



第5章：PYTHON流程控制

5.1 三种基本控制结构

顺序结构

- 按照书写顺序依次解释执行

选择结构

- 按照一个条件去选择其中一个分支执行

循环结构

- 满足某些条件时反复执行某些操作

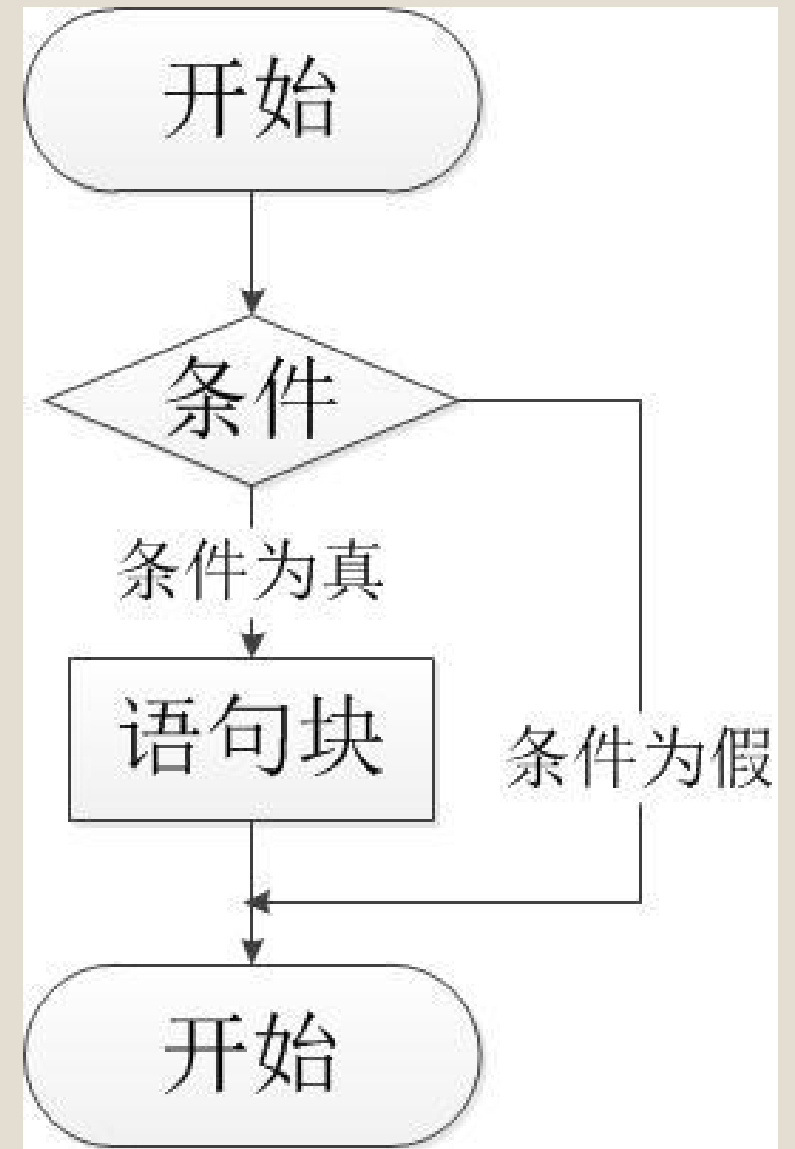
5.2 选择结构

5.2.1: 单选结构

if <条件1>:
 <语句块1>

语法：通过关系运算和逻辑运算的结果决定是否执行语句块，当条件为真则执行语句块，当条件为假则跳过语句块

牢记“冒号”，“缩进”，“对齐”三原则！



例题5.1： 比较大小

要求用户输入两个数字，按照升序将这两个数字输出。

```
a = int(input("Please input the first integer:"))
b = int(input("Please input the second integer:"))
print("before exchange:", a, b)
if a > b:                                #if语句条件
    a, b = b, a                          #if语句块
print("after exchange", a, b)           #if结构外语句，该句一定会执行
```

