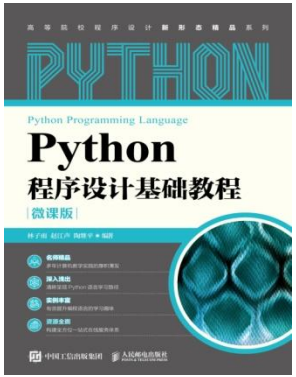
《Python程序设计基础教程（微课版）》

http://dblab.xmu.edu.cn/post/python



第2章 基础语法知识

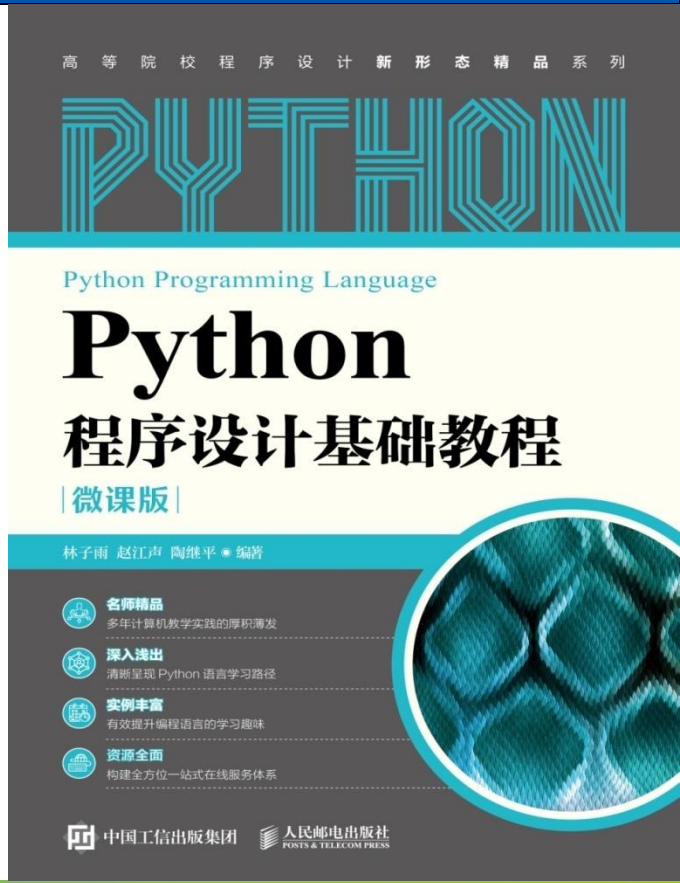
提纲

2.1关键字和标识符
2.2变量
2.3基本数据类型
2.4基本输入和输出
2.5运算符和表达式

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://dblab.xmu.edu.cn/post/python



