第 3 章 程序流程控制

3.1 算 法 概 述

3.1.1 算法及其要素和特性

算法是指为解决问题而采取的方法和步骤。

算法的要素有以下两部分：

（1）对数据对象的运算和操作。

（2）算法的控制结构（运算和操作时间的顺序）：顺序结构、循环结构和选择结构。其中顺序结构是最简单也最常用的结构，它的执行顺序是自上而下，依次执行。其余两种结构接下来会介绍。

算法的特征有如下几方面。

有穷性：算法的有穷性是指算法必须能够在执行有限步之后停止。

确切性：算法的每步都要有确切的定义。

输入项：一个算法要有 0 个或多个输入项，用来反映问题的原始状态，如果是 0 个输入项，则是算法有初始条件。

输出项：算法都有输出项，可以是一个也可以是多个输出项，用来反映对数据加工处理后的结果。

可行性：即算法的每个步骤都能在有限时间内完成。

因为计算机的运算速度并不是无限快的，所以在设计算法时一定要注意时间资源，同样，存储器的空间也是有限的，所以在设计算法时一定要尽可能地节约时间和空间两方面的开销。

3.1.2 算法表示方法

1. 用自然语言表示

该方法就是直接用自然语言描述算法。一般除了很简单的问题外，不用自然语言表示。

2. 用流程图表示

流程图可以很直观地表现出算法的过程，易于理解。流程图主要由图 3.1 所示的 4 种框加上流程线组合而成。

图 3.2 描述判断输入的年份是否为闰年的流程。

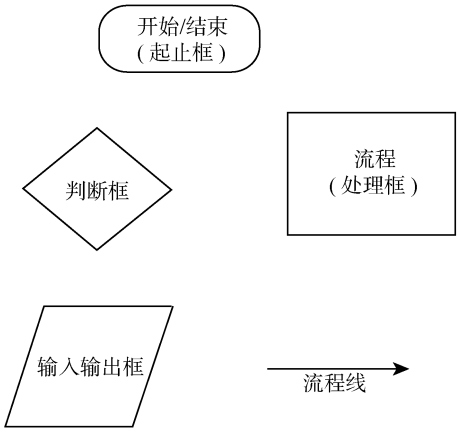

图 3.1　流程图的部件

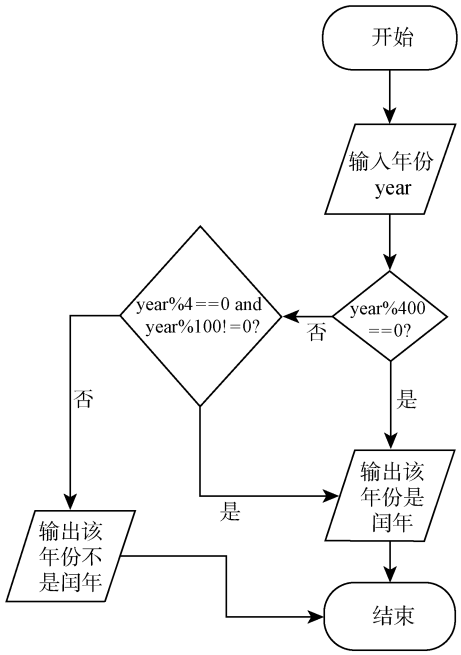

图 3.2　判断年份是否为闰年的流程图

3. 用伪代码表示

伪代码是一种用来书写程序或描述算法时使用的非正式、透明的表述方法。伪代码通常采用自然语言、数学公式和符号相结合来描述算法的操作步骤，同时采用计算机高级语言的控制结构来描述算法步骤的执行。只要自己或者别人能看懂即可。下面看一个例子。

用伪代码表示求一个列表中最大元素值的算法。

```
MaxElement(a_list : list)
#求一个列表中的最大元素
#a_list : list 代表输入的数据是一个 list 类型（关于 list 会在后面详细讲解）
#输出 a_list 中的最大元素
```

```
max_element = a_list[0]
for i ← 1 to len(a_list) - 1 do
      if  list[i] > max_element
           max_element = list[i]
return max_element
```


