Python语言程序设计
Design and Programming of The Python Language

主讲教师：张小东
联系方式：z_xiaodong7134@163.com
答疑地点：宋健研究院514

第3章 控制结构与异常处理

主要内容

- 顺序结构
- 分支控制结构
- 循环程序设计
- 异常处理

➢ 顺序结构

程序按照语句的书写次序自上而下顺序执行

【例3-1】输入圆的半径，计算圆的周长与面积

pi为math中定义的常量

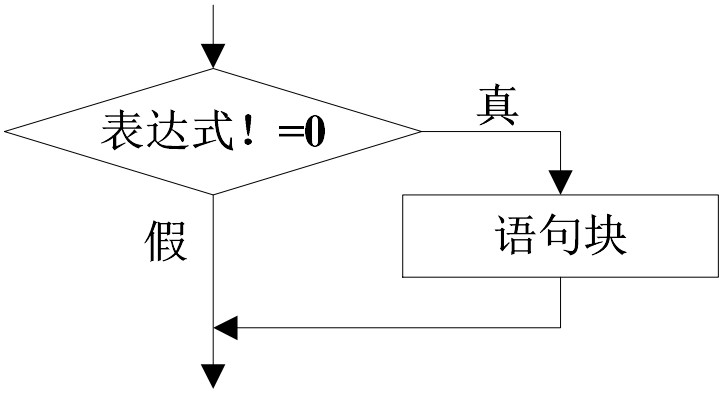
```
from math import *
r=float(input("输入圆半径："))
c=2*pi*r
s=pi*r**2
print("圆周长为：%.2f"%c,";圆的面积为：%.2f"%s)
```

