Python及其应用

主讲人：钱惠敏
E-mail: amandaqian@hhu.edu.cn

第4讲 程序控制结构

4.1 程序基本机构

4.2 相关基础知识

4.3 选择结构

4.4 循环结构

4.5 其它语句

4.1 程序基本结构

➢程序流程图

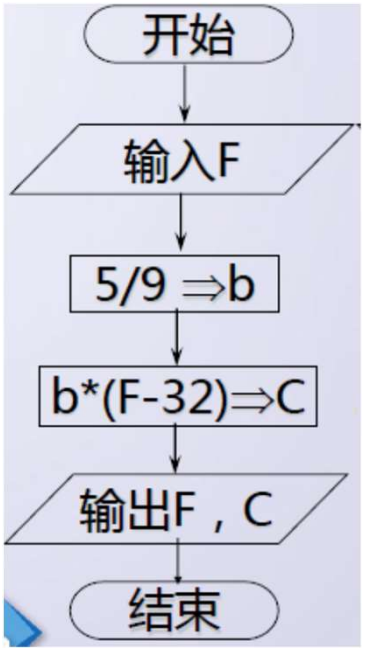
用规定的一系列图形、流程线和文字说明算法中的基本操作和控制流程。

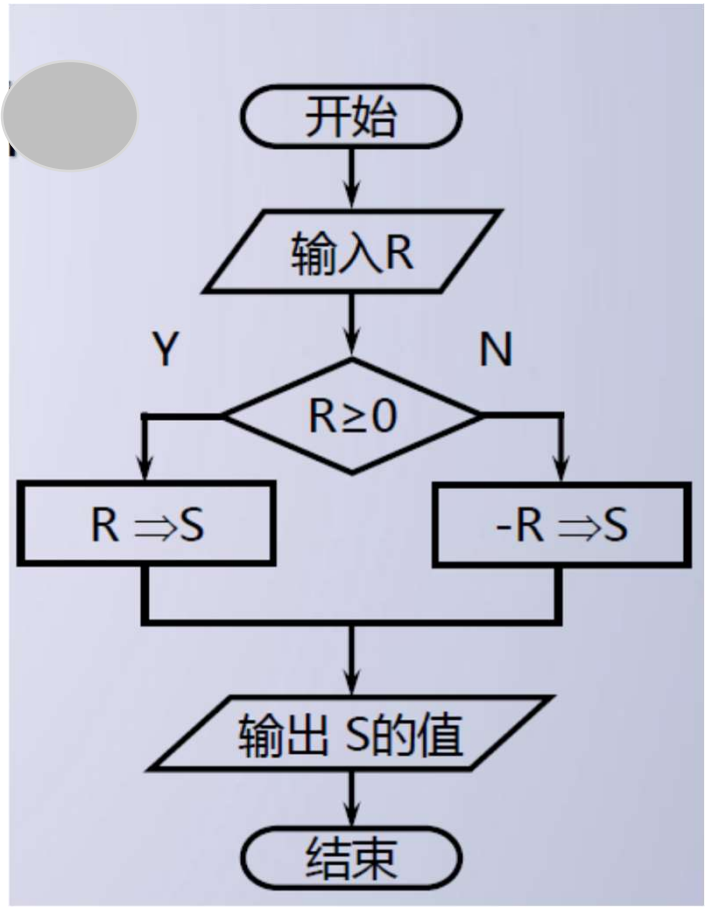
➢流程图的基本元素

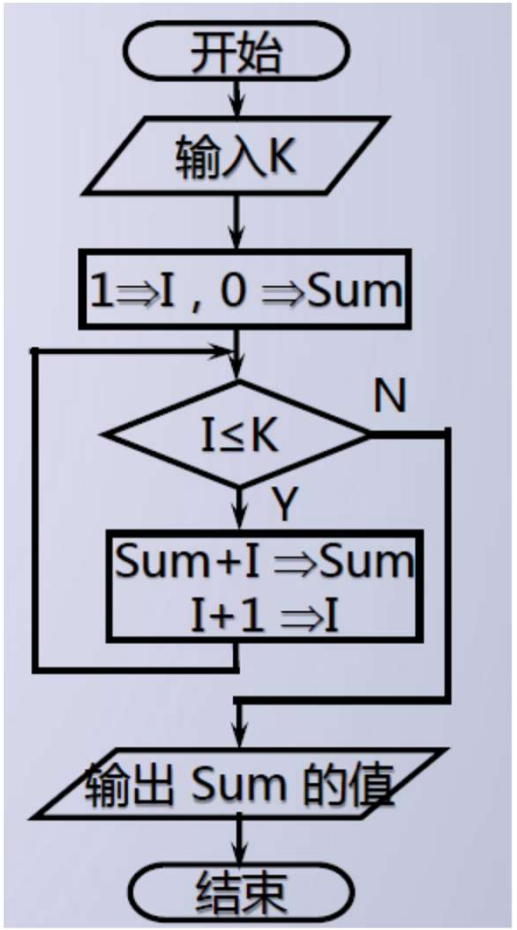
- ✓ 表示相应操作的框；
- ✓ 带箭头的流程线；
- ✓ 框内外必要的文字说明。