这里只是举一个例子，实际上，在 Python 中，如果要求一个列表的最大元素值，并不需要这么麻烦，只需要调用 Python 的内置函数 max()即可。

当然算法的表示方法还有很多种，在这里只是选取了比较常用的 3 种进行讲解。

3.2 顺 序 结 构

顺序结构是结构化程序设计中的基本结构，在该结构中，各语句或语句组按照出现的先后顺序依次执行，如图 3.3 所示。在选择结构和循环结构中，顺序结构也是组成部分。

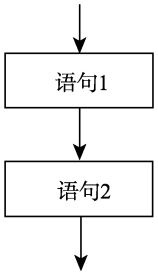

图 3.3 顺序结构流程图

【例 3.1】 输入 3 个数，计算这 3 个数的平均值。

```
a = float(input(''请输入 a 的值: ''))
b = float(input(''请输入 b 的值: ''))
c = float(input(''请输入 c 的值: ''))
f = (a + b + c) / 3
print(str.format(''3 个数的平均值为: {:%.2f}'',f))
```

3.3 选 择 结 构

用 if 语句可以构成选择结构。它根据给定的条件进行判断，以决定执行某个分支程序段。Python 的 if 语句有 3 种基本形式。

3.3.1 if 选择结构

该结构形式为：

```
if 条件:
    执行的操作 1
    执行的操作 2
```

其流程图如图 3.4 所示。

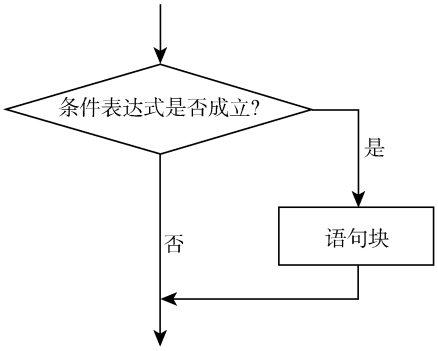

图 3.4　单分支选择结构流程图

该结构的意思是，如果条件为真则执行操作 1 和操作 2。除了 False（包括表达式的值为 False）、None、各种数据类型的 0 以及空的序列与空的字典外，其余的都可以看成条件为真。

需要注意的是，在 Python 中，如果后面的语句需要缩进，那么在该行代码末尾需要加冒号，看下面的例子。

【例 3.2】 判断一个输入的数是否为偶数。

```
b=input()
a=int(b)
if a%2==0:
    print('%d是偶数'%a)
```

在上述例子中，如果想这个数在不是偶数时也将结果打印出来，就需要用到 if…else 结构。当条件为真时，执行条件语句下的嵌套语句，否则执行 else 部分。

```
b=input()   #输入一个数
a=int(b)    #将输入转换为
if a%2==0:
    print('%d是偶数'%a)
else:
    print('%d不是偶数'%a)
```

可以看出 if…else 的语句结构为：

```
if 条件:
   操作1
else:
   操作2
```

即当条件满足时，执行操作 1；当条件不满足时，执行操作 2。一个 if 只能和一个 else 搭配。

其流程图如图 3.5 所示。

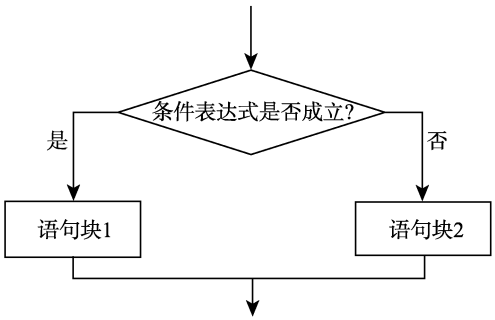

图 3.5　双分支选择结构流程图

但有时所需要的结果不是一次选择能得到的，可能需要多次判断，这就需要用到 if…elif…else 结构。

```
if 条件 1:
   操作 1
elif 条件 2:
   操作 2
else:
   操作 3
```