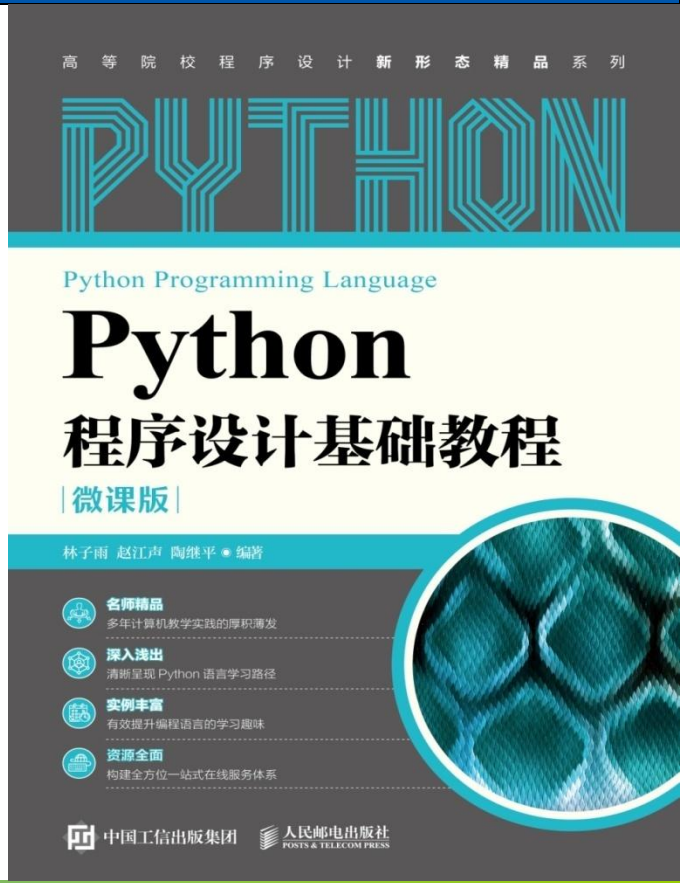
2.1关键字和标识符

2.1.1 关键字
2.1.2 标识符

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://dblab.xmu.edu.cn/post/python



2.1.1 关键字

Python语言中定义了一些专有词汇，统称为“关键字”，比如break、class、if、print等，它们都具有特定的含义，只能用于特定的位置。表2-1列出了Python语言的所有关键字。

表2-1 Python语言中的所有关键字

and	as	assert	async	'await	break
class	continue	def	del	elif	else
except	finally	for	from	False	global
if	import	'in	is	lambda	nonlocal
not	None	or	pass	raise	return
try	True	while	with	yield	

2.1.1 关键字

可以通过如下方式查看Python语言中的所有关键字：

```
>>> import keyword
>>> keyword.kwlist
```

需要注意的是，Python中的所有关键字是区分字母大小写的。例如，for是关键字，但是FOR就不属于关键字。

2.1.2 标识符

Python语言中的类名、对象名、方法名和变量名等，统称为“标识符”。为了提高程序的可读性，在定义标识符时，要尽量遵循“见其名知其意”的原则。Python标识符的具体命名规则如下：

（1）一个标识符可以由几个单词连接而成，以表明它的意思；

（2）标识符由一个或多个字母（A~Z 和 a~z）、数字、下划线（_），并且第一个字符不能是数字，没有长度限制；

（3）标识符不能是关键字；

（4）标识符中的字母是严格区分大小写的；

（5）标识符中不能包含空格、@、%和$等特殊字符；

（6）应该避免标识符的开头和结尾都使用下划线的情况，因为Python中大量采用这种名字定义了各种特殊方法和变量。

2.1.2 标识符

以下是合法的标识符：
MYNAME、name、name1、my_name
以下是非法的标识符
4gen　#以数字开头
for　#属于Python中的关键字
$book　#包含了特殊字符$

2.2变量

所谓“变量”，就是在程序运行过程中值可以被改变的量。和变量相对应的是“常量”，也就是在程序运行过程中值不能改变的量。需要注意的是，Python并没有提供定义常量的关键字，不过，在PEP8规范中定义了常量的命名规范，即常量由大写字母和下划线（_）组成，但是在实际应用中，常量首次被赋值以后，其值还是可以被其他代码修改。

2.2变量

变量的命名需要遵循以下规则：
（1）变量名必须是一个有效的标识符；
（2）变量名不能使用Python中的关键字；
（3）应选择有意义的单词作为变量名。
每个变量在使用前都必须赋值，变量赋值以后该变量才会在内存中被创建。Python中的变量在赋值时不需要类型声明。需要使用等号（=）来给变量赋值，等号运算符左边是一个变量名，等号运算符右边是存储在变量中的值。
例如，创建一个整型变量并为其赋值，可以使用如下语句：

```
>>> num = 128
```

2.2变量

可以看出，在为变量num赋值时，并没有声明其类型为整型，Python解释器会根据赋值语句来自动推断变量类型。

需要注意的是，在Python3.x中，允许变量名是中文字符，例如：

```
>>> 姓名 = "小明"
>>> print(姓名)
小明
```

2.2变量

但是，在实际编程中，不建议使用中文字符作为变量名。
在Python中，允许多个变量指向同一个值，例如：

```
>>> x=5
>>> id(x)
8791631005488
>>> y=x
>>> id(y)
8791631005488
```