4.1 程序基本结构（续1）

➢程序流程图







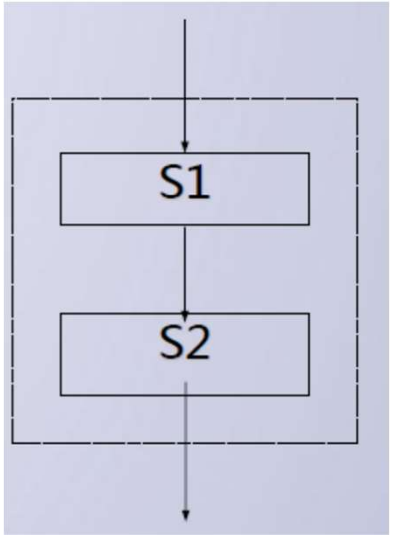
4.1 程序基本结构（续2）

- 程序设计的基本结构
  - ✓ 顺序结构
  - ✓ 选择结构
  - ✓ 循环结构

4.1 程序基本结构（续3）

➢程序设计的基本结构

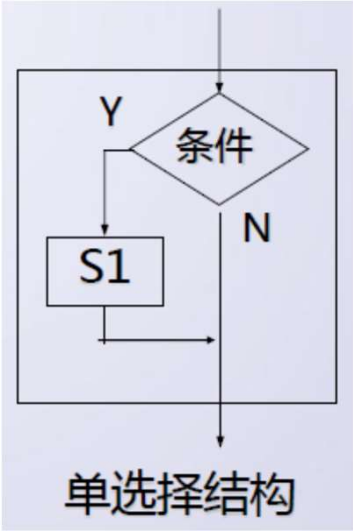
✓ 顺序结构：按语句的自然顺序依次执行



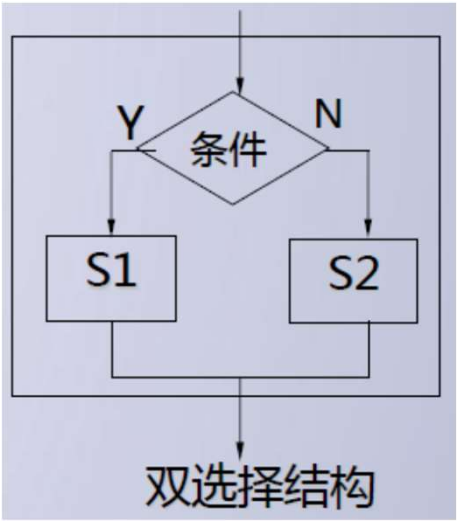
4.1 程序基本结构（续4）

➢程序设计的基本结构

✓ 选择结构



单选择结构

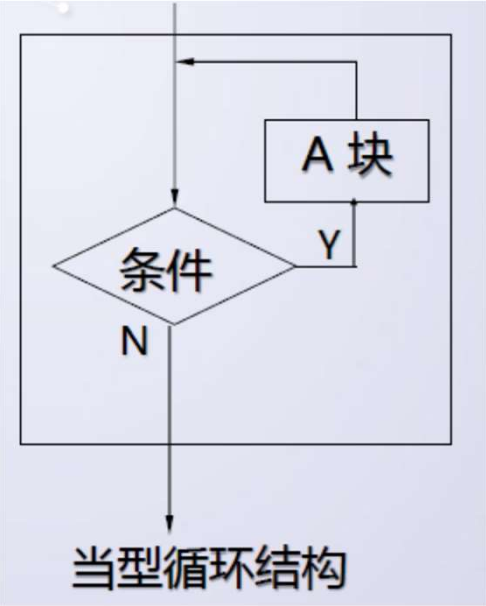


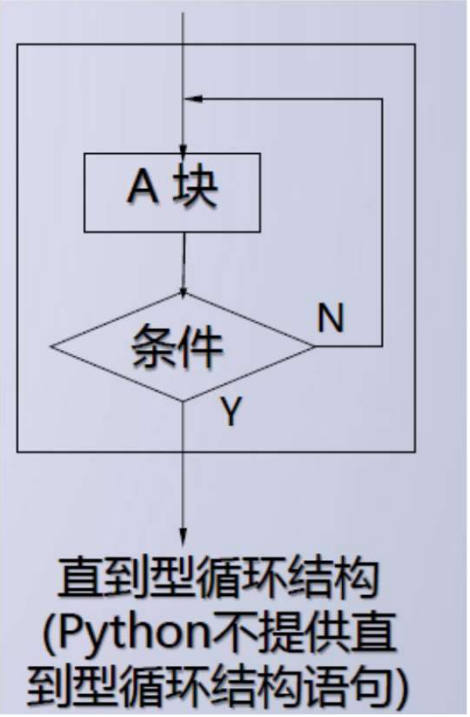
双选择结构

4.1 程序基本结构（续5）

➢程序设计的基本结构

✓ 循环结构





4.2 相关基础知识—import

➢import语句

```
>>> import math  #从Python标准库中引入math.py模块
>>> r=5
>>> print('半径为5的圆的面积为：%.2f' %(math.pi*r**2))
半径为5的圆的面积为：78.54
import math as m
```

➢import的标准语法：

```
import module1[, module2[,... moduleN]]
```

允许一个import导入多个模块，但各个模块间需要用逗号隔开。

```
import math, random
import math as m, random as r
```

4.2 相关基础知识—import （续1）

➢from 模块名 import 成员名 as 别名

from math import sqrt

from math import sqrt as st

from math import degrees, randians

from math import degrees as d, randians as r

from module import *

注意：from module1, module2 import *要慎用，以防module1与module2中有同名函数，此时可为其中一个模块取别名

4.2 相关基础知识—math库

➢math库

常用函数：

函数	数学表示	含义
圆周率pi	π	π的近似值，15位小数
自然常数e	e	e的近似值，15位小数
ceil(x)	⌈x⌉	对浮点数向上取整
floor(x)	⌊x⌋	对浮点数向下取整
pow(x,y)	x^y	计算x的y次方
log(x)	lg x	以e为基的对数，
log10(x)	$\log_{10}x$	以10为基的对数，
sqrt(x)	$\sqrt{x}$	平方根