知识点：语句a,b=b,a可以实现变量a和b数值的交换。

5.2 选择结构

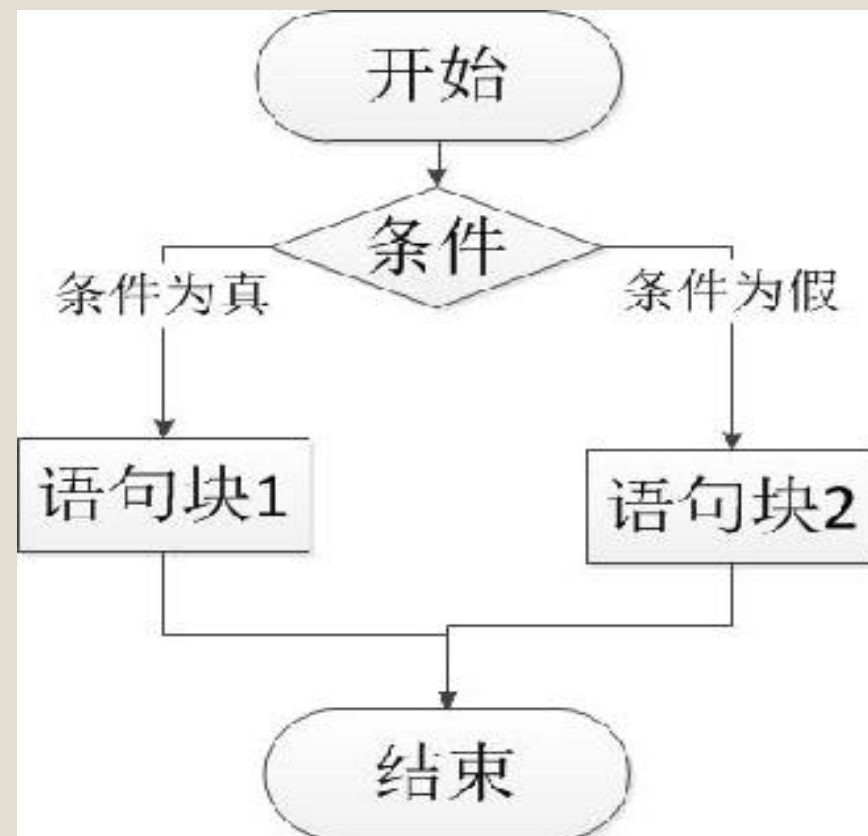
5.2.2: 双选结构

if <条件1>:

 <语句块1>

else:

 <语句块2>



语法：当if 条件满足时执行if后的语句块1，否则执行else后的语句块2

例题5.2：奇偶数判断

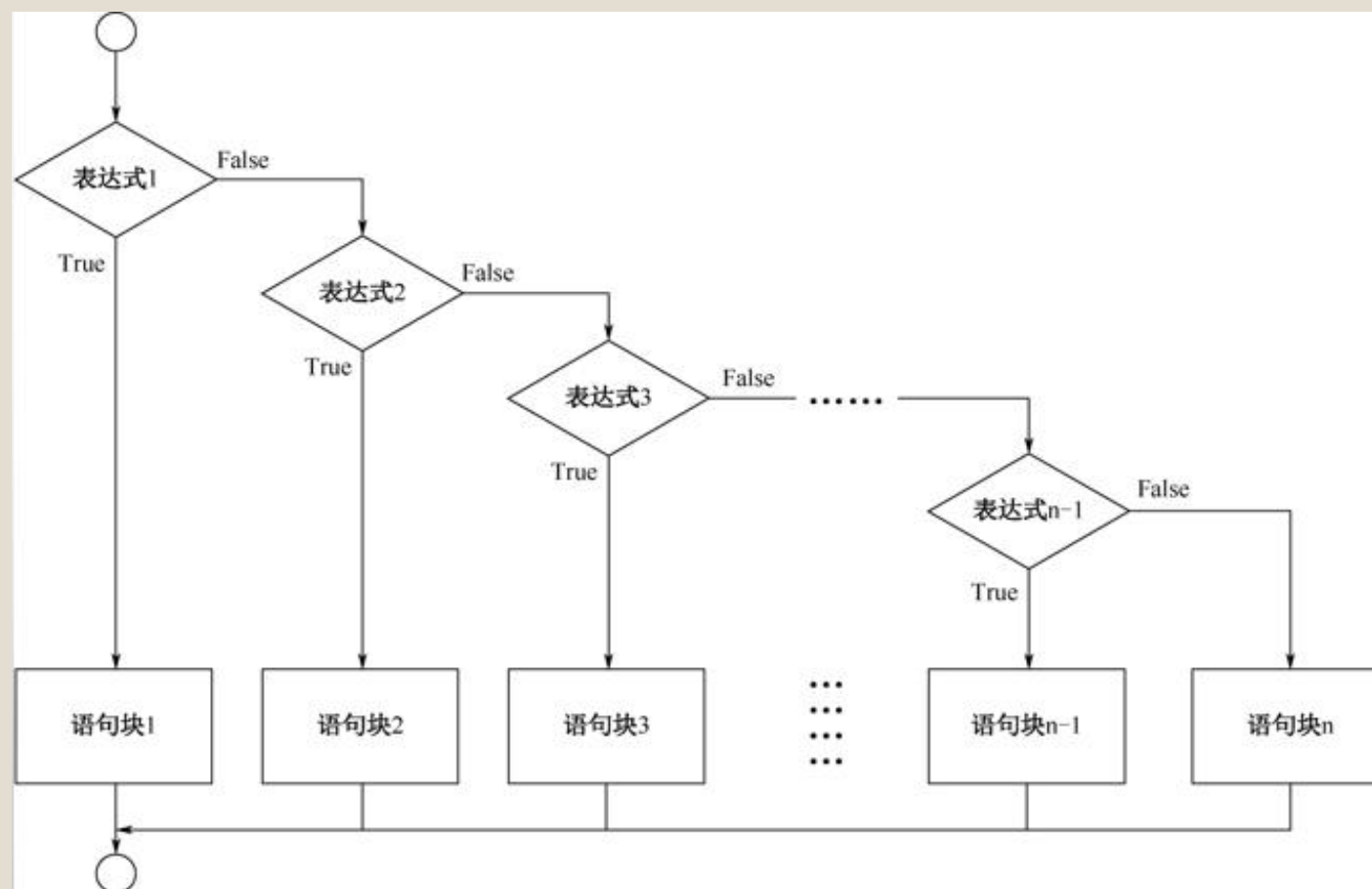
要求用户输入一个数字，判断该数字的奇偶性。

```
x = int(input("Please input an integer:"))
if x % 2 == 0:
    print(x, "is even.")    #如果if条件满足则执行该句
else:
    print(x, "is odd. ")    #如果if条件不满足则执行该句
```

5.2 选择结构

5.2.3: 多选结构

if <条件1>:
 <语句块1>
elif <条件2>:
 <语句块2>
....
else:
 <语句块n>



语法：当条件1为真则执行if后的语句块1，当条件2为真则执行语句块2，....，
否则执行else后的语句块n。

例题5.3：将百分制的成绩转换成绩点

(90~100 : 4, 80~89: 3, 70~79: 2, 60~69:1, 0~59: 0)

```
score = float(input("Please input the score:"))
if score < 0.0 or score > 100.0:    #第一个分支
    gpa = -1
elif score >= 90.0:                #第二个分支
    gpa = 4
elif score >= 80.0:                #第三个分支
    gpa = 3
elif score >= 70.0:                #第四个分支
    gpa = 2
elif score >= 60.0:                #第五个分支
    gpa = 1
else:                              #如果所有条件均不满足，则执行此分支
    gpa = 0
    #另一个if-else结构，用来输出结果
if gpa >= 0:
    print("GPA is", gpa)
else:
    print("invalid score:", score)
```


5.2 选择结构

5.2.3 选择结构的嵌套


在选择结构的语句块中包含另一个选择结构。

if<条件>:

```
if<条件>:  
    <语句1>  
elif<条件>:  
    <语句2>
```

else:

<语句3>



这是一个语句块，可以写任何功能的程序，插入一个选择结构也无可厚非

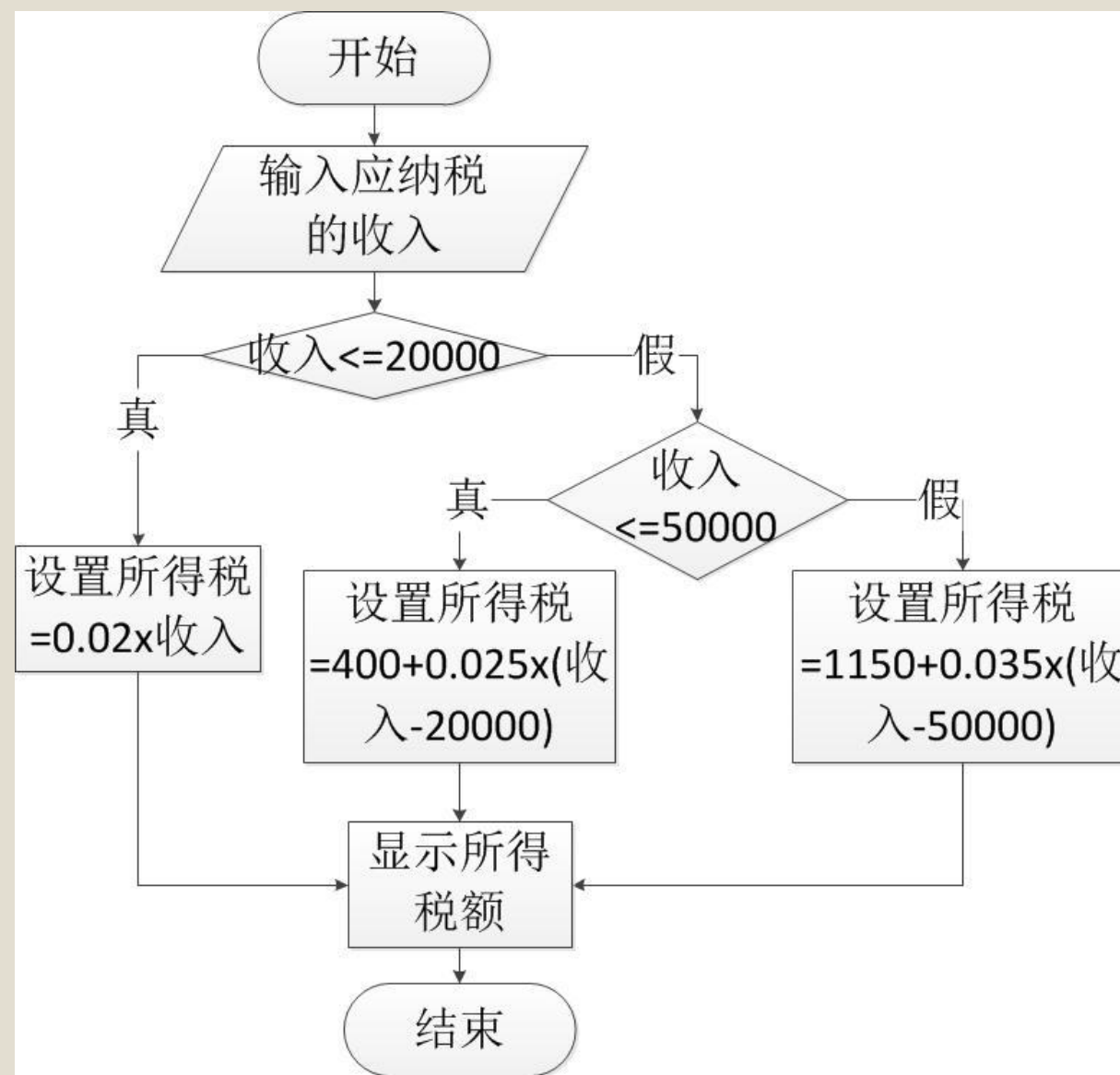
例题5.4： 三角形类型

用户输入三角形的边长，判断能否组成三角形；如果能够组成三角形则输出三角形的面积，并判断三角形满足哪些类型（直角，钝角，锐角三角形）？

```
if a < c:
    a, c = c, a
if a < b:
    a, b = b, a
if b < c:
    b, c = c, b #按abc降序排列三边长

if b + c > a: #如果构成三角形
    s = (a + b + c) / 2.0
    area = (s * (s - a) * (s - b) * (s - c))**0.5
    print("The triangle's area is", area)
    #计算最大内角的余弦值
    cosA = (b * b + c * c - a * a) / 2.0 / b / c
    if cosA > 0.0: #锐角三角形
        print("The triangle is an acute triangle.")
    elif cosA == 0.0: #直角三角形
        print("The triangle is a right triangle.")
    else: #钝角三角形
        print("The triangle is an obtuse triangle.")
else: #如果不构成三角形
    print("Not a triangle.")
```

