



《Python程序设计基础教程（微课版）》

<http://dblabb.xmu.edu.cn/post/python>



第3章 程序控制结构

林子雨 博士/副教授

厦门大学计算机科学系

E-mail: ziyulin@xmu.edu.cn ▶▶

主页: <http://dblabb.xmu.edu.cn/linziyu>





主讲教师



2017年度厦门大学奖教金获得者

2020年度厦门大学奖教金获得者

主讲教师：厦门大学 林子雨 博士/副教授

中国高校首个“数字教师”提出者和建设者

2009年7月从事教师职业以来

累计**免费**网络发布超过**1500万**字高价值教学和科研资料

网络浏览量超过**1500万**次



提纲

3.1 程序控制结构

3.2 选择语句

3.3 循环语句

3.4 跳转语句

3.5 综合实例

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dblalab.xmu.edu.cn/post/python>

高等院校程序设计新形态精品系列

PYTHON

Python Programming Language

Python

程序设计基础教程

| 微课版 |

林子雨 赵江声 陶继平 编著



名师精品

多年计算机教学实践的厚积薄发



深入浅出

清晰呈现 Python 语言学习路径



实例丰富

有效提升编程语言的学习趣味



资源全面

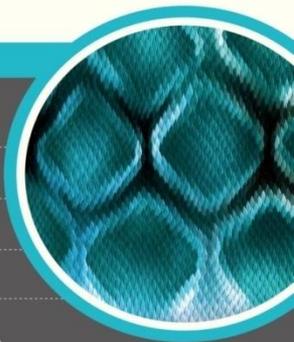
构建全方位一站式在线服务体系



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS





3.1 程序控制结构

Python程序具有三种典型的控制结构（如图3-1所示）：

（1）顺序结构：在程序执行时，按照语句的顺序，从上而下，一条一条地顺序执行，是结构化程序中最简单的结构。

（2）选择结构：又称为“分支结构”，分支语句根据一定的条件决定执行哪一部分的语句序列。

（3）循环结构：使同一个语句组根据一定的条件执行若干次。采用循环结构可以实现有规律的重复计算处理。

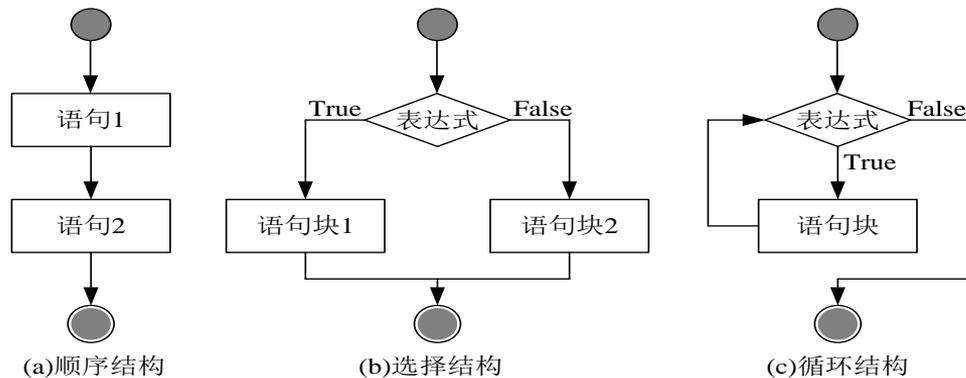


图3-1 程序的3种控制结构



3.2选择语句

3.2.1 if语句

3.2.2 if...else语句

3.2.3 if...elif...else多分支语句

3.2.4 if语句的嵌套

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dbl原因.xmu.edu.cn/post/python>

高等院校程序设计新形态精品系列

PYTHON

Python Programming Language

Python

程序设计基础教程

| 微课版 |

林子雨 赵江声 陶继平 编著



名师精品

多年计算机教学实践的厚积薄发



深入浅出

清晰呈现 Python 语言学习路径



实例丰富

有效提升编程语言的学习趣味



资源全面

构建全方位一站式在线服务体系



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS



3.2.1 if语句

简单的if语句就是对某种条件进行相应的处理，通常表现为“如果满足某种情况，那么就进行某种处理”，它的一般形式为：

if 表达式：
 语句块

其中，表达式可以是一个单一的值或者变量，也可以是由运算符组成的复杂语句。如果表达式的值为真，则执行语句块，如果表达式的值为假，则跳过语句块，继续执行后面的语句，具体流程如图3-2所示。

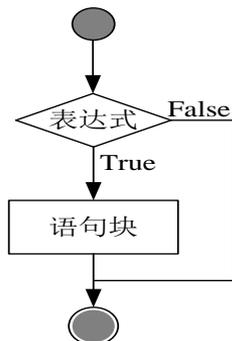


图3-2 if语句的执行流程



3.2.1 if语句

【例3-1】使用if语句求出两个数的较小值。

```
01      # two_number.py
02      a,b,c = 4,5,0
03      if a>b:
04          c = b
05      if a<b:
06          c = a
07      print("两个数的较小值是: ",c)
```



3.2.2 if...else语句

“if...else”语句也是选择语句的一种通用形式，通常表现为“如果满足某种条件，就进行某种处理，否则进行另一种处理”，它的一般形式为：

if 表达式:

 语句块1

else:

 语句块2

其中，表达式可以是一个单一的值或者变量，也可以是由运算符组成的复杂语句。如果表达式的值为真，则执行语句块1，如果表达式的值为假，则执行语句块2，具体流程如图3-3所示。需要注意的是，**else**不能单独使用，必须和**if**一起使用。

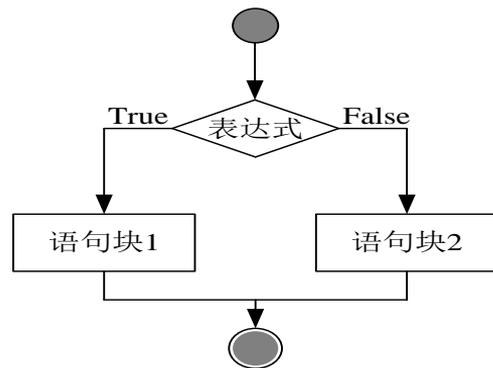


图3-3 if...else语句流程图



3.2.2 if...else语句

【例3-2】判断一个数是奇数还是偶数。

```
01      # odd_even.py
02      a = 5
03      if a % 2 == 0:
04          print("这是一个偶数。")
05      else:
06          print("这是一个奇数。")
```