第3章 控制结构与异常处理

主要内容

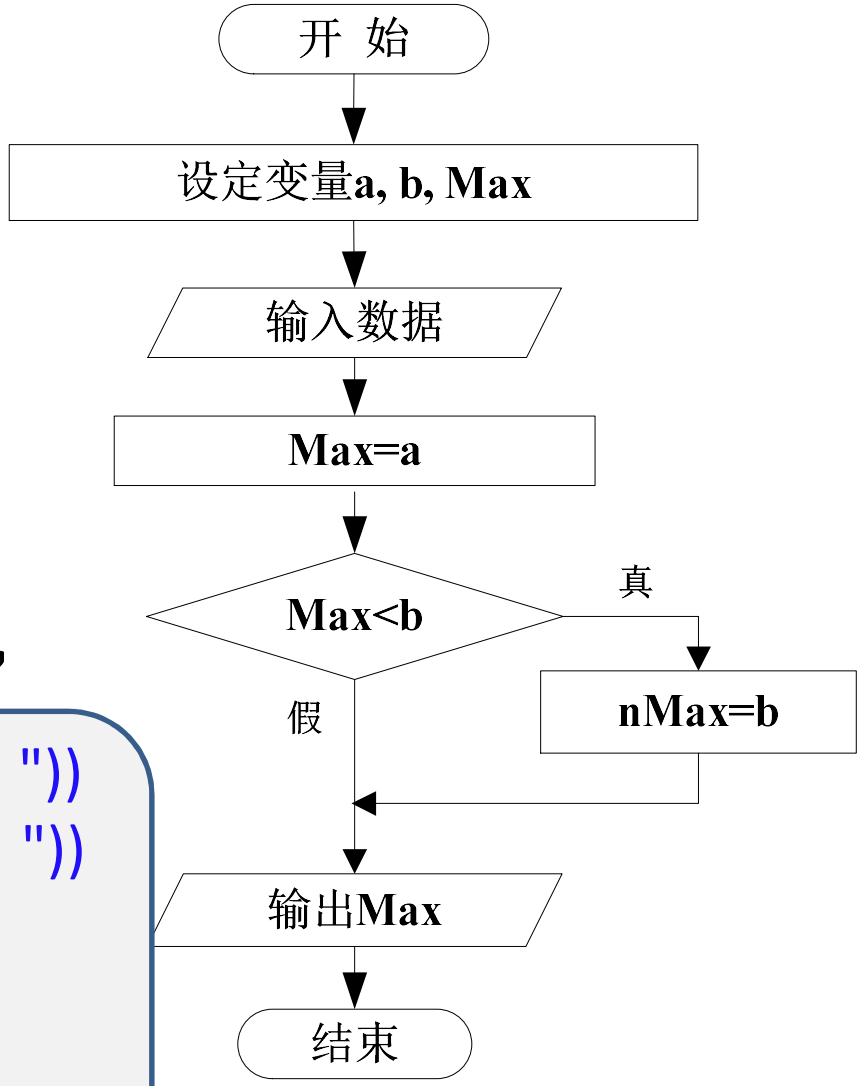
- 顺序结构
- 分支控制结构
- 循环程序设计
- 异常处理

◆一路分支结构

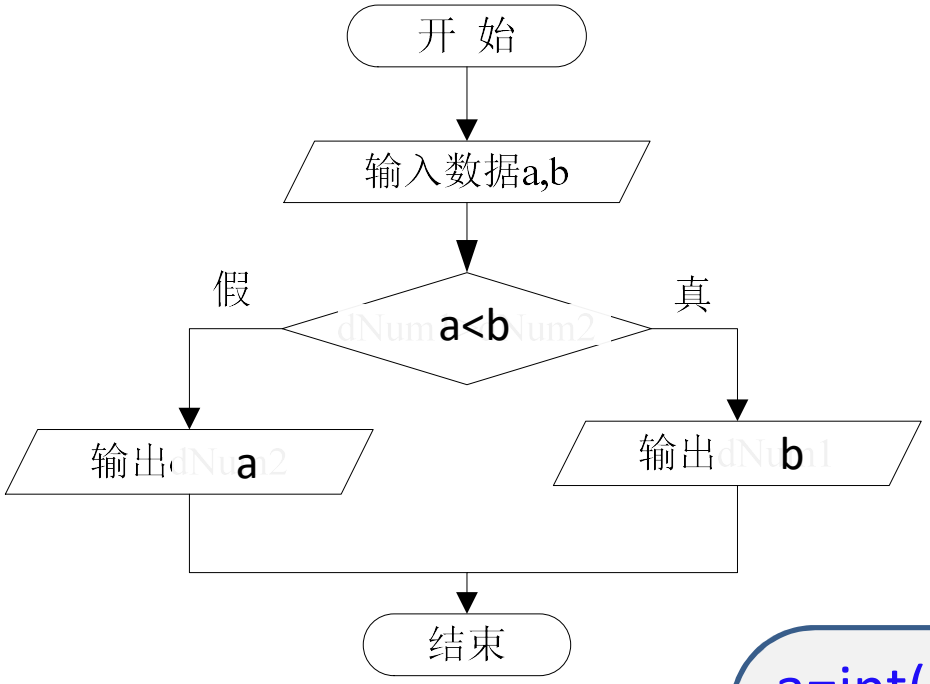


【例3-2】从键盘输入两个数，

```
a=int(input("输入第一个数：  "))
b=int(input("输入第二个数：  "))
Max=a
if Max<b:
    Max=b
print("max=",Max)
```



===分支控制结构===



- 语句格式

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

【例3-3】从键盘输入两个数

```
a=int(input("输入第一个数："))
b=int(input("输入第二个数："))
if a<b:
  print("max=",b)
else:
 print("max=",a)
```