在这段内码中，内置函数id()用来返回变量所指值的内存地址。可以看出，变量y和x具有相同的内存地址，这是因为，Python采用的是基于值的内存管理方式，如果为不同的变量赋值为相同值，这个值在内存中只有一份，多个变量指向同一块内存地址。

2.2变量

当修改了其中一个变量的值时，其内存地址将会发生变化，但是，这不会影响另一个变量，例如，我们可以在上面例子的基础上继续执行如下代码：

```
>>> x+=3
>>> x
8
>>> id(x)
8791631005584
>>> y
5
>>> id(y)
8791631005488
```

可以看出，当修改了变量x的值以后，其内存地址也发生了变化，但是，变量y的内存地址仍然没有发生变化。

2.2变量

需要说明的是，Python具有内存自动管理功能，对于没有任何变量指向的值，Python自动将其删除，回收内存空间，因此，开发人员一般情况下不需要考虑内存的管理问题。

Python还允许为多个变量同时赋值，例如：

```
>>> a = b = c = 1
>>> id(a)
8791631005360
>>> id(b)
8791631005360
>>> id(c)
8791631005360
```

上面这条赋值语句“a = b = c = 1”创建了一个整型对象，值为1。可以看出，三个变量a、b和c被分配到相同的内存空间上。

2.2变量

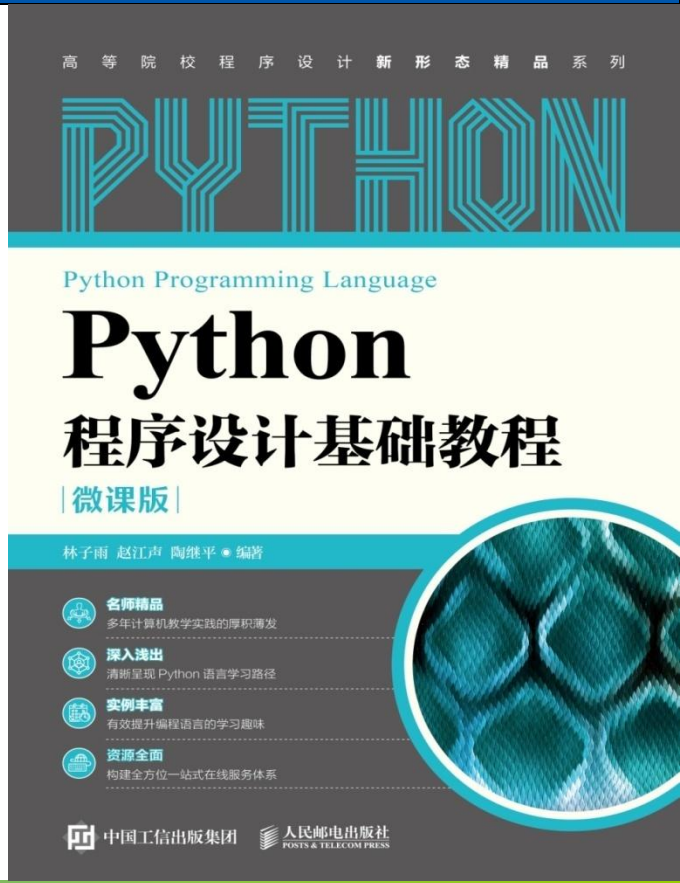
另外，Python语言是一种动态类型语言，变量的类型是可以随时变化的，例如：

```
>>> number = 512    #整型的变量
>>> print(type(number))
<class 'int'>
>>> number = "一流大学"   #字符串类型的变量
>>> print(type(number))
<class 'str'>
```

在上面的代码中，内置函数type()用来返回变量类型。可以看出，在刚开始创建变量number时，赋值为512，变量类型为整型，然后，再为变量number赋值为“一流大学”，它的类型就变为了字符串类型。

2.3基本数据类型

2.3.1数字
2.3.2字符串
2.3.3数据类型转换



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://dblab.xmu.edu.cn/post/python