函数	数学表示	含义
exp(x)		e的x次幂，
degrees(x)		将弧度值转换成角度
radians(x)		将角度值转换成弧度
sin(x)	sin x	正弦函数
cos(x)	cos x	余弦函数
tan(x)	tan x	正切函数
asin(x)	arcsin x	反正弦函数，x∈[-1.0,1.0]
acos(x)	arccos x	反余弦函数，x∈[-1.0,1.0]
atan(x)	arctan x	反正切函数，x∈[-1.0,1.0]

4.2 相关基础知识—random库

➢random库

常用函数：

函数	含义
seed(x)	给随机数一个种子值，默认随机种子是系统时钟
random()	生成一个[0, 1.0)之间的随机小数
uniform(a,b)	生成一个a到b之间的随机小数
randint(a,b)	生成一个a到b之间的随机整数
randrange(a,b,c)	随机生成一个从a开始到b以c递增的数
choice(<list>)	从列表中随机返回一个元素
shuffle(<list>)	将列表中元素随机打乱
sample(<list>,k)	从指定列表随机获取k个元素

4.2 相关基础知识—别样的赋值

➢序列解包（可选迭代解包）

```
>>> x,y,z=1,2,3
>>> x,y=y,x
>>> print(x,y,z)
2 1 3
```

x和y的值交换了

序列解包对不同变量赋不同值时非常有用！

```
>>> student={'name':'小智','number':'1001'}
>>> key,value=student.popitem()
>>> print(key)
>>> print(value)
number
1001
```

4.2 相关基础知识—别样的赋值 （续1）

➢链式赋值

```
>>> x=y=z=10
>>> print(x,y,z)
10 10 10
```

链式赋值是将同一个值赋给多个变量的

➢增量赋值（赋值运算符）

```
>>> x=5
>>> x+=1   #加
>>> x
6
```

```
>>> target='Hello’
>>> target+='world’
>>> target
'Helloworld’
>>> target*=2
>>> target
'HelloworldHelloworld'
```

4.2 相关基础知识—语 句 块

- 语句块是在满足一定条件下执行一次或多次的一组语句。
- 同一段语句块中的每行都要保持同样的缩进。
- 冒号（:）用来标识语句块的开始，块中的每一个语句都是缩进的（缩进量相同）。当退回到和已经闭合的块一样的缩进量时，就表示当前块已经结束了。

```
def fib(n): # 定义到 n 的斐波那契数列
    a, b = 0, 1
    while b < n:
        print(b, end=' ')
        a, b = b, a+b
    print()
```

4.3 选择结构—条件语句

➢布尔变量：True、False

哪些值会被解释器看作假（false）？

False None 0 “” () [] {}

➢if语句

```
#if 基本用法
greeting='hello'
if greeting == 'hello':
    print('hello')
```

4.3 选择结构—条件语句(续1)

➢else子句

```
greeting='hi'
if greeting == 'hello':
    print('hello')
else:
    print('该语句块不在if中，greeting的值不是hello')
```

4.3 选择结构—条件语句(续2)

➢elif子句(=elseif)

```
num = 10
if num > 10:
    print('num的值大于10')
elif 0<=num<=10:
    print('num的值介于0到10之间')
else:
    print('num的值小于0')
```

4.3 选择结构—条件语句(续3)