3.2.3 if...elif...else多分支语句

“if...elif...else”多分支语句用于针对某一事件的多种情况进行处理，通常表现为“如果满足某种条件，就进行某种处理，否则如果满足另一种条件则执行另一种处理”，它的一般形式为：

if 表达式1:

 语句块1

elif 表达式2:

 语句块2

elif 表达式3:

 语句块3

...

else:

 语句块n

其中，表达式可以是一个单一的值或者变量，也可以是由运算符组成的复杂语句。如果表达式1的值为真，则执行语句块1，如果表达式1的值为假，则进入elif的判断，依此类推，只有在所有表达式都为假的情况下，才会执行else中的语句，具体流程如图3-4所示。需要注意的是，elif和else都不能单独使用，必须和if一起使用。



3.2.3 if...elif...else多分支语句

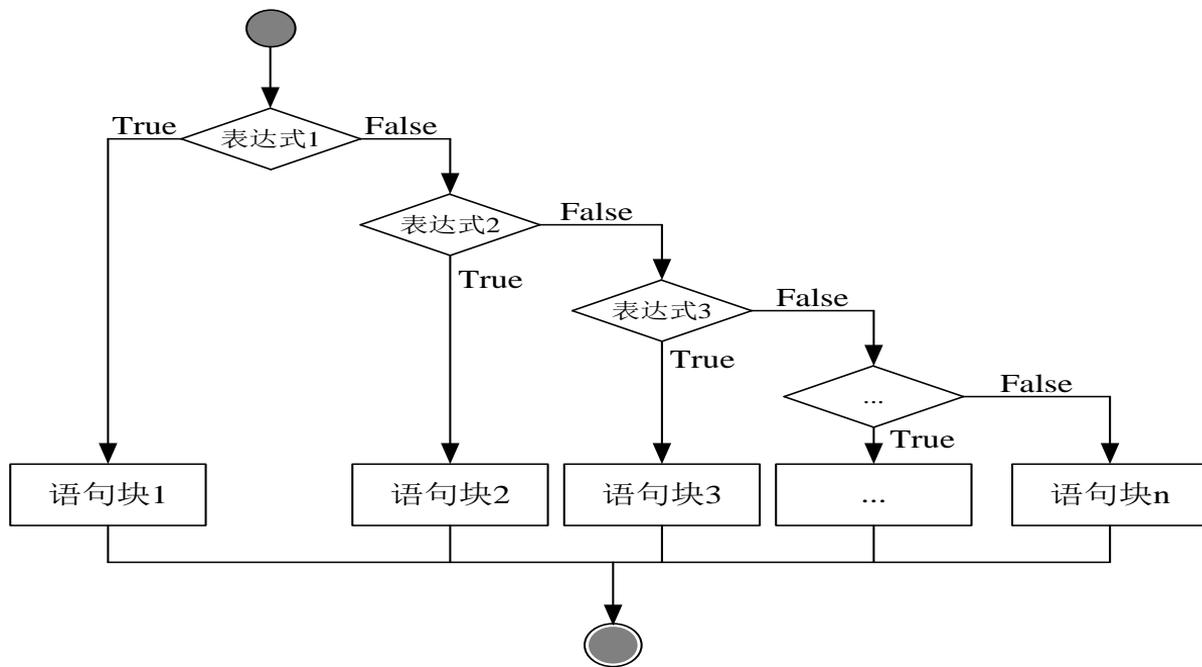


图3-4 if...elif...else语句流程图



3.2.3 if...elif...else多分支语句

【例3-3】判断每天上课的内容。

```
01      # lesson.py
02      day = int(input("请输入第几天课程: "))
03      if day == 1:
04          print("第1天上数学课")
05      elif day == 2:
06          print("第2天上语文课")
07      else:
08          print("其他时间上计算机课")
```



3.2.4 if语句的嵌套

前面介绍了3种形式的选择语句，即if、if...else 和if...elif...else，这3种选择语句之间可以相互嵌套。

例如，在最简单的if语句中嵌套 if...else语句，形式如下：

if 表达式1:

 if 表达式2:

 语句块1

 else:

 语句块2



3.2.4 if语句的嵌套

再比如，在if...else语句中嵌套if...else语句，形式如下：

```
if 表达式1:  
    if 表达式2:  
        语句块1  
    else:  
        语句块2  
else:  
    if 表达式3:  
        语句块3  
    else:  
        语句块4
```

在开发程序时，需要根据具体的应用场景选择合适的嵌套方案。需要注意的是，在相互嵌套时，一定要严格遵守不同级别语句块的缩进规范。



3.2.4 if语句的嵌套

【例3-4】判断是否为酒后驾车。假设规定车辆驾驶员的血液酒精含量小于20mg/100ml 不构成酒驾，酒精含量大于或等于20mg/100ml为酒驾，酒精含量大于或等于80mg/100ml为醉驾。

```
01      # drunk-driving.py
02      alcohol = int(input("请输入驾驶员每100ml血液酒精的含量: "))
03      if alcohol < 20:
04          print("驾驶员不构成酒驾")
05      else:
06          if alcohol < 80:
07              print("驾驶员已构成酒驾")
08          else:
09              print("驾驶员已构成醉驾")
```



3.2.4 if语句的嵌套

【例3-5】判断数学成绩属于哪个等级。成绩大于等于90分为优，成绩大于等于75分并且小于90分为良，成绩大于等于60分并且小于75分为及格，成绩小于60分为不及格。

```
01      # math_score.py
02      math = int(input("请输入数学成绩: "))
03      if math >= 75:
04          if math >= 90:
05              print("数学成绩为优")
06          else:
07              print("数学成绩为良")
08      else:
09          if math >=60:
10              print("数学成绩及格了")
11          else:
12              print("数学成绩不及格")
```



3.2.4 if语句的嵌套

【例3-6】判断某一年是否闰年。闰年的条件是：（1）能被4整除，但不能被100整除的年份都是闰年，如1996年、2004年是闰年；（2）能被100整除，又能被400整除的年份是闰年，如2000年是闰年。不符合这两个条件的年份不是闰年。

```
01      #year.py
02      year=int(input("请输入年份: "))
03      if year % 4 == 0:
04          if year % 100 == 0:
05              if year % 400 == 0:
06                  flag = 1
07              else:
08                  flag = 0
09          else:
10              flag = 1
11      else:
12          flag = 0
13      if flag == 1:
14          print(year,"年是闰年")
15      else:
16          print(year,"年不是闰年")
```



