Python快速入门

嵩 天

Python基本编程解析(上)

嵩 天

Python快速入门

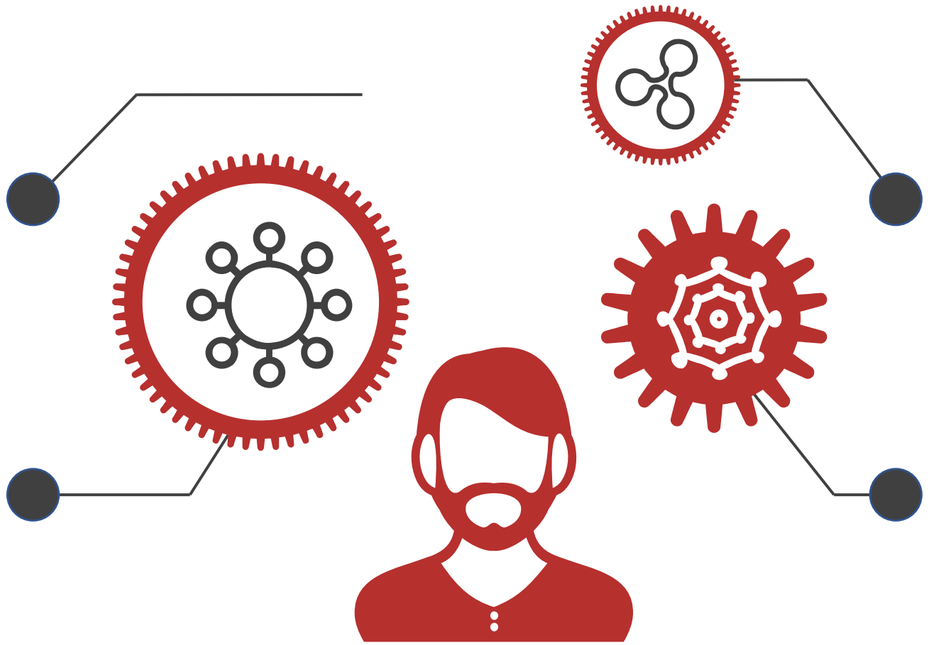
单元开篇

单元开篇

(1) 程序的格式框架

(2) 命名与保留字

(3) 数据类型



(4) 赋值与分支语句

(5) Python程序的输入输出

(6) “温度转换”代码分析

Python基本编程解析(上)

单元开篇

目的：概要了解Python编程的基本知识
学习编写10行的Python代码，开启编程之旅

1. 知道 Python编程的基本知识
2. 理解 Python温度转换程序
3. 能独立编写 一个10行左右类似功能的Python程序

Python基本编程解析(上)

Python快速入门

程序的格式框架

程序的格式框架

冒号+缩进：Python语法功能的一部分，表达代码的所属关系

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

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

程序的格式框架

冒号+缩进：Python语法功能的一部分，表达代码的所属关系

- 分支语句：if-elif-else
- 循环语句：for, while
- 异常处理：try-except-else-finally
- 函数定义：def
- 类定义：class

程序的格式框架

冒号+缩进：Python语法功能的一部分，表达代码的所属关系

- 冒号+缩进是语法的一部分，缩进不正确程序运行错误
- 冒号+缩进是表达代码间包含和层次关系的唯一方式
- 只需要所有缩进长度一致即可，可采用N个空格或Tab，建议4个空格

程序的格式框架

注释：程序中的辅助性说明信息

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

- 单行注释：#开始

例如：

```
# 这是一个单行注释
```

程序的格式框架

注释：程序中的辅助性说明信息

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
'''if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")'''
```

- 多行注释：一对'''

例如：''' 注释第一行
注释第二行
注释第三行'''

程序的格式框架

续行符 \: Python程序跨行书写的表示符号

```
#TempConvert.py
TempStr = input("请输入温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1] \
              ) - 32)/1.8
    print("温度是{}C".format(C))
else:
    print("输入格式错误")
```

- 续行符后不能存在空格
- 续行符后必须直接换行

Python快速入门

命名与保留字

Python123

命名与保留字

变量：程序中用于保存和表示数据的占位符号

- 变量的使用无需定义，可以直接使用
- 变量的赋值：使用等号(=)为变量赋值