2.3.1数字

1.整数

整数类型用来存储整数数值。在Python中，整数包括正整数、负整数和0。按照进制的不同，整数类型还可以划分为十进制整数、八进制整数、十六进制整数和二进制整数，具体如下：

（1）十进制整数：比如0、-3、8、110；

（2）八进制整数：使用8个数字0、1、2、3、4、5、6、7来表示整数，并且必须以0o开头，比如0o43、-0o123；

（3）十六进制整数：由0~9、A~F组成，必须以0x/0X开头，比如0x36、0Xa21f。

2.浮点数

浮点数也称为“小数”，由整数部分和小数部分构成，比如3.14、0.2、-1.648、5.8726849267842等。浮点数也可以用科学计数法表示，比如1.3e4、-0.35e3、2.36e-3等。

2.3.1数字

3.布尔类型

Python中的布尔类型主要用来表示“真”或“假”的值，每个对象天生具有布尔类型的True或False值。空对象、值为零的任何数字或者对象None的布尔值都是False。在Python3.x中，布尔值是作为整数的子类实现的，布尔值可以转换为数值，True的值为1，False的值为0，可以进行数值运算。

4.复数

复数由实数部分和虚数部分构成，可以用a + bj或者complex(a,b)表示，复数的实部a和虚部b都是浮点型。例如，一个复数的实部为2.38，虚部为18.2j，则这个复数为2.38+18.2j。

2.3.2字符串

字符串是 Python 中最常用的数据类型，它是连续的字符序列，一般使用单引号（' '）、双引号（" "）或三引号（''' '''或""" """）进行界定。其中，单引号和双引号中的字符序列必须在一行上，而三引号内的字符序列可以分布在连续的多行上，从而可以支持格式较为复杂的字符串。

例如，' xyz'、'123'、'厦门'、"hadoop"、'''spark'''、"""flink"""都是合法字符串，空字符串可以表示为''、" "或''' '''。

2.3.2字符串

下面代码中使用了不同形式的字符串：

```
01  # university.py
02  university = '一流大学'   #使用单引号，字符串内容必须在一行
03  motto = "自强不息，止于至善。"   #使用双引号，字符串内容必须在一行
04  #使用三引号，字符串内容可以分布在连续的多行上
05  target = '''建设成为世界一流的高水平、研究型大学，
06  为国家发展和民族振兴贡献力量！'''
07  print(university)
08  print(motto)
09  print(target)
```

上面这段代码的执行结果如下：

```
一流大学
自强不息，止于至善。
建设成为世界一流的高水平、研究型大学，
为国家发展和民族振兴贡献力量！
```

2.3.2字符串

Python支持转义字符，即使用反斜杠“\”对一些特殊字符进行转义。常用的转义字符如表2-1所示。

表2-1 常用的转义字符及其含义

转义字符	含义	转义字符	含义
\n	换行符	\r	回车
\t	制表符	\\	一个反斜杠\
\'	单引号	\"	双引号

比如，可以按照如下方式使用转义字符“\n”：

```
>>> print("自强不息\n止于至善")
自强不息
止于至善
```

2.3.3数据类型转换

在开发应用程序时，经常需要进行数据类型的转换。Python提供了一些常用的数据类型转换函数（见表2-2）。

表2-2 常用的数据类型转换函数

函数	作用
int(x)	把x转换成整数类型
float(x)	把x转换成浮点数类型
str(x)	把x转换成字符串
chr(x)	将整数x转换成一个字符
ord(x)	将一个字符x转换成对应的整数值

2.3.3数据类型转换

下面是一个关于学生成绩处理的具体实例，里面使用了数据类型转换函数：

```
>>> score_computer = 87.5
>>> score_englisht = 93.2
>>> score_math = 90.5
>>> score_sum = score_computer + score_englisht + score_math
>>> score_sum_str = str(score_sum)  #转换为字符串
>>> print("三门课程总成绩为："+score_sum_str)
三门课程总成绩为：271.2
>>> score_int = int(score_sum)  #丢弃小数部分，只保留整数部分
>>> score_int_str = str(score_int)
>>> print("去除小数部分的成绩为："+score_int_str)
去除小数部分的成绩为：271
```