◆ 二路分支结构

【例3-4】编写程序，

解一元二次方程

$a*x^2+bx+c=0$。

用户输入系数a,b,c，

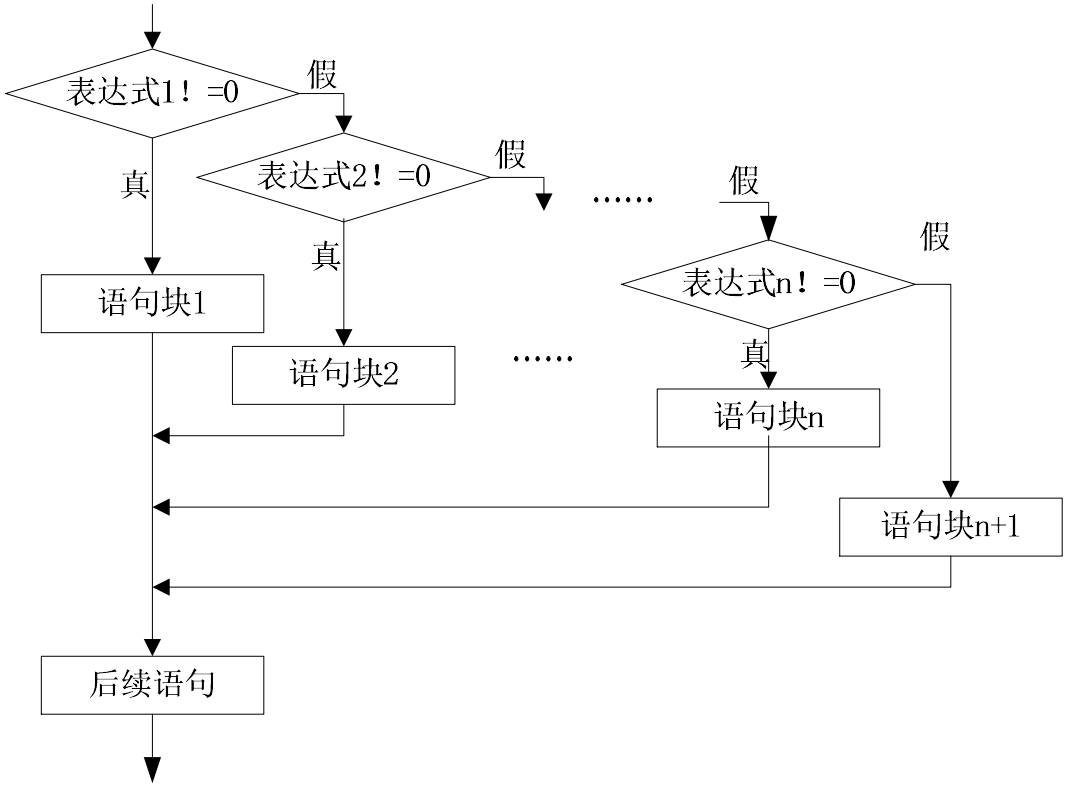
如果有实根计算实根

并显示，如果没有，

显示“没有实根”

```
from math import *
a=float(input('输入a:'))
b=float(input('输入b:'))
c=float(input('输入c:'))
beta=b*b-4*a*c
if beta>=0:
    x1=(-b+sqrt(beta))/(2*a)
    x2=(-b-sqrt(beta))/2/a
    print("x1=",x1,";x2=%.2f"%x2)
else:
    print("没有实根")
```

◆ 多路路分支结构



- 语句格式

```
if<条件1>:
    <语句块1>
elif<条件2>:
    <语句块2>
    ……
elif<条件n>:
    <语句块n>
else:
    <语句块n+1>
```

◆ 多路路分支结构

【例3-5】将百分制转换为5分制。转换规则为：90分及以上者转为5分，80分及以上者转为4分，70分及以上者为3分，60分及以上者2分，不及格为1分

```
a=int(input("请输入百分制成绩:")
b=0
if(a<0 or a>100):
    b=-1
elif(a>=90):
    b=5
elif(a>=80):
    b=4
elif(a>=70):
    b=3
elif(a>=60):
    b=2
else:
    b=1
if(b=-1):
   print('输入错误')
else:
   print(b)
```

必须在这儿定义

◆分支嵌套

【例3-6】对于例3-4，a, b, c可能构成一次方程或者构不成方程，需要判定。

➢ 问题分析

(1)当a=0, b=0则构不成方程，b!=0则构成一次方程，结果为x=-c/b

(2) 当a!=0，计算beta=b*b-4*a*c，若beta>=0，则按正常情况求出方程的两个实根；若beta<0，则deta=sqrt(-beta)

实部：real=-b/(2*a);　　虚部：imag=deta/(2*a)

输出方程有复根:complex(real, imag)和complex(real,-imag)

```
from math import *
a=float(input('输入a:'))
b=float(input('输入b:'))
c=float(input('输入c:'))
if(a==0):
    if(b==0):
        print('输入a=0,b=0不能构成方程！')
    else:
        x=-c/b
        print('输入为一元一次方程，根为：',x)
else:
    beta=b*b-4*a*c
    if(beta>=0):
        x1=-b+sqrt(beta)
        x2=-b-sqrt(beta)
        print('x1=%0.2f'%x1,'x2=%0.2f'%x2)
    else:
        deta=sqrt(-beta)
        real=float('%10.3f'%(-b/(2*a)))
        imag=float('%10.3f'%(deta/(2*a)))
        print('x1=',complex(real,imag),';x2=',complex(real,-imag))
```