➢嵌套代码：把if、else、elif这些条件语句再放入到if、else、elif这些条件的语句块中，作为更深一层次的条件判定语句。

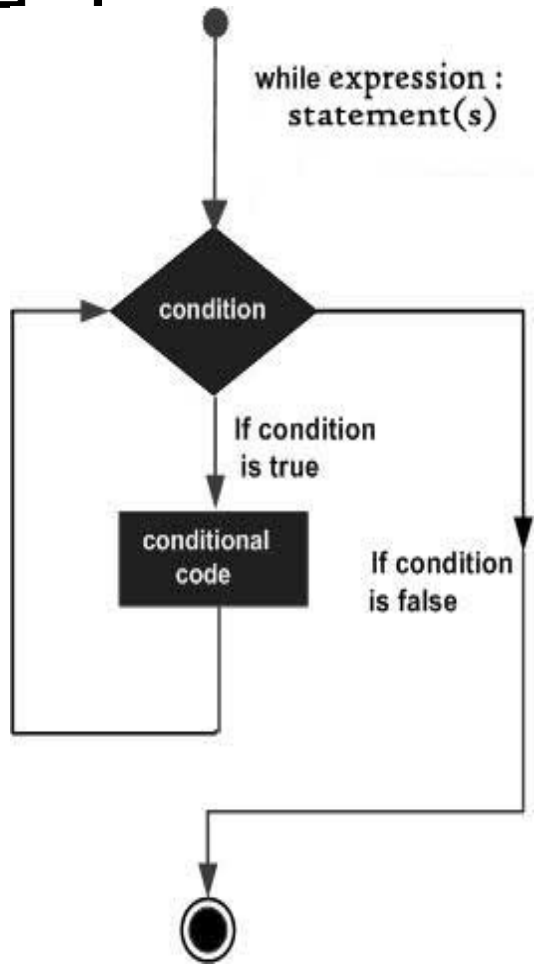
```
num = 10
if num%2==0:
    if num%3==0:
        print ("你输入的数字可以整除 2 和 3")
    elif num%4==0:
        print ("你输入的数字可以整除 2 和 4")
    else:
        print ("你输入的数字可以整除 2，但不能整除 3 和 4")
else:
    if num%3==0:
        print ("你输入的数字可以整除 3，但不能整除 2")
    else:
        print  ("你输入的数字不能整除 2 和 3")
```

4.4 循环结构—while循环

➢while循环

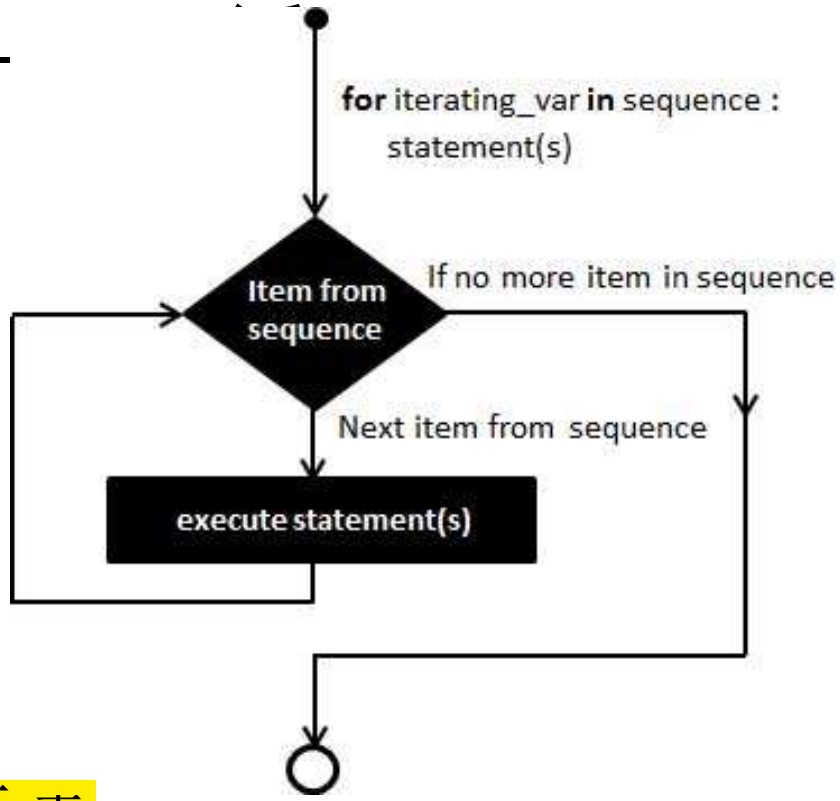
```
n=1
while n<=100:
    print('当前数字是：',n)
    n += 1
```

while 判断条件：
    执行语句……



➢执行语句可以是单个语句或语句块。判断条件可以是任何表达式，任何非零、或非空（null）的值均为真（true）。当判断条件为假（false）时，循环结束。

4.4 循环结构-



➢for循环

```
for iterating_var in sequence:
    statements(s)
```

➢sequence是任意序列。

iterating_var是需要遍历的元素。

statements是待执行的语句块。

```
fields=['a','b','c']
for f in fields:
    print('当前字母是：',f)
```