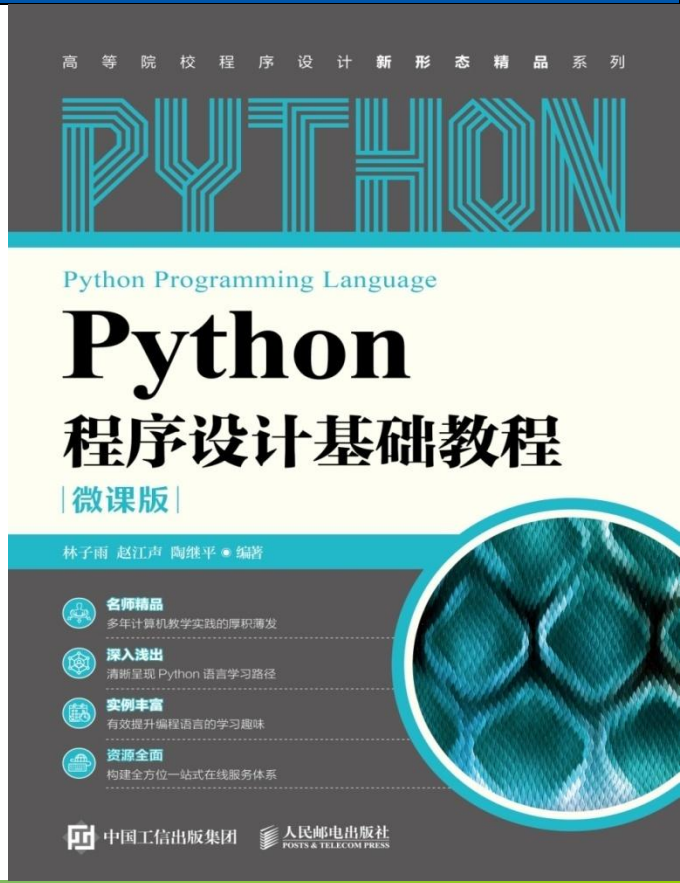
2.4基本输入和输出

2.4.1 使用input()函数输入
2.4.2 使用print()函数输出

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://dblab.xmu.edu.cn/post/python



2.4.1 使用input()函数输入

Python提供了内置函数input()，用于接收用户的键盘输入，该函数的一般用法为：

```
x = input("提示文字")
```

例如，编写代码要求用户输入名字，具体如下：

```
>>> name = input("请输入名字：")
请输入名字：小明
```

2.4.1 使用input()函数输入

需要注意的是，在Python3.x中，无论输入的是数字还是字符串，input()函数返回的结果都是字符串，下面代码很好地演示了这一点：

```
>>> x = input("请输入：")
请输入：8
>>> print(type(x))
<class 'str'>
>>> x = input("请输入：")
请输入：'8'
>>> print(type(x))
<class 'str'>
>>> x = input("请输入：")
请输入："8"
>>> print(type(x))
<class 'str'>
```

2.4.1 使用input()函数输入

从上面代码的执行效果可以看出，无论是输入数值8，还是字符串'8'或"8"，input()函数都是返回字符串。如果要接收数值，我们需要自己把接收到的字符串进行类型转换，例如：

```
>>> value = int(input("请输入："))
请输入：8
>>> print(type(value))
<class 'int'>
```

2.4.2 使用print()函数输出

1.print()函数的基本用法

Python使用内置函数print()将结果输出到IDLE或控制台上，其基本语法格式如下：

print(输出的内容)

其中，输出的内容可以是数字和字符串，也可以是表达式，下面是一段演示代码：

```
>>> print("计算乘积")
计算乘积
>>> x = 4
>>> print(x)
4
>>> y = 5
>>> print(y)
5
>>> print(x*y)
20
```