◆ 多路路分支结构

【例3-7】编写算法对输入的一个整数，判断它能否被3，5，7整除，并输出以下信息之一：

（1） 能同时被3，5，7整除；

（2） 能被其中两数（要指出哪两个）整除；

（3） 能被其中一个数（要指出哪一个）整除；

（4） 不能被3，5，7任一个整除。

◆ 多路路分支结构

【例3-6】编写算法对输入的一个整数，判断它能否被3，5，7整除。

➢ 算法设计与实现

如何区分被哪几个整除了？

```
n=int(input("Please enter a number:"))
k=(n % 3==0)+(n % 5==0)+(n % 7==0)
if k==3:
    print("All!")
elif k==2:
    print("two!")
elif k==1:
    print("one!")
else:
    print("none!")
```

算法分析：

(1) k的范围是0~3可以表示四种情况。

(2) 题目要求：八种情况！所以k的范围应该是0~7。

◆ 多路路分支结构

【例3-6】编写算法对输入的一个整数，判断它能否被3，5，7整除。

➢ 算法改进：

```
k=(n % 3==0)+(n % 5==0)*2+(n % 7==0)*4
```

```
n=int(input("Please enter a number:"))
k=(n % 3==0)+(n % 5==0)*2+(n % 7==0)*4
if k==7:
    print("All!")
elif k==6:
    print("5 and 7!")
elif k==5:
    print("3 and 7!")
elif k==4:
    print("7!")
elif k==3:
    print("3 and 5!")
elif k==2:
    print("5!")
elif k==1:
    print("3!")
else:
    print("none!")
```

第3章 控制结构与异常处理

主要内容

- 顺序结构
- 分支控制结构
- 循环程序设计
- 异常处理

➢ for循环

◆ 常用格式

for<variable> in range(begin, end, step)：

<循环体>/<语句块>

【例3-7】求1~n之间正整数的平方和。n由用户输入。

➢ 问题分析

$sum=1^2+2^2+3^2+...+n^2$

➢ 计算模型

sum+=i*i　　i∈[1,n]

```
n=int(input('input n:'))
sum=0
for i in range(1,n+1,1):
    sum+=i*i
print('sum=',sum)
```

➢ for循环

◆ 一般格式

for<variable> in <可迭代对象的集合>:

<循环体>/<语句块>

else:

<语句块>

【例3-8】求一组数：23,59,1,20,15,5,3的和及平均值。

```
list1=[23,59,1,20,15,5,3]
k=0
sum1=0
for i in list1:
    sum1+=i
    k+=1
else:
  print('和为：',sum1)
  print('平均值为：',sum1/k)
```

===循环程序设计===

➢ while循环

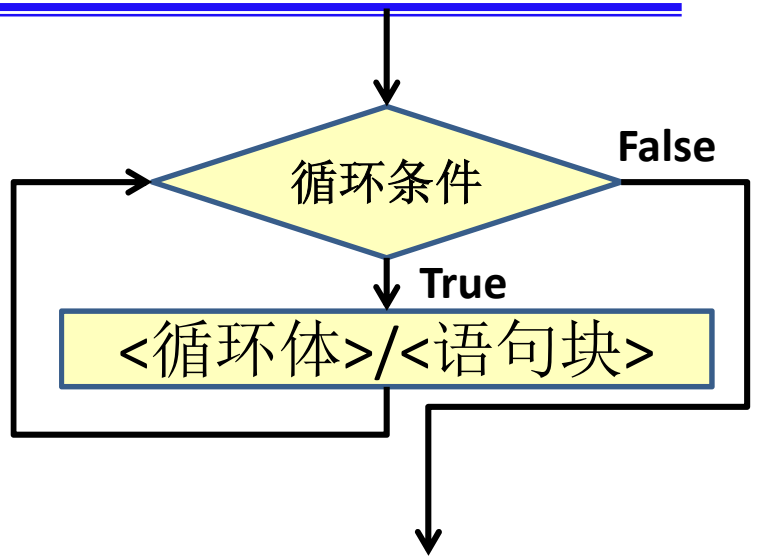
◆ 格式

while<循环条件>:

<循环体>/<语句块>

else:

<语句块>



【例3-9】利用下公式计算e的近似值。要求最后一项的值小于10^{-6}即可

$$e \approx 1+1/1!+1/2!+\ldots+1/n!$$

➢ 计算模型

u=1

ev=1

u=u/i　　i∈[1,n]

ev+=u　1/u>10E-6

```
u=1;ev=1;i=1
while(u>10e-6):
    u=u/i
    ev+=u
    i+=1
print("e=",ev)
```

➢ 循环和分支的嵌套

【例3-10】打印九九表

初始化为字符串

转化为字符串

```
for i in range(1,10):
    a=''
    for j in range(1,i+1):
        a+=str(i)+'*'+str(j)+'='+str(i*j)+' '
    print(a)
```

➢ 循环和分支的嵌套

【例3-11】寻找自幂数。自幂数：对于n位数，它的各位数字的n次方加起来的和仍等于这个数。如$1^3+5^3+3^3=153$，153就是一个三位数自幂数

✓ 计算模型：设n位数为k, digit为n位数某位上的值

1) 找n位数自幂数，k取值空间为$[10^{n-1},10^n-1]$

2) m=k; digit=m%10; total+=pow(digit,n); m=m//10

3) if m==k: print(k)

```
n=int(input("输入位数【1,2,3,4,5,6】:"))
while(0<n<7):
    start=pow(10,n-1)
    end=pow(10,n)
    print(n,"位自幂数为： ")
    for k in range(start,end):
        m=k
        total=0
        while m:
            digit=m%10
            total+=pow(digit,n)
            m//=10
        if(k==total):
            print(k,end='  ')
    n=int(input("\n输入位数【1,2,3,4,5,6】:"))
else:
    print("输入位数不在范围内，程序结束！ ")
```

➢ 循环中的特殊语句pass、break、continue

◆ pass 什么也不做

【例3-12】对指定列表中非2的倍数的数值求和

```
l=[1,2,3,4,5,6,7]
y=0
for item in l:
    if item%2==0:
        pass
    else:
        y+=item
print(y)
```

➢ 循环中的特殊语句pass、break、continue

◆ break 中断循环

◆ continue 结束某轮循环

【例3-13】判断某个数是否为质数

```
n=int(input('输入某个数：'))
flag=1
if n==2:
    print('质数！')
else:
    for i in range(2,n):
        if n%i==0:
            flag=0
            break
    if flag:
        print('质数！')
    else:
        print('非质数！')
```

```
n=int(input('输入某个数：'))
flag=1
if n==2:
    print('质数！')
else:
    for i in range(2,n):
        if n%i:
            continue
        flag=0
        break
    if flag:
        print('质数！')
    else:
        print('非质数！')
```

【例】警察局抓了a，b，c，d四名偷窃嫌疑犯，其中只有一人是小偷。审问中

a说：“我不是小偷。”

b说：“c是小偷。”

c说：“小偷肯定是d。”

d说：“c在冤枉人。”

现在已经知道四个人中三人说的是真话，一人说的是假话，问到底谁是小偷？

问题分析：将a, b, c, d将四个人进行编号，号码分别为1，2，3，4。则问题可用枚举尝试法来解决。

第3章 控制结构与异常处理

主要内容

- 顺序结构
- 分支控制结构
- 循环程序设计
- 异常处理

➢ 异常

异常：程序中产生的错误

后果：如果异常对象未被处理或捕获，程序就会用所谓的回溯(Traceback,一种错误信息)终止执行。

```
>>> num
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    num
NameError: name 'num' is not defined
```

```
>>> 1/0
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    1/0
ZeroDivisionError: division by zero
```

➢ 异常捕捉

◆ 一般格式

```
try:
        <statements1>
except <name1>:              #捕获异常name1
        <statements2>
except <name2, name3>:  #捕获异常name2, name3
        <statements3>
except <name4> as e:       #捕获异常name4, e作为其实例
        <statements4>
except:                          #捕获其它所有异常
        <statements5>
else:                             #无异常
        <statements6>
finally:                          #无论有否异常发生，保证执行
        <statements7>
```