其流程图如图 3.6 所示。

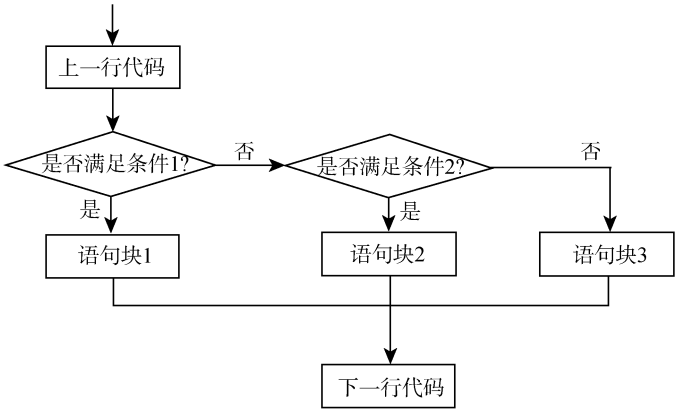

图 3.6　多分支选择结构流程图

在 if…elif…else 结构中，可以有多个 elif 语句，但只能有一个 else 语句，当所有的条件都为假时，才执行 else 部分。

下面用一个例子说明。

【例 3.3】 判断输入的年份是否为闰年。

```
str_year=input()    #输入年份
year=int(str_year) #将字符串转换为数字
if year%400==0:
    print("%d是闰年"%year)
```

```
elif year%4==0 and year%100!=0:
    print("%d 是闰年"%year)
else:
    print("%d 不是闰年"%year)
```

与 C 语言等不同的是，Python 中没有 switch 和 case 语句。但是 Python 可以用一系列的 if、elif 和 else 语句来达到相同的效果。

3.3.2 选择结构的嵌套

当 if 语句的操作语句中还有 if 语句时就构成了选择结构的嵌套，例如：

```
if 条件 1:
   if 选择结构
elif 条件 2:
   if 选择结构
else:
   if 选择结构
```

但是在使用选择结构的嵌套时，要注意 if 和 else 的搭配，一对 if 和 else 一定要对齐，下面用一个例子说明。

【例 3.4】 判断一个输入的数是否是偶数，是否能被 3 整除。

```
a=input()   #输入一个数
b=int(a)    #转换为整型
if b%2==0:
    if b%3==0:
        print("该数是偶数且能被 3 整除")
    else:
        print("该数是偶数但不能被 3 整除")
else:
    if b%3==0:
        print("该数不是偶数但能被 3 整除")
    else:
        print("该数不是偶数且不能被 3 整除")
```

3.4 循 环 结 构

有时想让同一个指令重复执行多次，如打印数字 1~10：

```
print(1)
print(2)
…
print(10)
```

这样的笨办法看起来好像也还能接受，那如果打印数字 1~10 000 呢，这时再用这种方法就不合适了，为了避免这种笨重的代码，就需要用到循环结构，接下来介绍几种循环结构。此外，还会介绍几种在循环中常用的内置函数。

3.4.1 while 循环结构

【例 3.5】 用 while 循环结构打印数字 1～10。

```
x=1
while x<=10:
    print(x)
    x+=1
```

是不是和之前的代码比起来，简洁了很多?

while 循环结构的一般形式为：

```
while 表达式:
    操作语句
```

其中，表达式是循环条件，操作语句为循环体。条件为真时执行循环体内操作，当执行完一次操作后，再对条件进行判断，如果条件继续为真，则继续执行循环体，然后再判断条件，直到条件为假退出 while 循环。简而言之，只要顶端的表达式为真，就会重复执行语句块。如果表达式一开始就为假，则循环体语句块不会执行。

注意，表达式后面是有冒号的。前面提到了，后面代码需要缩进的都需要以冒号结尾。

当想执行一个无限循环操作时，可以用“while True”，这样条件就能一直为真。

while 循环结构流程图如图 3.7 所示。

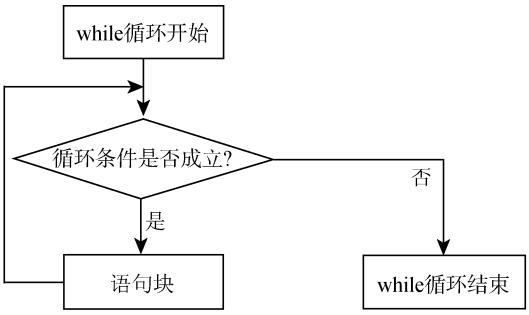

图 3.7　while 循环结构流程图

【例 3.6】 无限循环例子。

```
>>> while True:
        print('Python')
Python
Python
Python
Python
Python
…
```