2.4.2 使用print()函数输出

print()函数默认是换行的，即输出语句后自动切换到下一行，对于Python3.x来说，如果要实现输出不换行的功能，那么可以设置end=''，下面是一段演示代码：

```
# xmu.py
print("自强不息")
print("止于至善")
print("自强不息",end='')
print("止于至善")
```

这段代码的执行结果如下：

```
自强不息
止于至善
自强不息止于至善
```

2.4.2 使用print()函数输出

默认情况下，Python将结果输出到IDLE或者标准控制台，实际上，在输出时也可以重定向，例如可以把结果输出到指定文件，示例代码如下：

```
>>> fp = open(r'C:\motto.txt','a+')
>>> print("自强不息，止于至善！",file=fp)
>>> fp.close()
```

上述代码执行以后，就可以看到在Windows系统的C盘根目录下生成了motto.txt文件。

2.4.2 使用print()函数输出

2.使用%进行格式化输出

（1）整数的输出

对整数进行格式化输出时，可以采用如下方式：

%o：输出八进制整数；

%d：输出十进制整数；

%x：输出十六进制整数。

下面是具体实例：

```
>>> print('%o' % 30)
36
>>> print('%d' % 30)
30
>>> print('%x' % 30)
1e
>>> nHex = 0xFF
>>> print("十六进制是%x,十进制是 = %d,八进制是 = %o" % (nHex,nHex,nHex))
十六进制是ff,十进制是 = 255,八进制是 = 377
```

2.4.2 使用print()函数输出

2.使用%进行格式化输出

（2）浮点数的输出

对浮点数进行格式化输出时，可以采用如下方式：

- %f：保留小数点后6位有效数字，如果是%.3f则保留3位小数；
- %e：保留小数点后6位有效数字，按指数形式输出，如果是%.3e，则保留3位小数位，使用科学记数法；
- %g：如果有6位有效数字，则使用小数方式，否则使用科学记数法，如果是%.3g，则保留3位有效数字，使用小数方式或科学记数法。

2.4.2 使用print()函数输出

下面是具体实例：

```
>> print('%f' % 2.22)  # 默认保留6位小数
2.220000
>>> print('%.1f' % 2.22)  # 取1位小数
2.2
>>> print('%e' % 2.22)  # 默认6位小数，用科学计数法
2.220000e+00
>>> print('%.3e' % 2.22)  # 取3位小数，用科学计数法
2.220e+00
>>> print('%g' % 2222.2222)  # 默认6位有效数字
2222.22
>>> print('%.7g' % 2222.2222)  # 取7位有效数字
2222.222
>>> print('%.2g' % 2222.2222)  # 取2位有效数字，自动转换为科学计数法
2.2e+03
```

2.4.2 使用print()函数输出

（3）字符串的输出

对字符串进行格式化输出时，可以采用如下方式：

- %s：字符串输出；
- %10s：右对齐，占位符10位；
- %-10s：左对齐，占位符10位；
- %.2s：截取2位字符串；
- %10.2s：10位占位符，截取两位字符串

2.4.2 使用print()函数输出

下面是具体实例：

```
>>> print('%s' % 'hello world')  # 字符串输出
hello world
>>> print('%20s' % 'hello world')  # 右对齐，取20位，不够则补位
    □□□□□□□□□hello world
>>> print('%-20s' % 'hello world')  # 左对齐，取20位，不够则补位
hello world□□□□□□□□□
>>> print('%.2s' % 'hello world')  # 取2位
he
>>> print('%10.2s' % 'hello world')  # 右对齐，取2位
    □□□□□□□□he
>>> print('%-10.2s' % 'hello world')  # 左对齐，取2位
he□□□□□□□□
>>> name = '小明'
>>> age = 13
>>> print('姓名：%s，年龄：%d' % (name, age))
姓名：小明，年龄：13
```

在上面的代码执行结果中，“□”是本教程人为标记的空格，在屏幕上只会显示空白。

2.4.2 使用print()函数输出

3.使用“f-字符串”进行格式化输出

使用“f-字符串”进行格式化输出的基本格式如下：

```
print(f'{表达式}')
```

下面是具体实例：

```
>>> name = '小明'
>>> age = 13
>>> print(f'姓名：{name}，年龄：{age}')
姓名：小明，年龄：13
```