➢ 异常捕捉

◆ 按异常类名捕获异常

【例3-14】输入两整数，打印它们相除之后的结果，若输入的不是整数或除数为0，进行异常处理

```
k=0
while k<4:
    try:
        x=int(input('请输入第一个整数：'))
        y=int(input('请输入第二个整数：'))
        print('x/y=',x/y)
    except ValueError:
        print('请输入一个整数.')
    except ZeroDivisionError:
        print('除数不能为零.')
    k+=1
```

➢ 异常捕捉

◆ 使用异常实例

【例3-15】输入两整数，打印它们相除之后的结果，若输入的不是整数或除数为0，进行异常处理

```
k=0
while k<4:
    try:
        x=int(input('请输入第一个整数：'))
        y=int(input('请输入第二个整数：'))
        print('x/y=',x/y)
    except (ValueError, ZeroDivisionError) as e:
        print(e)
    k+=1
```

➢ 异常捕捉

◆ 捕获所有异常

【例3-16】输入两整数，打印它们相除之后的结果。若有异常则全部捕获。

```
k=0
while k<4:
    try:
        x=int(input('请输入第一个整数：'))
        y=int(input('请输入第二个整数：'))
        print('x/y=',x/y)
    except :                    #Exception as e:
        print('输入错误')  #print(e)
    k+=1
```

➢ 异常捕捉

◆ 自定义异常类　　异常类名称　　继承自exceptoin

✓ 自定义异常

```
class SomeCustomException(Exception):
    pass
```

✓ 自定义异常处理代码可以写在except语句里

◆ 抛出异常类（引发异常）

raise <class>　　#创建并抛出类的实例

raise <instance> #抛出类的实例

===异常处理===

➢ 异常捕捉

◆ 自定义异常类

```
class StrExcept(Exception):
    pass
class MathExcept(Exception):
    pass
```

【例3-17】输入与输出某个人的姓名、年龄、月收入，根据每个项目的约束条件，引发异常。约定名字长度必须在2-20字符之间，年龄在18-60之间，月工资大于800元，否则引发异常。

```
while True:
    try:
        x=input("请输入你的名字(2-20字符):")
        if len(x)<2 or len(x)>20:
            raise StrExcept
        y=int(input("请输入你的年龄(18-60):"))
        if y<18 or y>60:
            raise MathExcept
        z=int(input("请输入你的月收入(大于800):"))
        if z<800:
            raise MathExcept
```

满足条件抛出异常

➢ 异常捕捉

◆ 自定义异常类

【例3-17】输入与输出某个人的姓名、年龄、月收入，根据每个项目的约束条件，引发异常。约定名字长度必须在2-20字符之间，年龄在18-60之间，月工资大于800元，否则引发异常。

```
        print('姓名：',x)
            print('年龄：',y)
            print('年收入：',z*12)
            break
        except StrExcept:
            print('输入名称异常')
        except MathExcept:
            print('输入数值异常')
        except Exception as e:
            print('输入',e)
```

捕捉异常

➢ 异常捕捉

◆ assert语句（断言）

指期望满足用户指定的条件。当用户定义的约束不满足时触发AssertionError异常。它是条件式的raise语句。

✓一般形式

assert <test>, <data> # <data> 可选

逻辑表达式

<test>为假时的提示信息

✓等效代码

```
if not <test>:
    raise AssertionError(<data>)
```

Assert语句是用来收集用户定义的约束条件，而不是捕捉内在的程序设计错误

===异常处理===

➢ 异常捕捉

【例3-18】求x与y的最大公约数，使用assert语句来约束x、y取值为大于1正整数。

```
while True:
    try:
      x=int(input('请输入第一个数：'))
      y=int(input('请输入第二个数：'))
      assert x>1 and y>1,'x与y必须大于1'
      a=x
      b=y
      if a<b:
        a,b=b,a
        while b!=0:
            temp=a%b
            a=b
            b=temp
        else:
          print('%s和%s的最大公约数为：'
                    '%s'%(x,y,a))
          break
except Exception as e:
      print('捕捉到异常',e)
```

本章小结

- 顺序结构
- 分支控制
- 循环程序设计
- 异常处理