循环条件可以是任意的对象，前面已经介绍过了，在 Python 中，所有的对象都有对应的布尔值。

3.4.2 for 循环结构

用 while 循环结构可以实现很多循环，但如果想遍历一个序列（关于序列会在后面的章节中讲到，这里可以把它理解成一个集合）中的元素则可以使用 for 循环结构。for 循环结构的一般格式为：

```
for x in object:
    语句块
```

其中，x 是定义的一个用来赋值的变量，object 是要遍历的对象。for 循环结构执行时，逐个将序列对象中的值赋给 x，就可以在语句块中对 x 进行操作。x 的作用域（作用域后面会讲到）只在 for 语句里，可以在循环主体中对 x 进行修改。但当回到循环顶端时，又会被自己赋值成下一个元素。

for 循环结构流程图如图 3.8 所示。

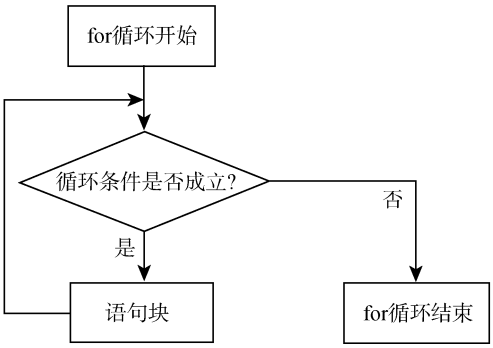

图 3.8　for 循环结构流程图

【例 3.7】 用 for 循环将数字 1~5 打印出来。

```
>>> for x in range(1,6):
...     print(x)
...
1
2
3
4
5
```

当需要遍历某个序列对象时，应优先选用 for 循环结构。当把表达式的值作为循环条件时应当选用 while 循环。如果不需要每个元素都访问或者想要改变列表的值，这时候也应该选用 while 循环。

Python 的内置函数 enumerate()实现遍历元素，它多用于在 for 循环中得到计数。

对于一个可迭代的对象，enumerate()将其组成一个索引序列，利用它可以同时获得索引和值。enumerate()返回的是一个 enumerate 对象。

```
>>> list_a=['a','b','c','d']
```

```
>>> print(type(enumerate(list_a)))
<class 'enumerate'>
```

【例 3.8】 enumerate()的用法。


```
>>> list_a=['a','b','c','d']
>>> for index,data in enumerate(list_a):
        print(index,data)
0 a
1 b
2 c
3 d
>>>
```

需要注意的是，for 循环只是把序列中对象的值赋给了另一个对象 x，也就是说对 x 的值进行的操作不会影响序列中对象的值，除非直接对序列进行操作。

```
>>> list_a=['a','b','c','d']
>>> for data in list_a:
        data='d'
        print(data)
d
d
d
d
>>> print(list_a)
['a', 'b', 'c', 'd']
>>> for i in range(len(list_a)):
        list_a[i]='d'
>>> print(list_a)
['d', 'd', 'd', 'd']
>>>
```

任何序列都适用于 for 循环，上面已经看到了 for 循环可以用于列表，此外还可以用于字符串、元组等，甚至字典和文件都可以用 for 循环。

【例 3.9】 for 循环用于字符串。

```
>>> string='hello world'
>>> for i in string:
        print(i)
h
e
l
l
o

w
o
r
l
d
>>>
```

3.4.3 break 和 continue 语句

break 和 continue 语句需要嵌套在循环结构中才能起作用。break 语句用来跳出所在的最近的一个循环结构，而 continue 语句则是跳到所在的最近的循环结构的首行即开头处重新判断条件。下面是两个例子。

【例 3.10】 打印数字 0~5。

```
>>> x=0
>>> while True:
...     if x>5:
...         break
...     print(x,end=' ')
...     x+=1
...
0 1 2 3 4 5
```

在这里虽然用了 while True 是一个无限循环，但因为当 x>5 后，break 语句执行，跳出了这个无限循环。

【例 3.11】 打印数字 0~5 中除了 3 之外的数。

```
>>> for x in range(6):
...     if x==3:
...         continue
...     print(x,end =' ')
...
0 1 2 4 5
```

