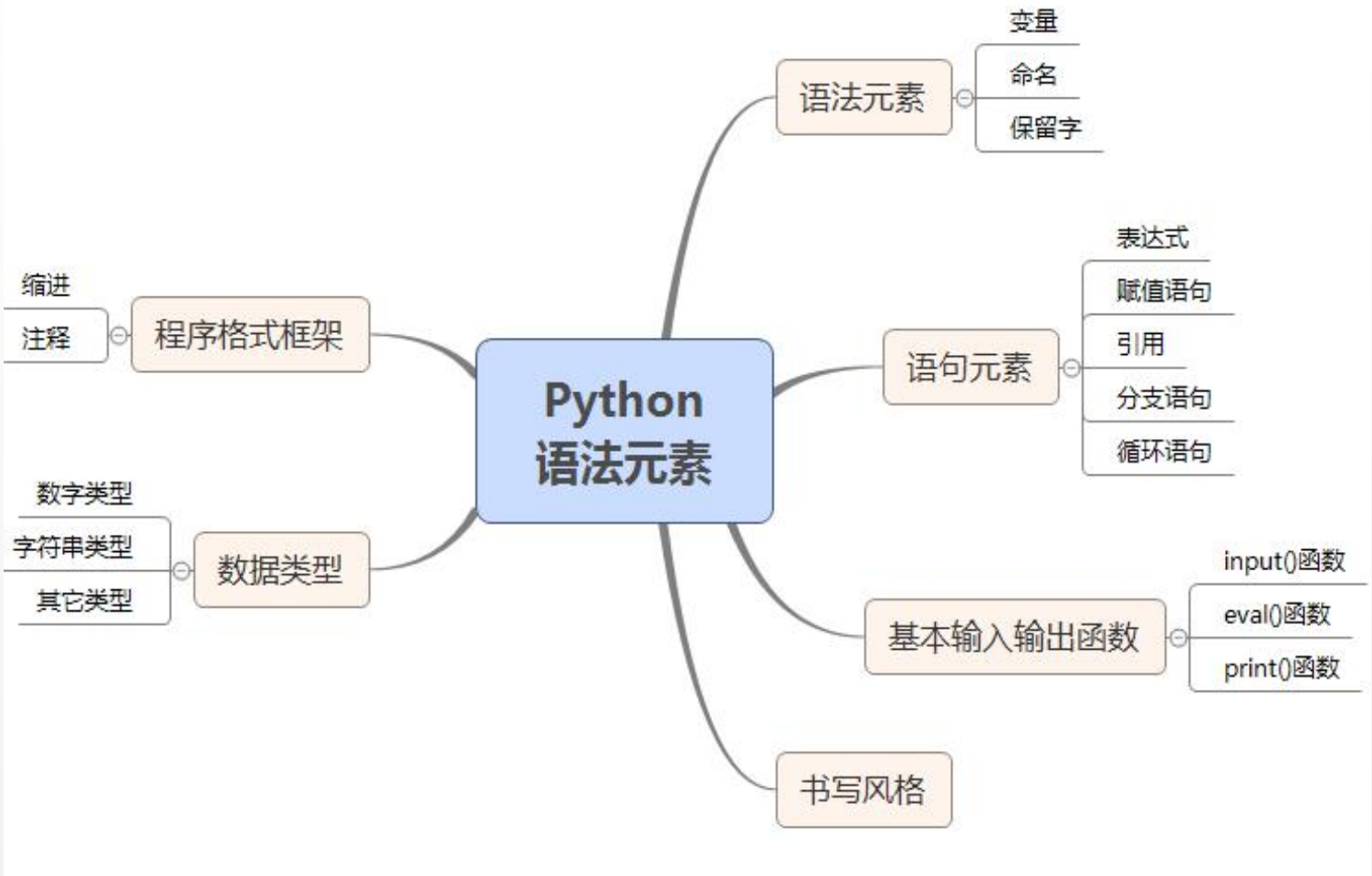
全国计算机等级考试二级教程
Python语言程序设计
(2018年版)

【第2章】
Python语言基本语法元素

考纲考点

- 程序的基本语法元素：程序的格式框架、缩进、注释、变量、命名、保留字、数据类型、赋值语句、引用
- 基本输入输出函数：input()、eval()、print()
- 源程序的书写风格