3.3 循环语句

3.3.1 while循环语句

3.3.2 for循环语句

3.3.3 循环嵌套

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dbl原因.xmu.edu.cn/post/python>

高等院校程序设计新形态精品系列

PYTHON

Python Programming Language

Python

程序设计基础教程

| 微课版 |

林子雨 赵江声 陶继平 编著



名师精品

多年计算机教学实践的厚积薄发



深入浅出

清晰呈现 Python 语言学习路径



实例丰富

有效提升编程语言的学习趣味



资源全面

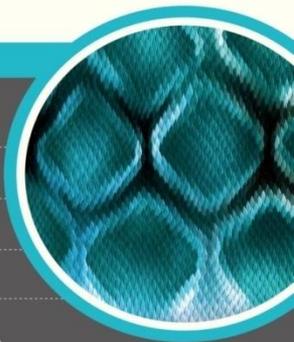
构建全方位一站式在线服务体系



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS





3.3.1 while循环语句

while语句是用一个表达式来控制循环的语句，它的一般形式为：

while 表达式:
语句块

当表达式的返回值为真时，执行语句块（或称为“循环体”），然后重新判断表达式的返回值，直到表达式的返回值为假时，退出循环。具体执行流程如图3-5所示。

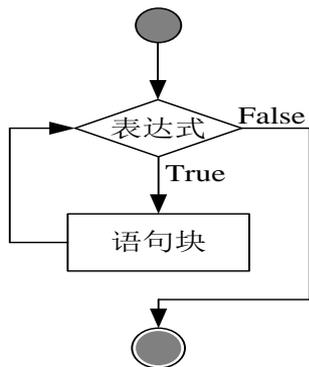


图3-5 while循环语句流程图



3.3.1 while循环语句

【例3-7】用while循环实现计算1~99的整数和。

```
01      # int_sum.py
02      n = 1
03      sum = 0
04      while(n <= 99):
05          sum += n
06          n += 1
07      print("1~99的整数和是: ",sum)
```



3.3.1 while循环语句

【例3-8】设计一个小游戏，让玩家输入一个数字，程序判断是奇数还是偶数。

```
01      # digit.py
02      prompt = '输入一个数字，我将告诉你，它是奇数，还是偶数'
03      prompt += '\n输入“结束游戏”，将退出本程序：'
04      exit = '结束游戏' # 退出指令
05      content = "" #输入内容
06      while content != exit:
07          content = input(prompt)
08          if content.isdigit(): #isdigit()函数用于检测字符串是否只由数字组成
09              number = int(content)
10              if (number % 2 == 0):
11                  print('该数是偶数')
12              else:
13                  print('该数是奇数')
14          elif content != exit:
15              print('输入的必须是数字')
```



3.3.1 while循环语句

在编写while循环语句时，一定要保证程序正常结束，否则会造成“死循环”（或“无限循环”）。例如，在下面的代码中，i的值永远小于100，运行后程序将不停地输出0。

```
01     i=0
02     while i<100:
03         print(i)
```



3.3.2 for循环语句

for语句是最常用的循环语句，一般用在循环次数已知的情况下，它的一般形式为：

for 迭代变量 in 对象:

 语句块

其中，迭代变量用于保存读取出的值；对象为要遍历或迭代的对象，该对象可以是任何有序的序列对象，如字符串、列表和元组等。被执行的语句块也称为“循环体”。for循环的具体执行流程如图3-6所示。

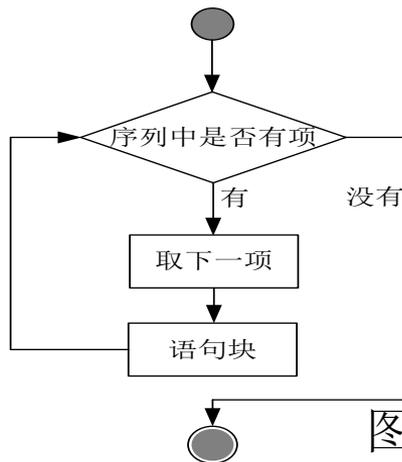


图3-6 for循环语句流程图



3.3.2 for循环语句

【例3-9】用for循环实现计算1~99的整数和。

```
01      # int_sum_for.py
02      sum=0
03      for n in range(1,100): #range(1,100)用于生成1到100（不包括100）的整
数
04          sum+=n
05      print("1到99的整数和是: ",sum)
```

在上面实例中，用到了range()函数，该函数的具体用法如下：

- range(stop): 生成从0开始到stop结束（不包含stop）的一系列数值。比如，range(3)生成的数值是0、1、2。
- range(start,stop): 生成从start开始到stop结束（不包含stop）的一系列数值。比如，range(2,5)生成的数值是2、3、4。
- range(start,stop,step): 生成从start开始到stop结束（不包含stop）、步长为step的一系列数值。比如，range(2,10,2)生成的数值是2、4、6、8，range(10,1,-2)生成的数值是10、8、6、4、2。



3.3.2 for循环语句

【例3-10】输出所有的“水仙花数”。所谓“水仙花数”是指一个3位数，其各位数字立方和等于该数本身。例如，153是一个水仙花数，因为 $153=1^3+5^3+3^3$ 。

```
01      # narcissus.py
02      for i in range(100,1000):
03          a = i % 10 #个位数
04          b = i // 10 % 10 #十位数
05          c = i // 100 #百位数
06          if(i == a ** 3 + b ** 3 + c ** 3):
07              print(i)
```



3.3.2 for循环语句

【例3-11】判断一个数是否是素数。判断一个数 m 是否是素数的算法是：让 m 被2到除，如果 m 能被2-到之间任何一个整数整除，则可以判断 m 不是素数；如果 m 不能被2-到之间的任何一个整数整除，则可以判断 m 是素数。

```
01 # prime.py
02 #由于程序中要用到求平方根的函数sqrt(), 因此需要导入math模块
03 import math
04 m = int(input("请输入一个数m: "))
05 n = int(math.sqrt(m)) # math.sqrt(m)返回m的平方根
06 prime = 1
07 for i in range(2,n+1):
08     if m % i == 0:
09         prime = 0
10 if(prime == 1):
11     print(m,"是素数")
12 else:
13     print(m,"不是素数")
```



3.3.3 循环嵌套

循环的嵌套就是在一个循环体内又包含另一个完整的循环结构，而在这个完整的循环体内还可以嵌套其他的循环结构。循环嵌套很复杂，在for语句、while语句中都可以嵌套，并且在它们之间也可以相互嵌套。例如，在while循环中嵌套while循环的格式如下：

while 表达式1:

 while 表达式2:

 语句块2

 语句块1

在for循环中嵌套for循环的格式如下：

for 迭代变量1 in 对象1:

 for 迭代变量2 in 对象2:

 语句块2

 语句块1



3.3.3 循环嵌套

在while循环中嵌套for循环的格式如下：

while 表达式:

for 迭代变量 **in** 对象:

 语句块2

 语句块1

在for循环中嵌套while循环的格式如下：

for 迭代变量 **in** 对象:

while 表达式:

 语句块2

 语句块1



3.3.3 循环嵌套

【例3-12】 分别输入两个学生的3门成绩，并分别计算平均成绩。
使用while循环嵌套实现，具体代码如下：

```
01 # avg_score_while.py
02 j = 1          # 定义外部循环计数器初始值
03 while j <= 2: # 定义外部循环为执行两次
04     sum = 0    # 定义成绩初始值
05     i = 1     # 定义内部循环计数器初始值
06     name = input('请输入学生姓名:') # 接收用户输入的学生姓名，赋值给name变量
07     while i <= 3: # 定义内部函数循环3次，就是接收3门课程的成绩
08         print('请输入第%d门的考试成绩: %i) # 提示用户输入成绩
09         sum = sum + int(input()) # 接收用户输入的成绩，赋值给sum
10         i += 1 # i变量自增1，i变为2，继续执行循环，直到i等于4时，跳出循环
11     avg = sum / (i-1) # 计算学生的平均成绩，赋值给avg
12     print(name, '的平均成绩是%d\n'%avg) # 输出学生成绩平均值
13     j = j + 1 # 内部循环执行完毕后，外部循环计数器j自增1，变为2，再进行外部循环
14 print('学生成绩输入完成!')
```



3.3.3 循环嵌套

【例3-13】用for循环的嵌套完成例【3-10】。

```
01      # narcissus_for.py
02      for a in range(10): #个位数的范围是0~9
03          for b in range(10): #十位数的范围是0~9
04              for c in range(1,10): #百位数的范围是1~9
05                  if(a + 10 * b + 100 * c == a ** 3 + b ** 3 + c ** 3):
06                      print(a + 10 * b + 100 * c)
```



3.3.3 循环嵌套

【例3-14】打印九九乘法表。

```
01      # multiplication_table.py
02      for i in range(1, 10):
03          for j in range(1, i+1):
04              print('{}x{}={}\t'.format(j, i, i*j), end=" ")
05          print()
```

该程序的执行结果如图3-7所示。

```
1x1=1
1x2=2   2x2=4
1x3=3   2x3=6   3x3=9
1x4=4   2x4=8   3x4=12   4x4=16
1x5=5   2x5=10  3x5=15  4x5=20  5x5=25
1x6=6   2x6=12  3x6=18  4x6=24  5x6=30  6x6=36
1x7=7   2x7=14  3x7=21  4x7=28  5x7=35  6x7=42  7x7=49
1x8=8   2x8=16  3x8=24  4x8=32  5x8=40  6x8=48  7x8=56  8x8=64
1x9=9   2x9=18  3x9=27  4x9=36  5x9=45  6x9=54  7x9=63  8x9=72  9x9=81
```

图3-7 九九乘法表打印效果



3.3.3 循环嵌套

【例3-15】输入一个行数（必须是奇数），输出类似如下的图形：

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



3.3.3 循环嵌套

```
01      # triangle.py
02      rows = int(input('输入行数（奇数）： '))
03      if rows%2!=0:
04          for i in range(0, rows//2+1): #控制打印行数
05              for j in range(rows-i,0,-1): #控制空格个数
06                  print(" ",end=") #打印空格，不换行
07                  for k in range(0, 2 * i + 1): #控制星号个数
08                      print("*",end=") #打印星号，不换行
09                  print("") #换行
10          for i in range(rows//2,0,-1): #控制打印行数
11              for j in range(rows-i+1,0,-1): #控制空格个数
12                  print(" ",end=") #打印空格，不换行
13                  for k in range(2*i-1,0,-1): #控制星号个数
14                      print("*",end=") #打印星号，不换行
15                  print("") #换行
```



3.4 跳转语句

3.4.1 break跳转语句

3.4.2 continue跳转语句

3.4.3 pass语句

本PPT是如下教材的配套讲义：

《Python程序设计基础教程（微课版）》

厦门大学 林子雨,赵江声,陶继平 编著，人民邮电出版社

《Python程序设计基础教程（微课版）》教材官方网站：

<http://dbl原因.xmu.edu.cn/post/python>

高等院校程序设计新形态精品系列

PYTHON

Python Programming Language

Python

程序设计基础教程

| 微课版 |

林子雨 赵江声 陶继平 编著



名师精品

多年计算机教学实践的厚积薄发



深入浅出

清晰呈现 Python 语言学习路径



实例丰富

有效提升编程语言的学习趣味



资源全面

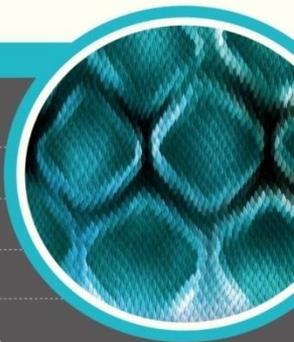
构建全方位一站式在线服务体系



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS





3.4.1 break跳转语句

break语句可以用在**for**、**while**循环中，用于强行终止循环。只要程序执行到**break**语句，就会终止循环体的执行，即使循环条件没有达到**False**条件或者序列还没被完全递归完，也会停止执行循环语句。如果使用嵌套循环，**break**语句将跳出当前的循环体。在某些场景中，如果需要在某种条件出现时强行中止循环，而不是等到循环条件为 **False** 时才退出循环，就可以使用 **break** 来完成这个功能。



3.4.1 break跳转语句

在while语句中使用break语句的形式如下：

while 表达式1:

 语句块

 if 表达式2:

 break

在for语句中使用break语句的形式如下：

for 迭代变量 in 对象:

 if 表达式:

 break



3.4.1 break跳转语句

【例3-16】使用break语句跳出for循环

```
01      # break.py
02      for i in range(0, 10) :
03          print("i的值是: ", i)
04          if i == 2 :
05              # 执行该语句时将结束循环
06              break
```

