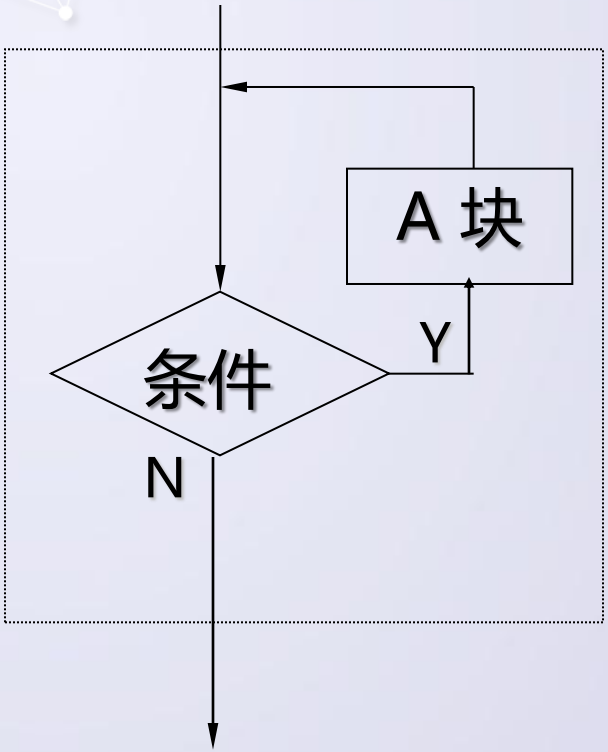
基本循环结构

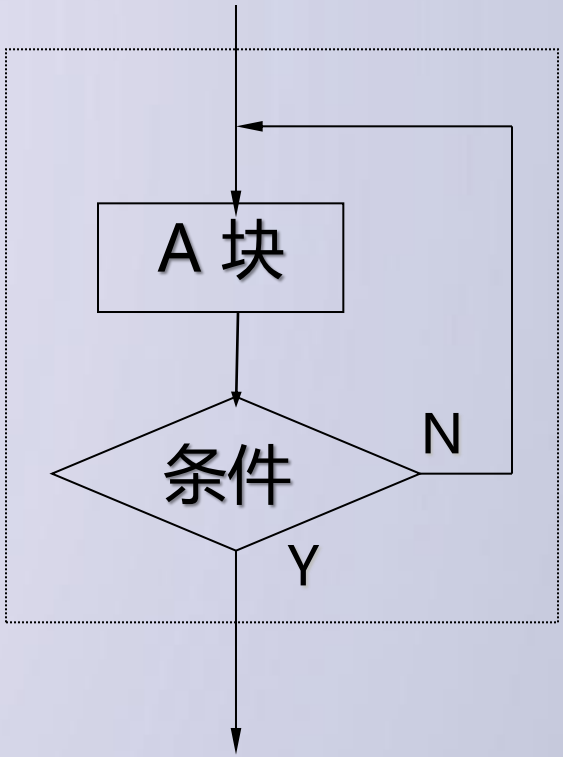
礼 欣

北京理工大学

循环结构回顾



当型循环结构



直到型循环结构
(Python不提供直到型循环结构语句)

python 语言程序设计

三种基本结构的特点： 一个入口，一个出口，不出现死循环和死语句

for循环

- Python可以使用for语句循环遍历整个序列的值

  for <var> in <sequence>:

  <body>

- 在for循环中，循环变量var遍历了队列中的每一个值，循环的语句体为每个值执行一次。

```
Type "copyright", "credits" or "license()" for more information.
>>> # Measure some strings:
>>> words = ['cat', 'window', 'defensestrate']
>>> for w in words:
        print(w, len(w))


cat 3
window 6
defensestrate 13
>>>
```

python 语言程序设计

for 循环

- 注意，for循环在执行过程中，直接在序列上进行遍历，而非在内存中生成一个新的序列拷贝进行

```
>>> for w in words[:]: #Loop over a slice copy of the entire list.
        if len(w) > 6:
                words.insert(0, w)


>>> words
['defensestrate', 'cat', 'window', 'defensestrate']
```

python 语言程序设计

for循环-求平均数

- 平均数计算程序的IPO如下:

  输入：待输入数字个数，数字

  处理：平均数算法

  输出：平均数

- 通用设计方案：

  输入数字的个数n

  将sum初始化为0

  循环n次：

  　　输入数字x

  　　将x加入sum中

  将sum/n作为平均数输出出来

for循环-代码

```
# average1.py
n = eval(input("How many numbers? "))
sum = 0.0
for i in range(n):
    x = eval(input("Enter a number >> "))
    sum = sum + x
print("\nThe average is", sum / n)
```

python 语言程序设计

for循环-执行

- 以下是程序的执行结果

```
>>>
How many numbers? 5
Enter a number >> 32
Enter a number >> 45
Enter a number >> 34
Enter a number >> 76
Enter a number >> 45

The average is 46.4
>>>
```

python 语言程序设计

for循环-缺点

- 程序开始时必须提供输入数字总数
- 大规模数字求平均值需要用户先数清楚个数
- for循环是需要提供固定循环次数的循环方式
- Python提供了另一种循环模式即无限循环，不需要提前知道循环次数，即我们提到的当型循环也叫条件循环

无限循环

- 语法：while语句

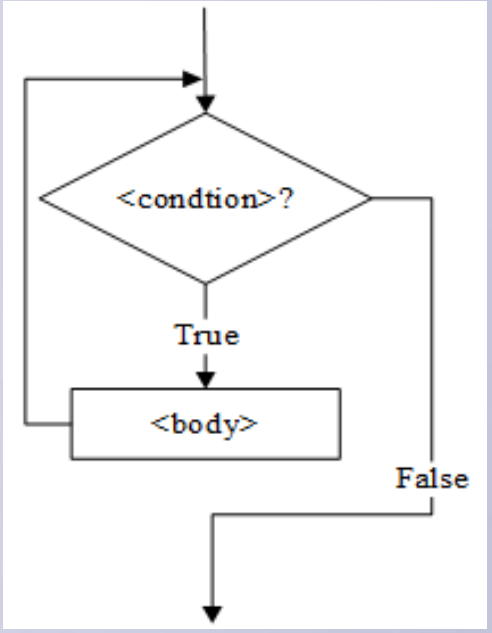
  while <condition>:

  <body>
- while语句中<condition>是布尔表达式
- <body>循环体是一条或多条语句
  - 当条件<condition>为真时，循环体重复执行
  - 当条件<condition>为假时，循环终止
- 在while循环中，条件总是在循环顶部被判断，即在循环体执行之前，这种结构又被称为前测循环

无限循环

- 下面是使用while循环完成从0到10的求和打印平均值的例子：

```
i = 0
while i <= 10:
    print(i)
    i = i + 1
```



- 如果循环体忘记累加i，条件判断一直为真，循环体将一直执行，这就是所谓的死循环程序
- 这时通常使用<Ctrl>-c来终止一个程序

python 语言程序设计

for/while 中的else, break用法

- Break 语句- 跳出最内层for/while 循环

```
#TestBreak.py
sum = 0
number = 0
while number < 20:
    number += 1
    sum += number
    if sum > 100:
        break
print("The number is", number)
print("The sum is", sum)
```

```
>>>
The number is 14
The sum is 105
>>>
```

python 语言程序设计

for/while 中的continue用法

- continue语句， 其作用为结束本次循环。即跳出循环体中下面尚未执行的语句，对于while循环，继续求解循环条件。而对于for循环程序流程接着遍历循环列表
- continue语句和break语句的区别是：
- continue语句只结束本次循环，而不终止整个循环的执行。而break语句则是结束整个循环过程，不再判断执行循环的条件是否成立：

```
for num in range(2, 10):
    if num % 2 == 0:
        print("Found an even number", num)
        continue
    print("Found a number", num)
```

```
>>>
Found an even number 2
Found a number 3
Found an even number 4
Found a number 5
Found an even number 6
Found a number 7
Found an even number 8
Found a number 9
>>>
```

python 语言程序设计

for/while 中的else用法

- Break 语句- 跳出最内层for/while 循环
- <for… else: …> <while… else: …>语句与循环的搭配使用，else:后的表达式在for循环列表遍历完毕后或while 条件语句不满足的情况下执行，例如：

```
for n in range(2, 10):
    for x in range(2, n):
        if n % x == 0:
            print(n, 'equals', x, '*', n // x)
            break
    else:
        # loop fell through without finding a factor
        print(n, 'is a prime number')
```

```
>>>
2 is a prime number
3 is a prime number
4 equals 2 * 2
5 is a prime number
6 equals 2 * 3
7 is a prime number
8 equals 2 * 4
9 equals 3 * 3
>>>
```

python 语言程序设计