知识导图


Python
语法元素
语法元素
变量
命名
保留字
语句元素
表达式
赋值语句
引用
分支语句
循环语句
基本输入输出函数
input()函数
eval()函数
print()函数
书写风格
程序格式框架
缩进
注释
数据类型
数字类型
字符串类型
其它类型

程序的格式框架

缩进

- Python语言采用严格的“缩进”来表明程序的格式框架。缩进指每一行代码开始前的空白区域，用来表示代码之间的**包含和层次关系**。
- 1个缩进＝4个空格
- 缩进是Python语言中表明程序框架的**唯一手段**

缩进

- 当表达分支、循环、函数、类等程序含义时，在if、while、for、def、class等保留字所在完整语句后通过英文冒号（:）结尾并在之后进行缩进，表明后续代码与紧邻无缩进语句的所属关系。

程序的格式框架

单层缩进

```
#e1.1TempConvert.py
TempStr = input("请输入带有符号
if TempStr[-1] in ['F','f']:
    C = (eval(TempStr[0:-1]) -
    print("转换后的温度是{:.2f}C
elif TempStr[-1] in ['C','c']
    F = 1.8*eval(TempStr[0:-1]
    print("转换后的温度是{:.2f}F
else:
    print("输入格式错误")
```

多层缩进

```
DARTS = 1000
hits = 0.0
clock()
for i in range(1, DARTS):
    x, y = random(), random()
    dist = sqrt(x ** 2 + y **
    if dist <= 1.0:
        hits = hits + 1
pi = 4 * (hits/DARTS)
print("Pi的值是{:.2f}".format
```

注释

- 注释是代码中的辅助性文字，会被编译或解释器略去，不被计算机执行，一般用于程序员对代码的说明。Python语言采用#表示一行注释的开始，多行注释需要在每行开始都使用#。

```
1 #注释的第一行
2 #注释的第二行
3 #注释的第三行
```

注释

- Python程序中的非注释语句将按顺序执行，注释语句将被解释器过滤掉，不被执行。注释一般用于在代码中标明作者和版权信息，或解释代码原理及用途，或通过注释单行代码辅助程序调试。

```
1  # 作者名称：著名的非著名相声演员
2  # 编写时间：2018年1月1日
3  # 版权声明：按照CC BY-NC-SA方式开源
4  print("期待世界和平")  # 2018年的良好祝愿
```

语法元素的名称

变量

- 变量是保存和表示数据值的一种语法元素，在程序中十分常见。顾名思义，变量的值是可以改变的，能够通过赋值（使用**等号=表达**）方式被修改，例如：

```
>>>a = 99
>>>a = a + 1
>>>print(a)
100
```

命名

- Python语言允许采用大写字母、小写字母、数字、下划线(_)和汉字等字符及其组合给变量命名，但名字的首字符不能是数字，中间不能出现空格，长度没有限制
- 注意：**标识符对大小写敏感**，python和Python是两个不同的名字

保留字

- **保留字，也称为关键字**，指被编程语言内部定义并保留使用的标识符。
- 程序员编写程序不能定义与保留字相同的标识符。
- 每种程序设计语言都有一套保留字，保留字一般用来构成程序整体框架、表达关键值和具有结构性的复杂语义等。
- 掌握一门编程语言首先要熟记其所对应的保留字。

保留字

- Python 3.x保留字列表 (33个)