可以发现在 continue 语句执行后，直接跳到了循环的开头，进行下一次操作，而没有执行 print 语句。

3.4.4 else 语句

循环里的 else 语句是在循环正常结束或循环一次也没执行时才执行的，非正常结束即通过 break 语句结束循环。

【例 3.12】 判断输入的一个数是否为素数。

```
import math
num=int(input('输入一个正整数，判断是否是素数：'))
if num<0:
    print("输入非法")
elif num == 1:
    print("%d既不是素数也不是合数" % num)
else:
    for i in range(2,int(math.sqrt(num))+1):#math.sqrt(x)是求 x 的开方值
        if num%i==0:
            print("%d不是素数"%num)
            break
    else:
        print("%d是素数"%num)
```

当循环正常结束时，输入的数除了 1 和此整数自身外，没法被其他自然数整除，说明是一

个素数，就需要执行 else 语句，而当通过 break 语句跳出循环时就说明不是素数，也就不需要执行 else 语句了。

循环 else 语句和哪一个 while 或 for 循环搭配就要与之保持相同的缩进。如果在 else 语句之前有多个循环结构且都与 else 语句缩进相同，那么 else 语句便是与在它前面且离它最近那一个循环结构搭配。

这里出现了一个 import 语句，用来导入 Python 的包，后面章节会讲到。

如果循环没有被执行过，循环 else 语句也会执行，因为没有执行过 break 语句。当首行条件一开始就为假时就会出现这种情况。

如果想在一个序列中搜索某一个元素，也可以使用循环 else 语句。

```
list_a = ['a', 'b', 'c', 'd', 'e']
for index, ch in enumerate(list_a):
    if ch == 'f':
        print(index)
        break
else:
    print('未找到')
```

3.4.5 pass 语句

pass 语句是无运算的占位语句，当语法需要语句但又没有实用的语句可以写时，就可以用 pass 语句占位。例如，想要一个无限循环，但每次迭代时又什么都不做，此时可以在循环体内用 pass 语句。

在定义函数时，可以先用 pass 语句填充函数体。待以后再用真正的函数体代替 pass 语句。因为无法使函数体为空而不产生语法错误，所以只能先用 pass 代替。

此外，如果想忽略 try 语句捕获的异常，也可以使用 pass 语句，异常会在后面讲到。

3.4.6 循环结构的嵌套

就像选择结构可以嵌套一样，循环结构也可以进行嵌套。可以将 while 循环结构嵌套进 for 循环结构，也可以将 for 循环结构嵌套进 while 循环结构。

for 循环嵌套进 while 循环结构的格式如下：

```
while 表达式:
    for 循环结构
```

while 循环嵌套进 for 循环结构的格式如下：

```
for x in object:
    while 循环结构
```

continue、break 以及 pass 语句也可以添加到嵌套的循环结构中，但是一定要搞清楚语句的作用域，break 语句只能跳出离它最近的那个 while 或 for 循环结构，不能结束整个循环结构。同理，continue 语句也只能回到离它最近的那个 while 或 for 循环结构的开头，而不是整个循环结构的开头。

3.5 实 例 精 选

【例 3.13】 打印九九乘法表。

```
for i in range(1, 10):
      for j in range(i, 10):
            print("%d*%d=%2d" % (i, j, i * j), end=" ")
print("") #换行
```

本例的输出结果如下：

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

在这里出现了一个函数 range()，range()在 Python 3.0 中是一个迭代器，会根据需要产生元素，当传入一个参数时，range()会产生 0 到该数的整数列表，包括 0 但不包括传进去的参数值；当传入两个参数时，第一个参数会作为下边界，第二个参数是上边界（产生的整数列表包括下边界但不包括上边界）当传入 3 个参数时，第 3 个参数作为步长。例如：

```
>>> list(range(5,-5,-2))
[5, 3, 1, -1, -3]
```

【例 3.14】 输出 1000 以内的水仙花数。

水仙花数是指一个 n（n≥3）位数，它的每个位上的数字的 n 次幂之和等于它本身。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
def main():
    for i in range(100,1000):
        a = i%10
        b = i//100
        c = (int(i/10))%10
        if i == a**3+b**3+c**3:
            print("%5d"%(i))

if __name__ == "__main__":
    main()
```

【例 3.15】 有 1、2、3、4 共 4 个数字，能组成多少个互不相同且无重复数字的 3 位数？都是多少？