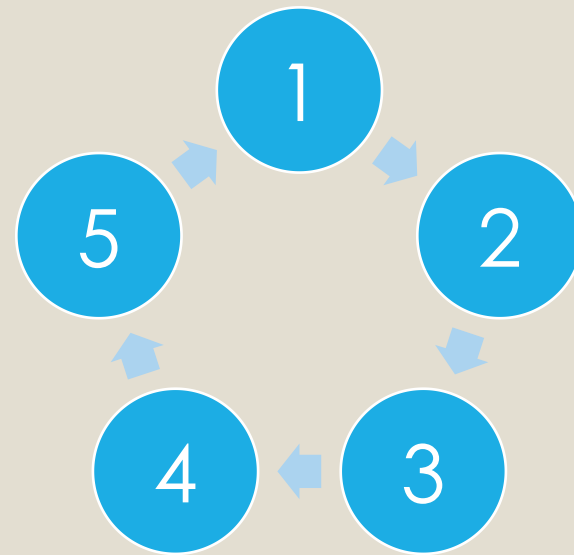
选择结构程序设计随堂练习（5-10分钟）



5.3 循环结构

循环结构是控制一个语句块重复执行的结构。

- while 循环：通过某个条件的真与假来控制循环
- for 循环：将循环内的语句块重复执行特定的次数来控制循环

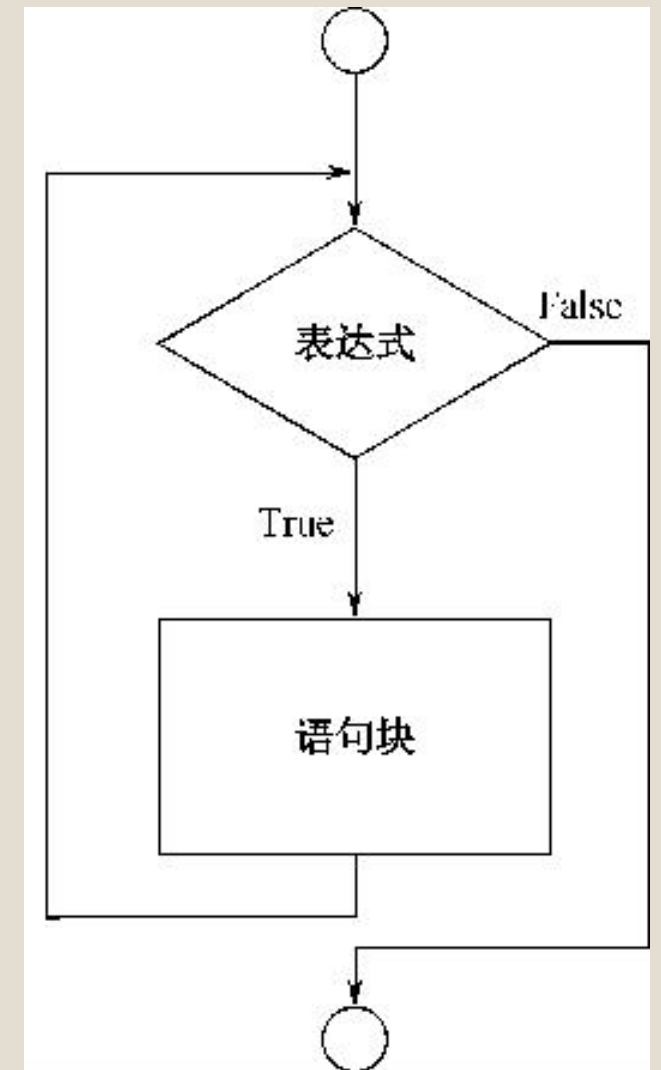


5.3 循环结构

5.3.1: while 循环

- while 循环:
 - while 表达式:
 - 语句块

语法: while 为关键字, 后边的表达式将返回一个布尔值或返回能转换为布尔值的对象。程序首先计算该表达式的值, 如果表达式返回 True, 则执行语句块, 然后程序跳转回 while 语句的第一行重新计算表达式的值, 直到表达式返回 False 时跳出 while 循环, 执行后面的语句。语句体每执行一次被称为这个循环的一次迭代。