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

数据类型

数据类型概述

- Python语言支持多种数据类型，最简单的包括**数字类型**、**字符串类型**，略微复杂的包括元组类型、集合类型、列表类型、字典类型等。

数字类型

- 表示数字或数值的数据类型称为数字类型，Python语言提供3种数字类型：**整数、浮点数和复数**，分别对应数学中的整数、实数和复数。

数字类型

- 一个整数值可以表示为十进制、十六进制、八进制和二进制等不同进制形式。

十进制：　1010

十六进制：0x3F2

八进制：　0o1762

二进制：　0b001111110010

数字类型

- 一个浮点数可以表示为带有小数点的一般形式，也可以采用科学计数法表示。例如：浮点数123.456，两种表示方式如下：

  一般形式：　　123.456

  科学计数法：　　1.23456e2

- 复数类型与数学中的复数相一致，采用a+bj的形式表示，存在实部和虚部。

字符串

- Python语言中，字符串是用两个双引号“ ”或者单引号‘ ’括起来的一个或多个字符。
- Python字符串的两种序号体系

字符串

- 如果字符串长度为L，正向递增需要以最左侧字符序号为0，向右依次递增，最右侧字符序号为L-1；反向递减序号以最右侧字符序号为-1，向左依次递减，最左侧字符序号为-L。

```
>>>"对酒当歌,人生几何?"[1]
酒
>>>"对酒当歌,人生几何?"[-1]
?
```

字符串

- 可以采用[N: M]格式获取字符串的子串，这个操作被形象地称为切片。[N: M]获取字符串中从N到M（但不包含M）间连续的子字符串。

```
>>>"譬如朝露,去日苦多。"[2:4]
朝露
>>>"譬如朝露,去日苦多。"[5:-2]
去日苦
```

字符串

- 可以通过Python默认提供的len()函数获取字符串的长度，一个中文字符和西文字符的长度都记为1。

```
>>>len("譬如朝露,去日苦多。")
10
>>>len("Hello World")
11
```

程序的语句元素

表达式

- **产生或计算新数据值的代码片段称为表达式。**表达式类似数学中的计算公式，以表达单一功能为目的，运算后产生运算结果，运算结果的类型由操作符或运算符决定。
- 表达式一般由数据和操作符等构成，这是构成Python语句的重要部分。

赋值语句

- Python语言中，= 表示“赋值”，即将等号右侧的值计算后将结果值赋给左侧变量，包含等号（=）的语句称为“赋值语句”

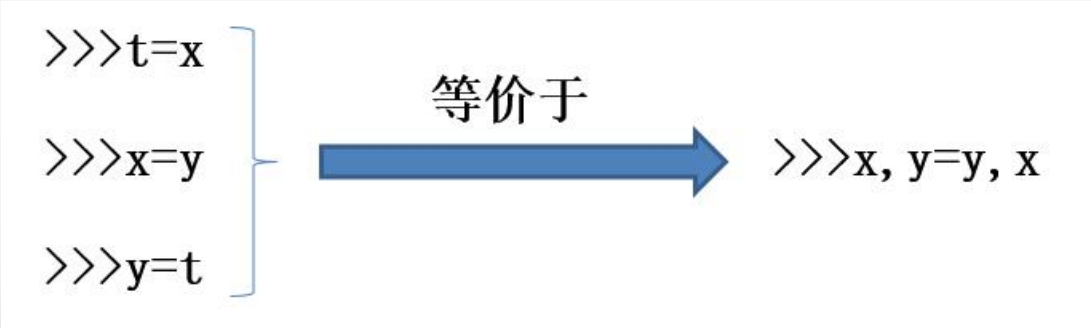
<变量> = <表达式>

- 同步赋值语句：同时给多个变量赋值

<变量1>, …, <变量N> = <表达式1>, …, <表达式N>

赋值语句

- 例：将变量x和y交换
- 采用单个赋值，需要3行语句：
  - 即通过一个临时变量t缓存x的原始值，然后将y值赋给x，再将x的原始值通过t赋值给y。
- 采用同步赋值语句，仅需要一行代码：



引用

- Python程序会经常使用当前程序之外已有的功能代码，这个过程叫“引用”。Python语言使用import保留字引用当前程序以外的功能库，使用方式如下：

import <功能库名称>

引用

- 引用功能库之后，采用<功能库名称>.<函数名称>()方式调用具体功能。

```
# 调用 turtle 库进行绘图操作
import turtle
turtle.fd(-200)       # fd()是 turtle 库中函数
turtle.right(90)      # right()是 turtle 库中函数
turtle.circle(200)    # circle()是 turtle 库中函数
```

其他语句

- 除了赋值语句外，Python程序还包括一些其他的语句类型，例如，**分支语句**和**循环语句**等。更多的分支和循环内容将在第4章介绍。这里仅简要介绍这两类语句的基本使用。

分支语句

- 分支语句是控制程序运行的一种语句，它的作用是根据判断条件选择程序执行路径。分支语句包括：单分支、二分支和多分支。
- 单分支语句是最简单的分支语句，使用方式如下：

if <条件>:

<语句块>

```
1  # 判断输入整数是否在[0,100]之间
2  num = eval(input("请输入一个整数:"))
3  if 0 <= num <= 100:          # 判断[0,100]
4     print("输入整数在0到100之间(含)")
```

循环语句

- 循环语句是控制程序运行的一类重要语句，与分支语句控制程序执行类似，它的作用是根据判断条件确定一段程序是否再次执行一次或者多次。循环语句包括遍历循环和条件循环。

while (<条件>):

<语句块1>

<语句块2>

```
1  # 输出10到100步长为3的全部整数
2  n = 10
3  while n < 100:
4     print(n, end=" ")
5     n = n + 3
```

基本输入输出函数

input()函数

- 获得用户输入之前，input()函数可以包含一些提示性文字

<变量> = input(<提示性文字>)

```
>>>a = input("请输入一个小数: ")
请输入一个小数: 123.456
>>>print(a)   # 此时a是字符串"123.456"
123.456
```

eval() 函数

- **eval(<字符串>)**函数是Python语言中一个十分重要的函数，它能够以Python表达式的方式解析并执行字符串，将返回结果输出

```
>>>a = eval("1.2 + 3.4")
>>>print(a)
4.6
```

eval() 函数

- eval()函数经常和input()函数一起使用，用来获取用户输入的数字，使用方式如下：

<变量> = eval(input(<提示性文字>))

```
>>>value = eval(input("请输入要计算的数值: "))
请输入要计算的数值: 1024.256
>>>print(value*2)
2047.512
```

print()函数

- print()函数用于输出运算结果，根据输出内容的不同，有三种用法。
- 第一种，仅用于输出字符串，使用方式如下：

print(<待输出字符串>)

```
>>>print("世界和平")
世界和平
```

print() 函数

- 第二种，仅用于输出一个或多个变量，使用方式如下：

print(<变量1>, <变量2>,..., <变量n>)

```
>>>value = 123.456
>>>print(value, value, value)
123.456 123.456 123.456
```

print()函数

- 第三种，用于混合输出字符串与变量值，使用方式如下：

print(<输出字符串模板>.format(<变量1>, <变量2>,..., <变量n>))

```
>>>a, b = 123.456, 1024
>>>print("数字{}和数字{}的乘积是{}".format(a, b, a*b))
数字123.456和数字1024的乘积是126417.944
```

print() 函数

- 对print()函数的end参数进行赋值

  print(<待输出内容>, end="<增加的输出结尾>")

```
>>>a = 24
>>>print(a, end=".")
24.
>>>print(a, end="%")
24%
```

实例解析：倒背如流

实例解析

- 计算机程序是机械记忆的高手，下面将编写一段程序，获得用户输入，采用倒叙方式将输入内容输出出来。该程序的输入和输出实例如下：
- 输入：To be or not to be, that's a question. ——莎士比亚
- 输出：亚比士莎—— .noitseuq a s'taht ,eb ot ton ro eb oT

实例解析

- 以下给出了一种实现，采用正向递增序号，利用len()函数将i设为最后一个字符的索引序号，然后逐次输出至第一个字符。

```
1  s = input("请输入一段文本: ")
2  i = len(s) - 1
3  while i >= 0 :
4      print(s[i], end="")
5      i = i - 1
```

实例解析

- 给出另一种实现，采用反向递减序号，将i首先设置为-1，直接索引最后一个字符，然后逐次输出至第一个字符。

```
s = input("请输入一段文本：")
i = - 1
while i >= - len(s):
    print(s[i], end="")
    i = i - 1
```

本章小结

本章具体讲解了初学Python需要知道的一些基本概念，初步介绍了Python基本语法元素，讲解了程序的格式框架、语法元素的名称、数据类型、程序的语句元素、基本输入输出函数等内容，进一步给出了Python源程序书写风格的思考和建议，帮助读者初步建立编写优美程序的基本观念。最后，讲解了“倒背如流”实例，通过完成将一段输入文本倒序输出的功能理解并实践Python基本语法元素。

Python程序说它可以倒背如流，人类的你要不要默写一下保留字来试试？