4.4 循环结构—for循环

➢for循环遍历字典元素

```
tups={'name':'小智','number':'1002'}
for tup in tups:   #for循环字典
    print('%s:%s' % (tup,tups[tup]))
name:小智
number:1002
```

```
tups={'name':'小智','number':'1002'}
for key,value in tups.items():
    print('%s:%s' % (key,value))
name:小智
number:1002
```

序列解包

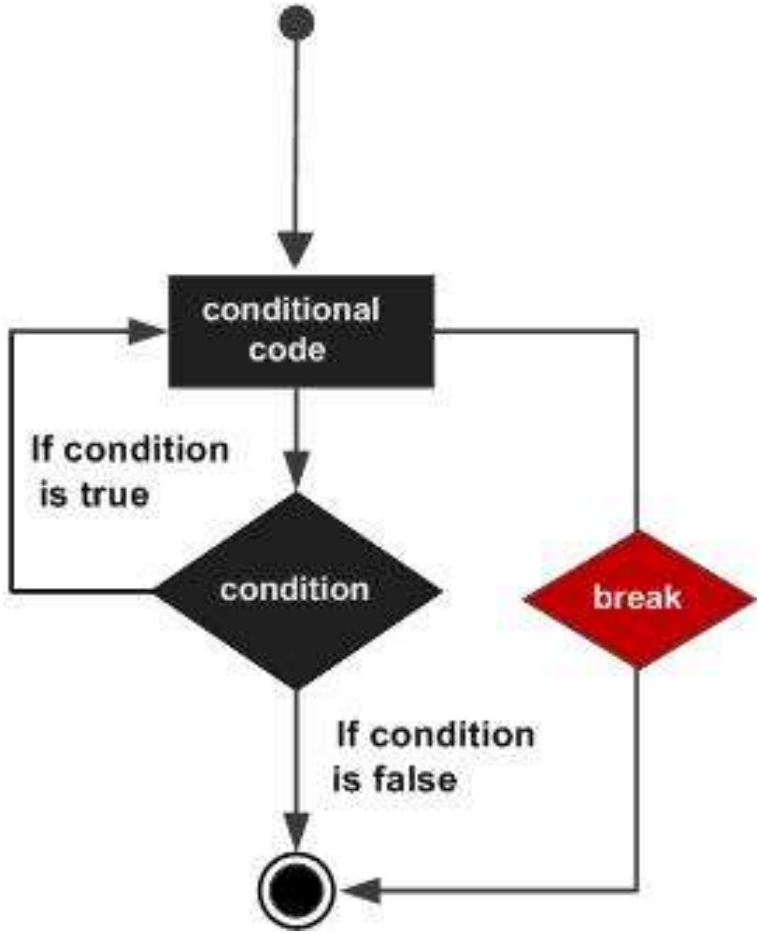
遍历（键，值）元组的方法

4.4 循环结构—跳出循环

➢break语句

在while和for循环中，break语句用来终止循环。如果使用了嵌套循环，break语句将停止执行最深层的循环，并开始执行下一行代码。

break

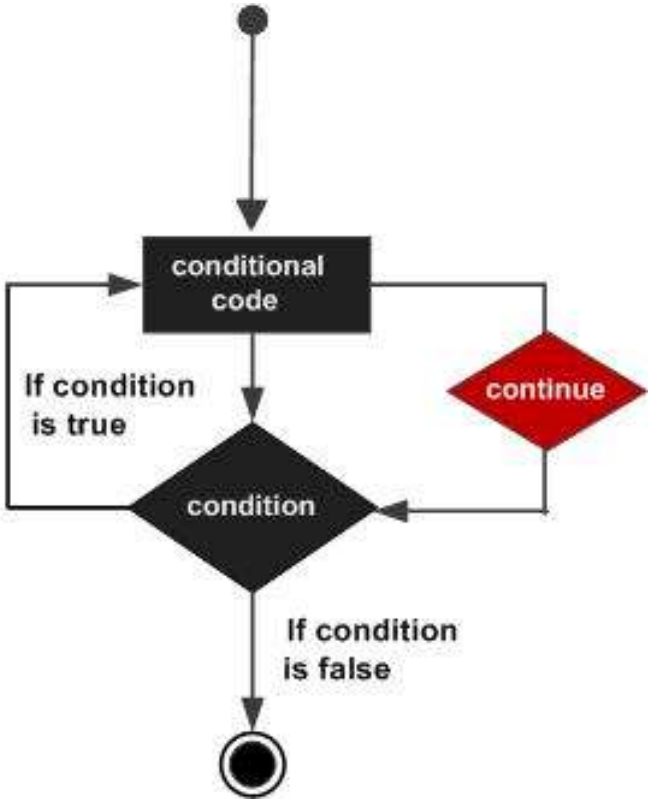


4.4 循环结构—跳出循环（续1）

➢continue

在while和for循环中，用于跳过当前循环的剩余语句，然后继续进行下一轮循环。

continue



4.4 循环结构—循环中的else子句

➢while循环使用else语句

在while条件语句为false时执行else的语句块

```
num = 0
while num < 3:
   print (num, " 小于 3")
   num = num + 1
else:
   print (num, " 大于或等于 3")
print("结束循环!")
```

4.4 循环结构—循环中的else子句（续1）

➢for循环使用else语句

在for条件语句为false或结束后没有被break中断时，执行else的语句块

```
names = ['xiaomeng', 'xiaozhi']
for name in names:
    if name == "xiao":