上面代码的执行结果如下：

i的值是: 0

i的值是: 1

i的值是: 2

从执行结果可以看出，当执行到i的值为2时，程序就退出了循环。



3.4.1 break跳转语句

【例3-17】使用break语句跳出while循环

```
01      # break1.py
02      x = 1
03      while True:
04          x += 1
05          print(x)
06          if x >= 4:
07              break
```

上面代码的执行结果如下：

```
2
3
4
```

从执行结果可以看出，当执行到x的值为4的时候，程序就跳出了循环。



3.4.1 break跳转语句

【例3-18】使用break语句跳出嵌套循环的内层循环

```
01      # break2.py
02      for i in range(0,3) :
03          print("此时i的值为:",i)
04          for j in range(5):
05              print("此时j的值为:",j)
06              if j==1:
07                  break
08          print("跳出内层循环")
```



3.4.1 break跳转语句

上面代码的执行结果如下：

此时i的值为: 0

此时j的值为: 0

此时j的值为: 1

跳出内层循环

此时i的值为: 1

此时j的值为: 0

此时j的值为: 1

跳出内层循环

此时i的值为: 2

此时j的值为: 0

此时j的值为: 1

跳出内层循环



3.4.1 break跳转语句

从执行结果可以看出，在内层循环中，每当执行到j的值为1时，程序就会跳出内层循环，转而执行外层循环的代码。

如果想达到 **break**语句不仅跳出单前所在循环，同时跳出外层循环的目的，可先定义布尔类型的变量来标志是否需要跳出外层循环，然后在内层循环、外层循环中分别使用两条 **break**语句来实现。



3.4.1 break跳转语句

【例3-19】使用break语句跳出嵌套循环的内层循环和外层循环

```
01      # break3.py
02      exit_flag = False
03      # 外层循环
04      for i in range(0, 5) :
05          # 内层循环
06          for j in range(0, 3) :
07              print("i的值为: %d, j的值为: %d" % (i, j))
08              if j == 1 :
09                  exit_flag = True
10                  # 跳出内层循环
11                  break
12          # 如果exit_flag为True, 跳出外层循环
13          if exit_flag :
14              break
```



3.4.1 break跳转语句

上面代码的执行结果如下：

i的值为: 0, j的值为: 0

i的值为: 0, j的值为: 1

从执行结果可以看出，当执行到i的值为0并且j的值为1时，程序不仅跳出了内层循环，也跳出了外层循环，程序执行结束。



3.4.2 continue跳转语句

`continue`语句和`break`语句不同，`break`语句跳出整个循环，而`continue`语句跳出本次循环，也就是说，程序遇到`continue`语句后，会跳过当前循环的剩余语句，然后继续进行下一轮循环。

在`while`语句中使用`continue`语句的形式如下：

`while` 表达式1:

语句块

`if` 表达式2:

`continue`

在`for`语句中使用`continue`语句的形式如下：

`for` 迭代变量 `in` 对象:

`if` 表达式:

`continue`



3.4.2 continue跳转语句

【例3-20】 使用continue语句跳出for循环的某次循环

```
01      # continue.py
02      for i in range(5):
03          if i == 3:
04              continue
05          print("i的值是:",i)
```

上面代码的执行结果如下：

i的值是: 0

i的值是: 1

i的值是: 2

i的值是: 4

从执行结果可以看出，当执行到i等于3时，程序跳出了该次循环，没有执行打印语句，继续执行下一次循环。



3.4.2 continue跳转语句

【例3-21】使用continue语句跳出while循环的某次循环

```
01      # continue1.py
02      i = 0
03      while i < 5:
04          i += 1
05          if i == 3:
06              continue
07          print("i的值是:",i)
```

上面代码的执行结果如下：

i的值是: 1

i的值是: 2

i的值是: 4

i的值是: 5

从执行结果可以看出，当执行到i等于3时，程序跳出了该次循环，没有执行打印语句，继续执行下一次循环。



3.4.2 continue跳转语句

【例3-22】计算从0到100中所有奇数的和。

```
01      # continue2.py
02      sum = 0
03      x = 0
04      while True:
05          x = x + 1
06          if x > 100:
07              break
08          if x % 2 == 0:
09              continue
10          sum += x
11      print(sum)
```



3.4.3 pass语句

在Python中还有一个pass语句，表示空语句，它不做任何事情，一般起到占位作用。

【例3-23】应用for循环输出1~10之间的偶数，在不是偶数时，应用pass语句占个位置，方便以后对不是偶数的数进行处理。

```
01      # pass.py
02      for i in range(1,10):
03          if i % 2 == 0:
04              print(i,end=' ')
05          else:
06              pass
```



3.5综合实例

【例3-24】利用蒙特卡罗方法计算圆周率。

蒙特卡罗方法是一种计算方法。原理是通过大量随机样本去了解一个系统，进而得到所要计算的值。它非常强大和灵活，又相当简单易懂，很容易实现。对于许多问题来说，它往往是最简单的计算方法，有时甚至是唯一可行的方法。

这里介绍一下使用蒙特卡罗方法计算圆周率 π 的基本原理。如图3-8所示，假设有一个正方形的边长是 $2r$ ，内部有一个相切的圆，圆的半径为 r ，则它们的面积之比是 $\pi/4$ ，即用圆的面积（ πr^2 ）除以正方形的面积（ $4r^2$ ）。

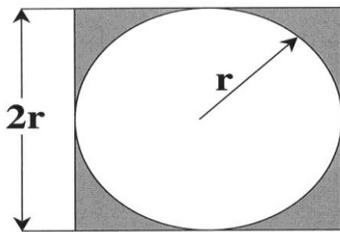


图3-8 一个正方形和一个圆形



3.5综合实例

现在，如图3-9所示，在这个正方形内部，随机产生10000个点（即10000个坐标对 (x, y) ），计算它们与中心点的距离，从而判断是否落在圆的内部。如果这些点均匀分布，那么圆内的点应该占到所有点的 $\pi/4$ ，因此，将这个比值乘以4，就是 π 的值。