```
cnt = 0  # count the sum of result
for i in range(1, 5):
    for j in range(1, 5):
```

```
            for k in range(1, 5):
                if i != j and i != k and j != k:
                    print(i * 100 + j * 10 + k)
                    cnt += 1
print(cnt)
```

【例 3.16】 企业发放的奖金根据利润提成。利润（i）低于或等于 10 万元时，奖金可提 10%；利润高于 10 万元低于 20 万元时，低于 10 万元的部分按 10%提成，高于 10 万元的部分可提成 7.5%；20 万元～40 万元时，高于 20 万元的部分，可提成 5%；40 万元～60 万元时高于 40 万元的部分可提成 3%；60 万元～100 万元时，高于 60 万元的部分，可提成 1.5%；高于 100 万元时，超过 100 万元的部分按 1%提成。从键盘输入当月利润 i，求应发放奖金总数。

```
i = int(input('Enter the profit:'))
arr = [1000000, 600000, 400000, 200000, 100000, 0]
rat = [0.01, 0.015, 0.03, 0.05, 0.075, 0.1]
r = 0
for idx in range(0, 6):
    if i > arr[idx]:
        r += (i - arr[idx]) * rat[idx]
        print((i - arr[idx]) * rat[idx])
    i = arr[idx]
print(r)
```

【例 3.17】 一个整数，它加上 100 后是一个完全平方数，再加上 168 又是一个完全平方数，求该数。

```
import math
num = 1
while True:
    if math.sqrt(num +100) -int(math.sqrt(num + 100)) == 0 and math.sqrt
(num + 268)-int(math.sqrt(num + 268)) == 0:
        print(num)
        break
    num += 1
```

【例 3.18】 输入某年、某月、某日，判断这一天是这一年的第几天。

```
#!/usr/bin/python3
date = input("输入年月日(yyyy-mm-dd):")
y, m, d = (int(i) for i in date.split('-'))
sum = 0
special = (1, 3, 5, 7, 8, 10)
for i in range(1, int(m)):
    if i == 2:
        if y % 400 == 0 or (y % 100 != 0 and y % 4 == 0):
            sum += 29
        else:
            sum += 28
    elif (i in special):
        sum += 31
    else:
        sum += 30
sum += d
print("这一天是一年中的第%d天" % sum)
```

【例 3.19】 输出斐波那契数列中的某一项。

斐波那契数列（Fibonacci sequence）又称黄金分割数列，指的是这样一个数列：0、1、1、2、3、5、8、13、21、34……

在数学上，斐波那契数列是以递归的方法来定义的：

```
F0 = 0    (n=0)
F1 = 1    (n=1)
Fn = F[n-1]+ F[n-2](n≥2)
```

程序如下所示。

```
#!/usr/bin/python
# -*- coding: UTF+-8 -*-
a,b = 1,1
for i in range(9):
    a,b = b,a+b
    return a
print(a)          #输出斐波那契数列中第 10 个数
```

【例 3.20】 古典问题：有一对兔子，从出生后第 3 个月起每个月都生一对兔子，小兔子长到第 3 个月后每个月又生一对兔子，假如兔子都不死，求每个月的兔子总数。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
a = 1
b = 1
for i in range(1,21,2):
    print('%d %d'%(a,b))
    a += b
    b += a
```

【例 3.21】 判断 101～200 有多少个素数，并输出所有素数。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
from math import sqrt
def main():
    for i in range(101,201):
        flag = 1
        k = int(sqrt(i))
        for j in range(2,k+1):
            if i%j == 0:
                flag = 0
                break
        if flag == 1:
            print('%5d'%(i))
if __name__ == "__main__":
    main()
```

【例 3.22】 将一个正整数分解质因数。例如，输入 90，打印出 90=2*3*3*5。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
def main():
```

```
    n = int(input('Enter a number:'))
    print(n,'=')
    while(n!=1):
        for i in range(2,n+1):
            if (n%i)==0:
                n//=i
                if(n == 1):
                    print('%d'%(i))
                else:
                    print('%d *'%(i))
                break
if __name__ == "__main__":
    main()
```