例题5.5: while循环示例

1: 打印1~100的所有整数

```
number = 1
while number <= 100:    #循环继续条件
    print(number)
    number += 1
```

2: 计算1+2+3+...+98+99+100

```
total=0
number = 1
while number <= 100:    #循环继续条件
    total+=number
    number += 1

print(total)
```

3.3 循环结构

3.3.1: for 循环

计数器循环:

- 这种使用一个控制变量统计执行次数的循环。

for 循环:

- for 变量 in 序列:
- 语句块

range 函数:
创建范围区间

- for i in range(initialValue, endValue):
- 语句块

range() 函数返回的是一个可迭代对象，可以用list函数将其转换为一个列表，包含从initialValue, 到endValue - 1的整数!

例题5.6: for循环练习

1: 打印1~100的所有整数

```
for number in range(1, 101):    #循环继续条件
    print(number)
```

2: 计算 $1+2+3+\dots+98+99+100$

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

循环结构随堂练习（10分钟）

1

- 计算1~1000范围内3的倍数和7的倍数的数字之和

2

- 水仙花数是一个 3 位数，它的每个位上的数字的 3 次幂之和等于它本身(如: $1^3 + 5^3 + 3^3 = 153$)。求所有水仙花数

5.4 break,continue和pass与嵌套

break:

- 只能出现在while循环或for循环中。当程序执行到break语句时，将跳出整个循环结构而继续执行后面的语句。

continue:

- 只能出现在while循环或for循环中。当程序执行到continue语句时，将立即终止当前迭代，而开始下一次迭代。

pass:

- 什么都不做，保证代码格式的规范和完整性。

循环结构

嵌套的含义是在循环结构中的语句块中仍然有循环。当break语句和continue语句出现在嵌套的循环结构中时，将只作用于最内层循环。

需要注意的是，循环嵌套所进行的指令数量是乘法上升的，这也意味着计算机运行该程序的时间也是乘法上升的。

break示例：求最大公约数

```
a = int(input("输入第1个数a: "))
b = int(input("输入第2个数b: "))

if a < b:
    a, b = b, a

for i in range(b-1, 1, -1):
    if a%i==0 and b%i==0:
        print("%d 和 %d的最大公约数为 %d"%(a, b, i))
        break
else:
    print("%d 和 %d 没有最大公约数"%(a, b))
```

break示例：密码登陆，3次机会。

```
PASSWORD = "12345678"

flag = False
for i in range(1, 4):
    pwd = input("Please input the password:")
    if pwd == PASSWORD:          #如果密码正确
        flag = True             #设置flag变量
        break                   #结束输入密码的循环
    else:
        print("Password is not correct, Try Again.")
if flag:
    print("You just logged in.")
else:
    print("You failed to log in.")
```

continue示例： 1-100间不是7的倍数的数字之和

```
sum = 0
for num in range(1, 101):
    if num % 7 == 0:           #如果数字是7的倍数
        continue             #跳过该数，进行下一次迭代
    sum += num                 #如果该数没有被跳过，则加到总和中
print("The sum is", sum)
```

continue示例： 分析如下代码运行的结果？

```
i=0
total=0
while i<5:
    i = i + 1
    if i == 3:
        continue
    total = total + i
print(total)
```

pass示例：输出字符串中的每个字符，h除外

```
for letter in 'Python':  
    if letter == 'h':  
        pass  
    else:  
        print('当前字母:', letter)  
  
print("Good bye!")
```


5.5 循环与else子句的搭配

在循环未经break打断的情况下，整个循环完成时执行else子句下的语句块

#寻找x除了自身和1之外的最大因子

```
x = int(input("请输入一个数:"))
for i in range(x-1, 1, -1):
    if x%i==0:
        print(i)
        break
else:
    print("%d是质数"%x)
```

for循环替换为while循环后语法仍然成立!