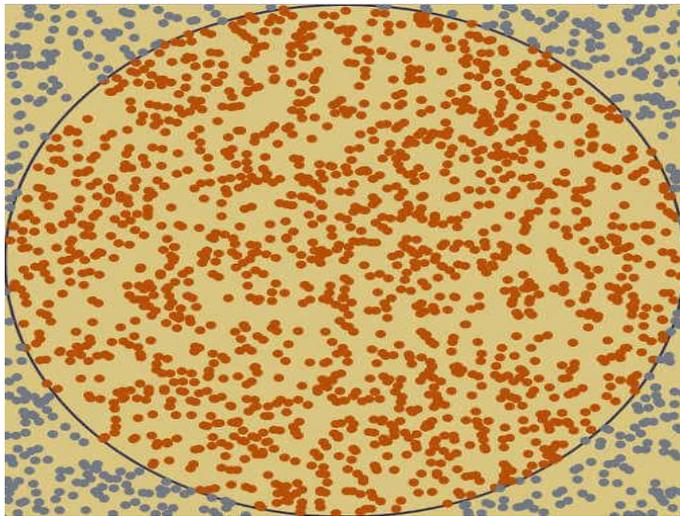


图3-9 蒙特卡罗方法计算圆周率 π 的基本原理



3.5综合实例

程序代码如下：

```
01      # pi.py
02      from random import random
03      n=10000
04      N=0
05      for i in range(1,n):
06          x,y=random(),random()  #random()函数用于生成一个0到1之间的随机
数
07          dis=pow(x**2+y**2,0.5)  #pow(a,b)函数返回a的b次幂
08          if dis<=1:
09              N=N+1
10      pi=4*N/n
11      print("圆周率为{}".format(pi))
```

在上面的代码中，随机产生的点的个数n的值越大，计算得到的圆周率的值越精确。



3.5综合实例

【例3-25】实现一个斐波那契数列。

斐波那契数列（Fibonacci sequence），又称黄金分割数列，因数学家莱昂纳多·斐波那契（Leonardoda Fibonacci）以兔子繁殖为例子而引入，故又称为“兔子数列”，指的是这样一个数列：0、1、1、2、3、5、8、13、21、34、.....在数学上，斐波那契数列以如下递归的方法定义：

$$F(0)=0 \quad (n=0)$$

$$F(1)=1 \quad (n=1)$$

$$F(n)=F(n-1)+F(n-2) \quad (n \geq 2)$$



3.5综合实例

实现一个斐波那契数列的程序代码如下：

```
01      # fibonacci.py
02      i, j = 0, 1
03      while i < 10000:
04          print(i)
05          i, j = j, i+j
```



3.5综合实例

【例3-26】 求出100~200之间的所有素数（素数只能被1和该数本身整除）。

```
01      # prime_all.py
02      import math
03      i = 0
04      for n in range(100,201):
05          prime = 1
06          k = int(math.sqrt(n)) # sqrt(n)方法返回数字n的平方根
07          for i in range(2,k+1):
08              if n % i == 0:
09                  prime = 0
10          if prime ==1:
11              print("%d是素数" % n)
```



3.5综合实例

【例3-27】打印出如下效果的实心三角形：

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



3.5综合实例

```
01      # triangle1.py
02      num = int(input("请输入打印行数: "))
03      for i in range(num):
04          tab = False #控制是否换行
05          for j in range(i+1):
06              print('*',end=") #打印星号, 不换行
07              if j == i:
08                  tab = True #控制是否换行
09          if tab:
10              print('\n',end = ") #换行
```




3.5综合实例

```
01 # triangle2.py
02 num = int(input("请输入打印行数: "))
03 for i in range(num):
04     tab = False #控制是否换行
05     for j in range(i + 1):
06         # 判断是否最后一行
07         if i != num-1:
08             # 循环完成, 修改换行标识符
09             if j == i :
10                 tab = True
11             # 判断打印空格还是*
12             if (i == j or j == 0):
13                 print("*",end=") #打印星号, 不换行
14             else :
15                 print(' ',end=") #打印空格, 不换行
16             # 最后一行, 全部打印星号
17             else :
18                 print("*", end=") #打印星号, 不换行
19         if tab:
20             print("\n", end=") #换行
```



3.5综合实例

【例3-29】 将一张面值为100元的人民币等值换成10元、5元和1元的零钞，有哪些组合。

```
01      # money.py
02      for i in range(100 // 1 + 1):
03          for j in range((100 - i * 1) // 5 + 1):
04              for k in range ((100 - i * 1 - j * 5) // 10 + 1):
05                  if i * 1 + j * 5 + k * 10 == 100:
06                      print("1元%d张, 5元%d张, 10元%d张" % (i,j,k))
```



3.5综合实例

【例3-30】求100以内能被3和7整除的数。

```
01      # devide.py
02      for i in range(1,101):
03          if i % 3 == 0 and i % 7 == 0:
04              print(i)
```