【例 3.23】 利用条件运算符的嵌套来完成此题：学习成绩≥90 分的同学用 A 表示，60～89 分的用 B 表示，60 分以下的用 C 表示。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
def main():
    s = int(input('Enter a number:'))
    if s>=90:
        grade = 'A'
    elif s>=60:
        grade = 'B'
    else:
        grade = 'C'
    print(grade)
if __name__ == '__main__':
    main()
```

【例 3.24】 求 s=a+aa+aaa+aaaa+aa…a 的值，其中 a 是一个数字。例如 2+22+222+2222+22222（此时共有 5 个数相加），几个数相加由键盘控制。

```
#!/usr/bin/python
#-*- coding:utf-8 -*-
def main():
    basis = int(input("Input the basis number:"))
    n = int(input("Input the longest length of number:"))
    b = basis
    sum = 0
    for i in range(0,n):
        if i==n-1:
            print("%d "%(basis))
        else:
            print("%d +"%(basis))
        sum+=basis
        basis = basis*10+b
    print('= %d'%(sum))
if __name__ == '__main__':
    main()
```

【例 3.25】 一个数如果恰好等于它的真因子之和，这个数就称为“完数”。例如，6=1＋2＋3。编程找出 1000 以内的所有完数。

```
for i in range (1, 1000):
    sum = 0
    for j in range (1, i):
        if (i%j === 0):
            sum += j
    if sum == i:
        print('YES')
    else:
        print('NO')
```

【例 3.26】 一个球从 100m 高度自由落下，每次落地后反跳回原高度的一半，再落下，求：它在第 10 次落地时，共经过多少米？第 10 次反弹多高？

```
s = 100
h = 50.0
for i in range(2,11):
    s += h
    h /= 2
print("the sum length of path:%f"%s)
print("the last height is:%f"%h)
```

【例 3.27】 猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个，第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃前一天剩下的一半零一个。到第 10 天早上想再吃时，只剩下一个桃子。求第一天共摘了多少个桃子。

```
n = 1
for i in range(9,0,-1):
    n = (n+1)<<1
print(n)
```

【例 3.28】 两个乒乓球队进行比赛，各出 3 人。甲队为 a、b、c 3 人，乙队为 x、y、z 3 人。已抽签决定比赛名单。有人向队员打听比赛的名单。a 说他不和 x 比，c 说他不和 x、z 比，请编程序找出 3 队赛手的名单。

```
for i in range(ord('x'),ord('z') + 1):
    for j in range(ord('x'),ord('z') + 1):
        if i != j:
            for k in range(ord('x'),ord('z') + 1):
                if (i != k) and (j != k):
                    if (i != ord('x')) and (k != ord('x')) and (k !=
ord('z')):
                        print('order is a -- %s\t b -- %s\tc--%s' %
(chr(i),chr(j),chr(k)))
```

【例 3.29】 打印出如下图案（菱形）。

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

```
for i in range(1,8,2):
    print(' '*(4-(i+1)/2)+'*'*i)
for i in range(5,0,-2):
    print(' '* (4-(i+1)/2)+'*'*i)
```


【例 3.30】 有一分数序列为 2/1，3/2，5/3，8/5，13/8，21/13…求出这个数列的前 20 项之和。

```
u = 2.0
d = 1.0
s = 0.0
for i in range(0,20):
    s = s+u/d
    u = u+d
    d = u-d
print('%f'%s)
```

【例 3.31】 求 1+2!+3!+…+20!的和。

```
s = 0
t = 1
for i in range(1,21):
    t*=i
    s+=t
print(s)
```

【例 3.32】 请输入星期的第一个字母来判断一下是星期几，如果第一个字母一样，则继续判断第二个字母。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
letter = input("please input:")
#while letter  != 'Y':
if letter == 'S':
    print ('please input second letter:')
    letter = input("please input:")
    if letter == 'a':
        print ('Saturday')
    elif letter  == 'u':
        print ('Sunday')
    else:
        print ('data error')
elif letter == 'F':
    print ('Friday')
elif letter == 'M':
    print ('Monday')
elif letter == 'T':
    print ('please input second letter')
    letter = input("please input:")
    if letter  == 'u':
        print ('Tuesday')
    elif letter  == 'h':
        print ('Thursday')
    else:
        print ('data error')
elif letter == 'W':
```

```
    print ('Wednesday')
else:
    print ('data error')
```