5.6 程序设计综合实训

1: 输出100之内的所有质数，每行10个依次输出！

```
MAX_NUM = 100
count = 0
print("The prime numbers in [1,100] are")
for number in range(2, MAX_NUM):    #外层循环每次迭代为一个待检测的数
    for divisor in range(2, number):
        if number % divisor == 0:
            break                    #能够被1和自身以外的数整除,不是质数, 打断循环
    else:                             #如果没有找到1和自身外的因子, 该数为质数
        count += 1
        print(number, end="\t")      #打印该数
        if count % 10 == 0:          #每十个数换行打印
            print()
```

2: 输出九九乘法表

```
print("Multiplication Table")
for i in range(1, 10): #外层循环，每次迭代输出一行
    for j in range(1, i + 1):
        print(i, "*", j, "=", i * j, end=' \t')
    print()
```

print()函数end关键字参数默认为"\n"，修改为"\t"

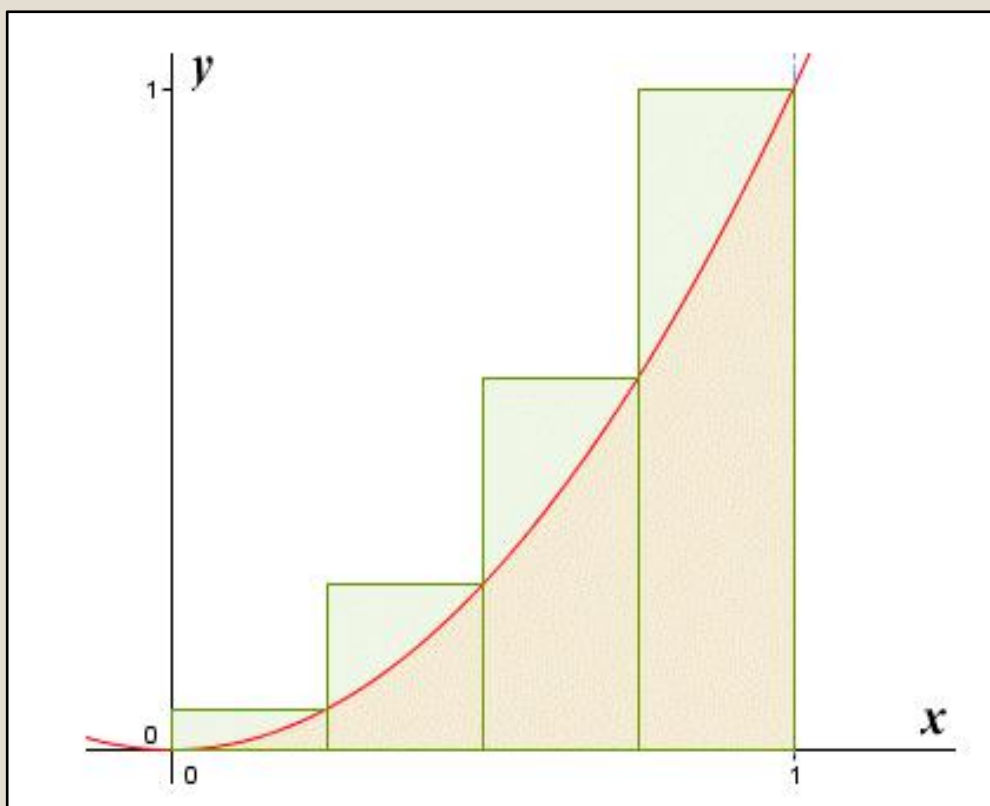
3: 计算圆周率pi的近似值（蒙特卡罗）

```
import math
import random
#输出math模块定义的pi常量
print("PI given by math.PI is", math.pi)

NUMBER_OF_TRIALS = 1000000
hit = 0
for i in range(0, NUMBER_OF_TRIALS):
    x = random.uniform(-1.0, 1.0)
    y = random.uniform(-1.0, 1.0)
    if x * x + y * y <= 1.0:
        hit += 1
print("PI estimated by Monte Carlo simulation is",
      hit/ NUMBER_OF_TRIALS * 4)
```

#每次一个点的随机选取
#随机产生横坐标
#随机产生纵坐标
#如果点在圆内
#增加计数器的值

4: 微积分: 计算 $f(x)=\text{pow}(x,2)$ 从0到1的积分 (微元法)。



```
import math

N = 100000
S = 0
i = 1
while i <= N:
    S += 1/N * pow(i/N, 2)
    i += 1

print(S)
```