附录A：主讲教师林子雨简介



主讲教师：林子雨

单位：厦门大学计算机科学与技术系

E-mail: ziyulin@xmu.edu.cn

个人网页: <http://dmlab.xmu.edu.cn/post/linziyu>

数据库实验室网站: <http://dmlab.xmu.edu.cn>

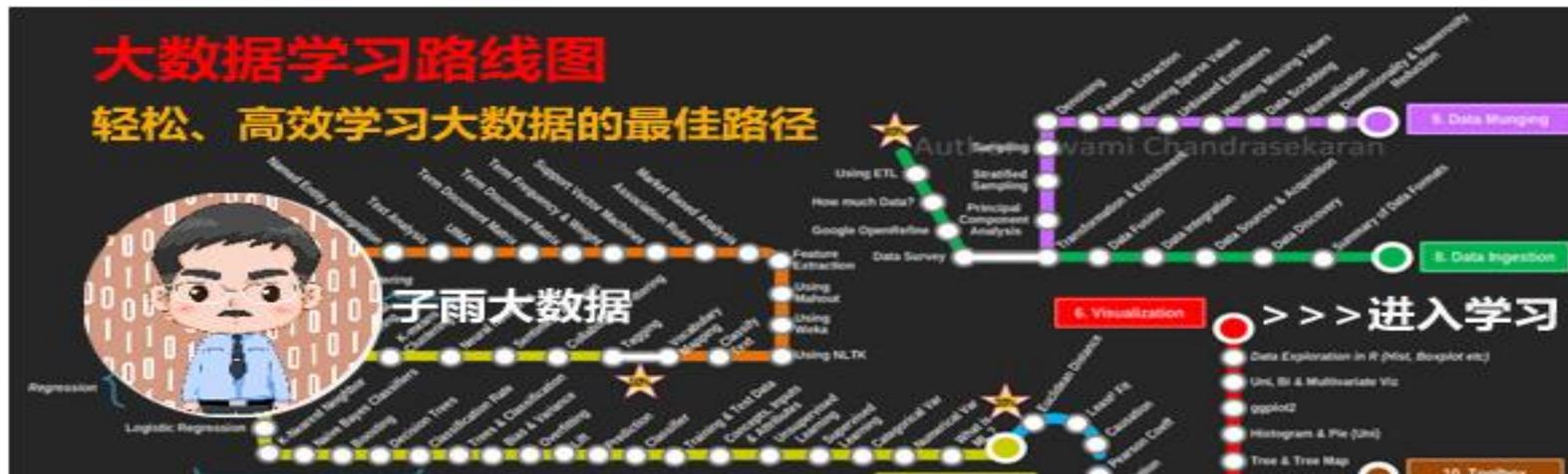


扫一扫访问个人主页

林子雨，男，1978年出生，博士（毕业于北京大学），全国高校知名大数据教师，现为厦门大学计算机科学系副教授，厦门大学信息学院实验教学中心主任，曾任厦门大学信息科学与技术学院院长助理、晋江市发展和改革局副局长。中国计算机学会数据库专业委员会委员，中国计算机学会信息系统专业委员会委员。国内高校首个“数字教师”提出者和建设者，厦门大学数据库实验室负责人，厦门大学云计算与大数据研究中心主要建设者和骨干成员，2013年度、2017年度和2020年度厦门大学教学类奖教金获得者，荣获2019年福建省精品在线开放课程、2018年厦门大学高等教育成果特等奖、2018年福建省高等教育教学成果二等奖、2018年国家精品在线开放课程。主要研究方向为数据库、数据仓库、数据挖掘、大数据、云计算和物联网，并以第一作者身份在《软件学报》《计算机学报》和《计算机研究与发展》等国家重点期刊以及国际学术会议上发表多篇学术论文。作为项目负责人主持的科研项目包括1项国家自然科学基金青年基金项目(No.61303004)、1项福建省自然科学基金项目(No.2013J05099)和1项中央高校基本科研业务费项目(No.2011121049)，主持的教改课题包括1项2016年福建省教改课题和1项2016年教育部产学协作育人项目，同时，作为课题负责人完成了国家发改委城市信息化重大课题、国家物联网重大应用示范工程区域试点泉州市工作方案、2015泉州市互联网经济调研等课题。中国高校首个“数字教师”提出者和建设者，2009年至今，“数字教师”大平台累计向网络免费发布超过1000万字高价值的研究和教学资料，累计网络访问量超过1000万次。打造了中国高校大数据教学知名品牌，编著出版了中国高校第一本系统介绍大数据知识的专业教材《大数据技术原理与应用》，并成为京东、当当网等网店畅销书籍；建设了国内高校首个大数据课程公共服务平台，为教师教学和学生学习大数据课程提供全方位、一站式服务，年访问量超过400万次，累计访问量超过1500万次。



附录B：大数据学习路线图



大数据学习路线图访问地址：<http://dblab.xmu.edu.cn/post/10164/>



附录C：林子雨大数据系列教材



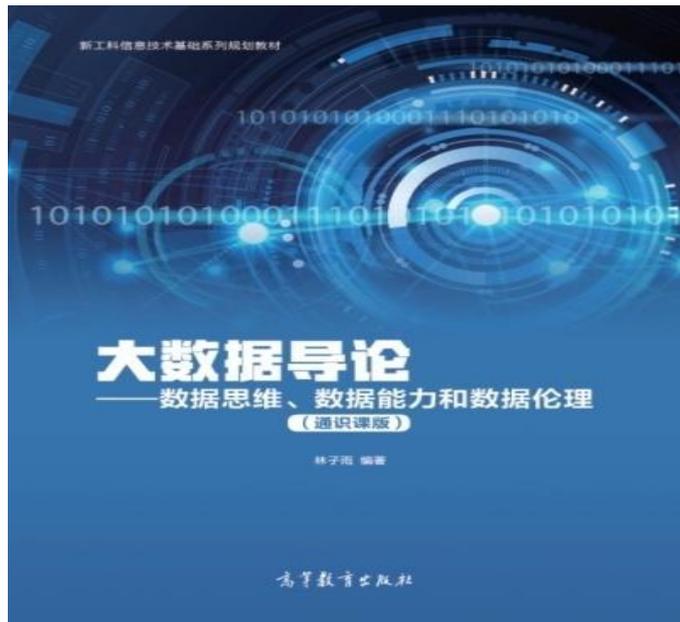
林子雨大数据系列教材
用于导论课、专业课、实训课、公共课

了解全部教材信息：<http://dbllab.xmu.edu.cn/post/bigdatabook/>



附录D：《大数据导论（通识课版）》教材

开设全校公共选修课的优质教材



本课程旨在实现以下几个培养目标：

- 引导学生步入大数据时代，积极投身大数据的变革浪潮之中
- 了解大数据概念，培养大数据思维，养成数据安全意识
- 认识大数据伦理，努力使自己的行为符合大数据伦理规范要求
- 熟悉大数据应用，探寻大数据与自己专业的应用结合点
- 激发学生基于大数据的创新创业热情

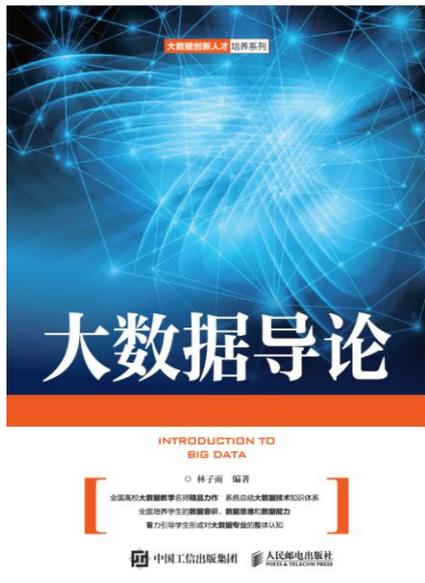
高等教育出版社 ISBN:978-7-04-053577-8 定价：32元 版次：2020年2月第1版
教材官网：<http://dbllab.xmu.edu.cn/post/bigdataintroduction/>



附录E：《大数据导论》教材

- 林子雨 编著《大数据导论》
- 人民邮电出版社，2020年9月第1版
- ISBN:978-7-115-54446-9 定价：49.80元

教材官网：<http://dblab.xmu.edu.cn/post/bigdata-introduction/>



开设大数据专业导论课的优质教材



扫一扫访问教材官网



附录F：《大数据技术原理与应用（第3版）》教材

《大数据技术原理与应用——概念、存储、处理、分析与应用（第3版）》，由厦门大学计算机科学系林子雨博士编著，是国内高校第一本系统介绍大数据知识的专业教材。人民邮电出版社 ISBN:978-7-115-54405-6 定价：59.80元