2.4.2 使用print()函数输出

3.使用format进行格式化输出

相对于基本格式化输出采用‘%’的方法，format()功能更强大，该函数把字符串当成一个模板，通过传入的参数进行格式化，并且使用大括号‘{}’作为特殊字符代替‘%’。其用法有如下三种形式：

- 不带编号的“{}”；
- 带数字编号，可以调换显示的顺序，如“{1}”、“{2}”；
- 带关键字的，如“{key}”、“{value}”。

2.4.2 使用print()函数输出

下面是具体实例：

```
>>> print('{} {}'.format('hello','world'))  # 不带字段
hello world
>>> print('{0} {1}'.format('hello','world'))  # 带数字编号
hello world
>>> print('{0} {1} {0}'.format('hello','world'))  # 打乱顺序
hello world hello
>>> print('{1} {1} {0}'.format('hello','world'))  # 打乱顺序
world world hello
>>> print('{a} {b} {a}'.format(b='hello',a='world'))  # 带关键字
world hello world
```

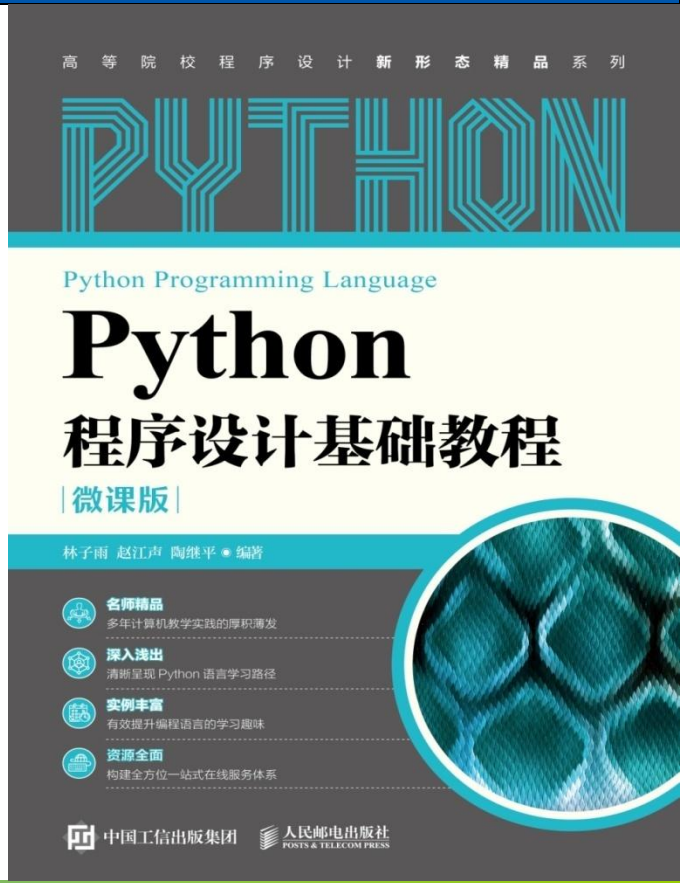
2.5运算符和表达式

2.5.1 算术运算符和表达式
2.5.2 赋值运算符和表达式
2.5.3 比较运算符和表达式
2.5.4 逻辑运算符和表达式
2.5.5 运算符的优先级与结合性

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://dblab.xmu.edu.cn/post/python



2.5.1 算术运算符和表达式

算术运算符主要用于数字的处理。常用的算术操作符如表2-3所示。

表2-3 算术运算符与表达式

运算符	说明	表达式
+	加（两个对象相加）	4+5（结果是9）
-	减（得到负数，或是一个数减去另一个数）	7-10（结果是-3）
*	乘（两个数相乘，或是返回一个被重复若干次的字符串）	4*5（结果是20）
/	除（x除以y）	10/4（结果是2.5）
%	取模（返回除法的余数）	10%4（结果是2）
**	幂（返回x的y次幂）	10**2（结果是100）
//	取整除（返回商的整数部分）	10//4（结果是2）

