i 是循环变量，取值 0～3。

(3) arr[j],arr[j＋1]＝arr[j＋1],arr[j]，两个变量的值交换，这里不再赘述。

3.6　习题

一、单选题

1. 下列 Python 保留字中，不用于表示分支结构的是(　　)。

A. elif　　B. in　　C. if　　D. else

2. 实现多路分支的控制结构是(　　)。

A. if　　B. try　　C. if…elif…else　　D. if…else

3. 下列选项中，能够实现 Python 循环结构的是(　　)。

A. loop　　B. do…for　　C. while　　D. if

4. 关于 Python 的循环结构，下列选项中描述错误的是（　　）。

A. break 用来跳出最内层 for 或 while 循环，程序从循环代码后继续执行

B. 每个 continue 语句只能跳出当前层次的循环

C. 遍历循环中的遍历结构可以是字符串、文件、组合数据类型和 range()函数等

D. continue 结束整个循环过程，不再判断循环的执行条件

5. 执行下列代码，下列选项中描述正确的是（　　）。

```
sum = 0
for i in range(1,11):
  sum += i
  print(sum)
```

A. 循环内语句块执行 11 次

B. 输出的最后一个数是 66

C. 如果 print(sum) 语句不缩进，输出结果不变

D. 输出的最后一个数是 55

6. 关于 Python 的控制结构，下列选项中描述错误的是（　　）。

A. 每个 if 条件后要使用冒号“:”

B. 在 Python 中，没有 switch…case 语句

C. Python 中的 pass 是空语句，一般用作占位语句

D. elif 可以单独使用

7. 执行下列语句，a、b、c 的值分别是（　　）。

```
a = "water"
b = "juice"
c = "milk"
if a > b:
  c = a
  a = b
  b = c
```

A. water juice milk　　B. water milk juice

C. juice milk water　　D. juice water water

8. 执行下列代码，输出结果是（　　）。

```
for s in "Hello,Python":
    if s == "P":
        continue
    print(s,end = "")
```

A. Hello　　B. Hello,ython　　C. Hello,Python　　D. Python

9. 执行下列代码，输出结果是（　　）。

```
for s in " Hello,Python":
    if s == "P":
        break
    print(s,end = "")
```

A. Hello,　　B. Hello,ython　　C. Hello,Python　　D. Python

10. 执行下列代码，输出值的个数是(　　)。

```
num = 11
start = 1
if num % 2!= 0:
    start = 1
for x in range(start,num + 2,2):
    print(x)
```

A. 6　　B. 7　　C. 11　　D. 10

二、编程题

1. 从键盘输入 3 个数作为三角形的边长，输出由这 3 条边构成的三角形的面积(保留两位小数)。

2. 编程实现：学习成绩大于或等于 90 分用 A 表示，60～89 分用 B 表示，60 分以下的用 C 表示。

3. 编程实现：判断输入的一个整数能否同时被 3 和 7 整除，若能整除则输出 Yes，否则输出 No。

4. 编写程序，根据输入的年份(4 位整数)判断该年份是否为闰年。

5. 编程实现：一个简单的出租车计费系统，当输入行程的总里程时，输出乘客应付的车费(保留一位小数)。计费标准具体为起步价 10 元/3km，超过 3km 的费用为 1.2 元/km，超过 10km 的费用为 1.5 元/km。

6. 编程实现：输出 1～100 的奇数。

7. 编程实现：数字逆序输出，从控制台输入三位数，如 123，逆序输出 321。

8. 编程求解：有四个数字 1、2、3、4，能组成多少个互不相同且无重复数字的三位数？请写出结果。

9. 编程实现：求 1＋2!＋3!＋…＋20!。

10. 编程实现：有一分数序列 2/1，3/2，5/3，8/5，13/8，21/13，…，求这个数列的前 20 项之和。

11. 编程实现：输出钻石图形。

```
    *
   ***
  *****
 *******
  *****
   ***
    *
```

12. 编程实现：用 for 循环输出九九乘法表。

13. 编写程序，输出斐波那契数列的前 20 项，要求每行输出 5 项。

14. 编程实现：输出 100～1000 的所有水仙花数。

15. 编写程序，计算 s＝a＋aa＋aaa＋…$\underbrace{+aaa\cdots aaa}_{n个}$ 的值，a 为 1～9 中的某个数字，n 是一个正整数。例如，当 a＝2，n＝5 时，s＝2＋22＋222＋2222＋22222＝24690。