        print("名称：",name)
        break
    print("循环名称列表 " + name)
else:
    print("没有循环数据!")
print("结束循环!")
```

4.5 其他语句

➢is：同一性运算符

```
>>> x=y=[1,2,3]
>>> z=[1,2,3]
>>> x==y
True
>>> x==z
True
>>> x is y
True
>>> x is z
False
```

is运算符是判定同一性而不是相等性的。变量x和y都被绑定到同一个列表上，而变量z被绑定在另外一个具有相同数值和顺序的列表上。它们的值可能相等，但是却不是同一个对象。

从内存的角度思考， x与y所指向的内存空间是不一样的，x和y指向同一块内存空间，z指向另一块内存空间。

4.5 其他语句（续1）

➢序列比较和字符串比较

```
>>> [1,2]<[2,1]
True
>>> [1,2]<[1,2]
False
>>> [1,2]==[1,2]
True
>>> [2,[1,2]]<[2,[1,3]]
True

>>> ‘abc’<‘bcd’
True
>>> ‘abc’==‘abc’
True
```

4.5 其他语句（续2）

- pass语句： pass不做任何事情，一般用做占位语句，作用是保持程序结构的完整性

```
pass
```

- 排序和翻转

```
>>> sorted([5,3,7,1])
[1, 3, 5, 7]
>>> sorted('hello,world!')
['!', ',', 'd', 'e', 'h', 'l', 'l', 'l', 'o', 'o', 'r', 'w']

>>> list(reversed('hello,world!'))
['!', 'd', 'l', 'r', 'o', 'w', ',', 'o', 'l', 'l', 'e', 'h']
>>> ''.join(reversed('hello,world!'))
'!dlrow,olleh'
```

字符串连接