全书共有17章，系统地论述了大数据的基本概念、大数据处理架构Hadoop、分布式文件系统HDFS、分布式数据库HBase、NoSQL数据库、云数据库、分布式并行编程模型MapReduce、Spark、流计算、Flink、图计算、数据可视化以及大数据在互联网、生物医学和物流等各个领域的应用。在Hadoop、HDFS、HBase、MapReduce、Spark和Flink等重要章节，安排了入门级的实践操作，让读者更好地学习和掌握大数据关键技术。

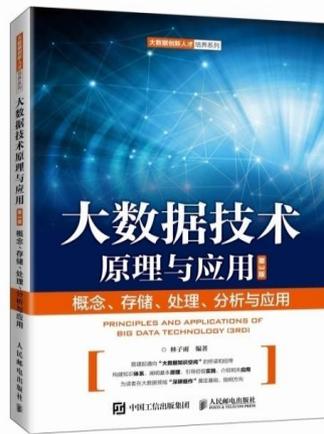
本书可以作为高等院校计算机专业、信息管理等相关专业的大数据课程教材，也可供相关技术人员参考、学习、培训之用。

欢迎访问《大数据技术原理与应用——概念、存储、处理、分析与应用》教材官方网站：

<http://dbllab.xmu.edu.cn/post/bigdata3>



扫一扫访问教材官网





附录G：《大数据基础编程、实验和案例教程（第2版）》

本书是与《大数据技术原理与应用（第3版）》教材配套的唯一指定实验指导书

大数据教材



配套实验指导书



1+1黄金组合
厦门大学林子雨编著

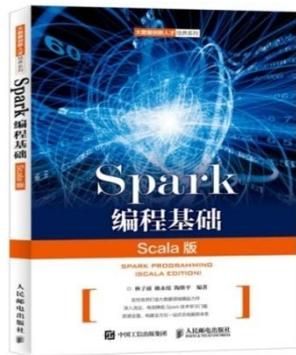
- 步步引导，循序渐进，详尽的安装指南为顺利搭建大数据实验环境铺平道路
- 深入浅出，去粗取精，丰富的代码实例帮助快速掌握大数据基础编程方法
- 精心设计，巧妙融合，八套大数据实验题目促进理论与编程知识的消化和吸收
- 结合理论，联系实际，大数据课程综合实验案例精彩呈现大数据分析全流程

林子雨编著《大数据基础编程、实验和案例教程（第2版）》

清华大学出版社 ISBN:978-7-302-55977-1 定价：69元 2020年10月第2版



附录H: 《Spark编程基础 (Scala版)》



《Spark编程基础 (Scala版)》

厦门大学 林子雨, 赖永炫, 陶继平 编著

披荆斩棘, 在大数据丛林中开辟学习捷径
填沟削坎, 为快速学习Spark技术铺平道路
深入浅出, 有效降低Spark技术学习门槛
资源全面, 构建全方位一站式在线服务体系

人民邮电出版社出版发行, ISBN:978-7-115-48816-9

教材官网: <http://dmlab.xmu.edu.cn/post/spark/>

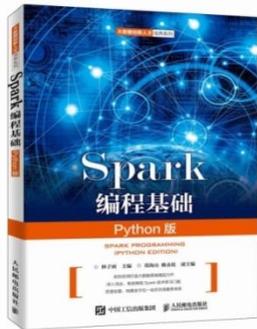


本书以Scala作为开发Spark应用程序的编程语言, 系统介绍了Spark编程的基础知识。全书共8章, 内容包括大数据技术概述、Scala语言基础、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作, 以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源, 包括讲义PPT、习题、源代码、软件、数据集、授课视频、上机实验指南等。



附录I: 《Spark编程基础 (Python版)》

《Spark编程基础 (Python版)》



厦门大学 林子雨, 郑海山, 赖永炫 编著

披荆斩棘, 在大数据丛林中开辟学习捷径
填沟削坎, 为快速学习Spark技术铺平道路
深入浅出, 有效降低Spark技术学习门槛
资源全面, 构建全方位一站式在线服务体系



人民邮电出版社出版发行, ISBN:978-7-115-52439-3

教材官网: <http://dbllab.xmu.edu.cn/post/spark-python/>

本书以Python作为开发Spark应用程序的编程语言, 系统介绍了Spark编程的基础知识。全书共8章, 内容包括大数据技术概述、Spark的设计与运行原理、Spark环境搭建和使用方法、RDD编程、Spark SQL、Spark Streaming、Structured Streaming、Spark MLlib等。本书每个章节都安排了入门级的编程实践操作, 以便读者更好地学习和掌握Spark编程方法。本书官网免费提供了全套的在线教学资源, 包括讲义PPT、习题、源代码、软件、数据集、上机实验指南等。



附录J：高校大数据课程公共服务平台



高校大数据课程

公 共 服 务 平 台

<http://dblab.xmu.edu.cn/post/bigdata-teaching-platform/>



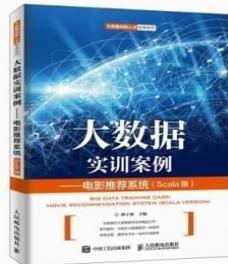
扫一扫访问平台主页 扫一扫观看3分钟FLASH动画宣传片



附录K：高校大数据实训课程系列案例教材

为了更好地满足高校开设大数据实训课程的教材需求，厦门大学数据库实验室林子雨老师团队联合企业共同开发了《高校大数据实训课程系列案例》，目前已经完成开发的系列案例包括：

- 《电影推荐系统》（已经于2019年5月出版）
- 《电信用户行为分析》（已经于2019年5月出版）
- 《实时日志流处理分析》
- 《微博用户情感分析》
- 《互联网广告预测分析》
- 《网站日志处理分析》



系列案例教材将于2019年陆续出版发行，教材相关信息，敬请关注网页后续更新！

<http://dblab.xmu.edu.cn/post/shixunkecheng/>



扫一扫访问大数据实训课程系列案例教材主页

The background is a solid blue color with faint, light-blue silhouettes of people. At the top, there are two groups of people holding hands, suggesting a community or team. On the right side, there is a silhouette of a person standing with their hand on their head. In the bottom left, there are silhouettes of people sitting at a table, possibly in a meeting or classroom setting.

Thank You!

Department of Computer Science, Xiamen University, 2022