【例 3.33】 数字比较。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
if __name__ == '__main__':
    i = 10
    j = 20
    if i > j:
        print('%d 大于 %d' % (i,j))
    elif i == j:
        print('%d 等于 %d' % (i,j))
    elif i < j:
        print('%d 小于 %d' % (i,j))
    else:
        print('未知')
```

【例 3.34】 海滩上有一堆桃子，5 只猴子来分。第一只猴子把这堆桃子平均分为 5 份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成 5 份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第 3～5 只猴子都是这样做的，问海滩上原来最少有多少个桃子。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
if __name__ == '__main__':
    i = 0
    j = 1
    x = 0
    while (i < 5) :
        x = 4 * j
        for i in range(0,5) :
            if(x%4 != 0) :
                break
            else:
                i += 1
            x = (x/4) * 5 +1
        j += 1
    print(x)
```

【例 3.35】 809*??=800*??+9*??，其中，??代表的两位数, 809*??为 4 位数，8*??的结果为两位数，9*??的结果为 3 位数。求??代表的两位数，及 809*??后的结果。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
a = 809
for i in range(10,100):
    b = i * a
    if b >= 1000 and b <= 10000 and 8 * i < 100 and 9 * i >= 100 and b==800*i+9*i:
        print(b,' = 800 * ', i, ' + 9 * ', i)
```

【例 3.36】 八进制数转换为十进制数。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
if __name__ == '__main__':
    n = 0
    p = input('input a octal number:\n')
    for i in range(len(p)):
        n = n * 8 + ord(p[i]) - ord('0')
    print(n)
```

【例 3.37】 输入一个奇数，然后判断最少几个 9 除以该数的结果为整数。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
if __name__ == '__main__':
    zi = int(input('输入一个数字:\n'))
    n1 = 1
    c9 = 1
    m9 = 9
    sum = 9
    while n1 != 0:
        if sum % zi == 0:
            n1 = 0
        else:
            m9 *= 10
            sum += m9
            c9 += 1
    print('%d 个 9 可以被 %d 整除 : %d' % (c9,zi,sum))
    r = sum / zi
    print('%d / %d = %d' % (sum,zi,r))
```

【例 3.38】 输出指定范围的阿姆斯特朗数。

如果一个 n 位正整数等于其各位数字的 n 次方之和，则称该数为阿姆斯特朗数。

```
lower = int(input("最小值: "))
upper = int(input("最大值: "))
for num in range(lower,upper + 1):
    #初始化 sum
    sum = 0
    #指数
    n = len(str(num))
    #检测
    temp = num
    while temp > 0:
        digit = temp % 10
        sum += digit ** n
        temp //= 10
    if num == sum:
        print(num)
```

【例 3.39】 时间函数举例。这是一个猜数游戏，判断一个人反应的快慢。

```
#!/usr/bin/python
# -*- coding: UTF-8 -*-
```

```
if __name__ == '__main__':
    import time
    import random
    play_it = input('do you want to play it.(\'y\' or \'n\')')
    while play_it == 'y':
        c = input('input a character:\n')
        i = random.randint(0,2**32) % 100
        print('please input number you guess:\n')
        start = time.clock()
        a = time.time()
        guess = int(input('input your guess:\n'))
        while guess != i:
            if guess > i:
                print('please input a little smaller')
                guess = int(input('input your guess:\n'))
            else:
                print('please input a little bigger')
                guess = int(input('input your guess:\n'))
        end = time.clock()
        b = time.time()
        var = (end - start) / 18.2
        print(var)
        # print('It took you %6.3 seconds' % time.difftime(b,a)))
        if var < 15:
            print('you are very clever!')
        elif var < 25:
            print('you are normal!')
        else:
            print('you are stupid!')
        print('Congradulations')
        print('The number you guess is %d' % i)
        play_it = input('do you want to play it.')
```

【★例 3.40】 在 for 循环体内改变循环变量的值实例。

```
for i in range(3):
   print "original:",i
   i=i+3
   print "new",i
original: 0
new 3
original: 1
new 4
original: 2
new 5
```

3.6 实验与习题

1. 一颗球从 100m 高处落下，假设每次弹起距离为落下高度的一半，第几次落下后弹起高度小于 10m?

2. 打印九九乘法表。