2.5.2 赋值运算符和表达式

赋值运算符主要用来为变量等赋值。赋值运算符的功能是将右侧表达式的值赋值给左侧的变量，因此，赋值运算符（=）并不是数学中的等于号。Python中常用的赋值运算符如表2-4所示。

表2-4 赋值运算符与表达式

运算符	说明	表达式	等价形式
=	简单的赋值运算	a=b	a=b
+=	加赋值	a+=b	a=a+b
-=	减赋值	a-=b	a=a-b
*=	乘赋值	a*=b	a=a*b
/=	除赋值	a/=b	a=a/b
%=	取余数赋值	a%=b	a=a%b
**=	幂赋值	a**=b	a=a**b
//=	取整除赋值	a//=b	a=a//b

需要注意的是，赋值运算符左侧只能是变量名，因为只有变量才拥有存储空间，可以把数值放进去。例如，表达式“a+b=c”或者“a=b+c=10”都是非法的。

2.5.3 比较运算符和表达式

比较运算符也称为“关系运算符”，主要用于比较大小，运算结果为布尔型。当关系表达式成立时，运算结果为True，当关系表达式不成立时，运算结果为False。Python中的比较运算符如表2-5所示。

表2-5 比较运算符与表达式

运算符	说明	表达式
>	大于	4>5（结果为False）
<	小于	4<5（结果为True）
==	等于	4==5（结果为False）
!=	不等于	4!=5（结果为True）
>=	大于等于	5>=4（结果为True）
<=	小于等于	4<=5（结果为True）

2.5.4 逻辑运算符和表达式

逻辑运算符用于对布尔型数据进行运算，运算结果仍为布尔型。Python中的逻辑运算符如表2-6所示。

表2-6 逻辑运算符与表达式

运算符	说明	表达式
and	逻辑与	exp1 and exp2
or	逻辑或	exp1 or exp2
not	逻辑非	not exp

2.5.4 逻辑运算符和表达式

在表2-6中，exp、exp1和exp2都是表达式。使用逻辑运算符进行逻辑运算时，其运算结果如表2-7所示。

表2-7 使用逻辑运算符进行逻辑运算时的结果

表达式1	表达式2	表达式1 and 表达式2	表达式1 or 表达式2	not 表达式1
True	True	True	True	False
True	False	False	True	False
False	False	False	False	True
False	True	False	True	True

2.5.5 运算符的优先级与结合性

•所谓优先级，就是当多个运算符同时出现在一个表达式中时，先执行哪个运算符。例如对于表达式“3+4*5”，Python会先计算乘法，再计算加法，得到结果为23，因为乘法的优先级要高于加法的优先级。

•所谓结合性，就是当一个表达式中出现多个优先级相同的运算符时，先执行哪个运算符：先执行左边的叫“左结合性”，先执行右边的叫“右结合性”。例如，对于表达式“100/5 *4”，/和*的优先级相同，应该先执行哪一个呢？这个时候就不能只依赖运算符优先级决定了，还要参考运算符的结合性。/和*都具有左结合性，因此先执行左边的除法，再执行右边的乘法，最终结果是80.0。

2.5.5 运算符的优先级与结合性

Python中大部分运算符都具有左结合性，也就是从左到右执行；只有乘方运算符（**）、单目运算符（例如not）、赋值运算符和三目运算符例外，它们具有右结合性，也就是从右向左执行。表2-8列出了常用的Python运算符的优先级和结合性。

表2-8 Python运算符优先级和结合性一览表

运算符	说明	结合性	优先级
()	小括号	无	
**	幂	右	高
+（正号）、-（负号）	符号运算符	右	↑
*、/、//、%	乘除	左	
+、-	加减	左	
==、!=、>、>=、<、<=	比较运算符	左	
not	逻辑非	右	
and	逻辑与	左	低
or	逻辑或	左	


Thank You!