```
s = 0
for i in range(10) :
    s += i
print(s)
```

命名与保留字

命名：设定标识符的过程，用于变量、函数、类名等

- Python 3使用Unicode编码，因此，可以支持中文等非西文符号
- 命名采用大小写字母、数字、下划线和中文等字符组成
- 大小写敏感、首字符不能是数字、不与保留字相同

命名与保留字

命名：设定标识符的过程，用于变量、函数、类名等

合法命名：

TempStr, Python, python, Python_Good, 应用基础, python123, True,...

注意：部分_* 或 __* 形式的名字被留作系统功能使用，建议不这样命名

命名与保留字

保留字：被编程语言内部定义并保留使用的标识符，共33个

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

Python快速入门

数据类型

Python123

数据类型

Python语言包括9种基本数据类型

- **数字类型**：整数、浮点数、复数
- **字节类型**：字符串、字节串
- **组合类型**：集合、元组、列表、字典

数据类型

(1) 数字类型：整数类型

- 与数学中的整数含义相同，无取值范围
- 整数包括二进制、八进制、十进制、十六进制等4种形式
- 如：0b1010 = 0o12 = 10 = 0xa

(2) 数字类型：浮点数类型

- 与数学中的实数含义相同，带有小数及小数的数字，存在取值范围
- 浮点数包括常规方法和科学计数法2种方式表示
- 如：0.0043 = 4.3e-3　　科学计数法：<a>e<b> 表示 $a*10^b$

数据类型

(3) 数字类型：复数类型

- 与数学中的复数概念相同，定义 **j** = $\sqrt{-1}$，复数表示为 a+b**j**
- z = a+b**j**，a是实部，b是虚部，a和b都是浮点数
- z.real获得z的实部，z.imag获得z的虚部

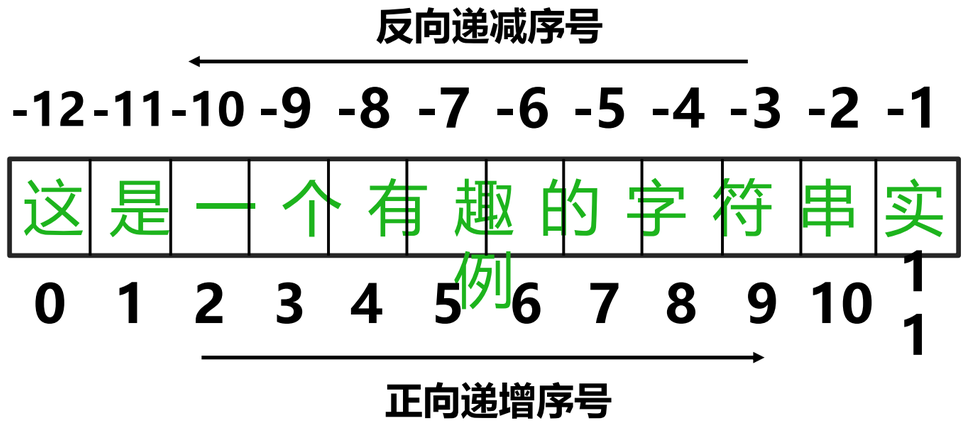
数据类型

(4) 字节类型：字符串类型

- 由0个或多个字符组成的有序字符序列
- 字符串由一对单引号或一对双引号表示，如："字符串" 或 '字符串'
- 字符串是字符的有序序列，可以用序号访问，如："字符串"[1]="符"

数据类型

(4) 字节类型：字符串类型（续）



- 索引：s[N] 通过序号获取单个字符

如："字符串"[-1]="串"

- 切片：s[N:M]获取M到N(不含)子串

如："字符串"[0:-1]="字符"

数据类型

(5) 字节类型：字节串类型

- 由0个或多个字节组成的有序序列，每字节对应值为0-255
- 字节串由前导符b或B与一对单引号或双引号表示，如：b"a\xf6"
- 0-255间非可打印字符用\xNN方式表示，N是一个十六进制字符

数据类型

(6) 组合类型：集合类型

- 多个元素的无序组合
- 集合使用大括号{}表示，元素间用逗号分隔，建立非集合使用{}或set()函数
- 如：**A = {"python", 123, ("python",123)}**

数据类型

(7) 组合类型：元组类型

- 序列类型的一种，元素间的有序组合，一旦创建不能被修改
- 元组使用小括号()表示，元素间用逗号分隔，小括号可以省略
- 如：rgb**color = 211, 11, 125**

数据类型

(8) 组合类型：列表类型

- 序列类型的一种，元素间的有序组合，类型不限，创建后可以随时被修改
- 列表使用中括号[]表示，元素间用逗号分隔，括号不可省略
- 如：**ls = ["cat", "tiger", 1024]**

数据类型

(9) 组合类型：字典类型

- 字典类型是键值对的集合，反映了数据之间的映射关系
- 字典使用大括号{}表示，键值间用冒号分隔，键值对间用逗号分隔
- 如：**d = {**"中国":"北京", "美国":"华盛顿", "法国":"巴黎"}

数据类型

Python语言包括9种基本数据类型

- **数字类型**：整数、浮点数、复数
- **字节类型**：字符串、字节串
- **组合类型**：集合、元组、列表、字典

Python快速入门

赋值与分支语句

Python123

赋值与分支语句

赋值语句：给变量赋予新数据值的过程

- 赋值语句以等号（=）为标志，运算=右侧值赋予左侧，同时赋值数据类型
- 如：C = (100-32) / 1.8　运算后C的值为 37.777777777778

赋值与分支语句

同步赋值语句：同时给多个变量赋值的过程

<变量1>, <变量2>, ..., <变量N> = <表达式1>, <表达式2>, ..., <表达式N>

- 同时赋值，可用于交换变量值

```
x = 99
y = 11
x, y = y, x
```

赋值与分支语句

分支语句：单分支、二分分支、多分支

- 分支语句使用保留字：if, elif, else

赋值与分支语句

单分支：仅使用if的分支语句

```
if    <条件>  :
      <语句块>
```

```
guess = eval(input())
if guess == 99:
    print("猜对了")
```

赋值与分支语句

二分支：使用if-else的分支语句

```
if   <条件>  :
        <语句块1>
else :
        <语句块2>
```

```
guess = eval(input())
if guess == 99:
    print("猜对了")
else:
    print("猜错了")
```

赋值与分支语句

多分支：使用if-elif-else的分支语句

```
if   <条件1> :
       <语句块1>
elif <条件2> :
       <语句块2>
       ……
else :
       <语句块N>
```

```
guess = eval(input())
if guess > 99:
    print("猜大了")
elif guess < 99:
    print("猜小了")
else:
    print("猜对了")
```

Python快速入门

Python程序的输入输出

Python123

Python程序的输入输出

输入函数input()：从控制台获得用户输入

<变量> = **input**(<提示性信息>)

- <提示性信息>为字符串形式，可省略
- <变量>为字符串类型

```
echo = input("请输入:")
```

Python程序的输入输出

输出函数print()：以字符形式向控制台输出结果

print(<拟输出字符串或字符串变量>)

- print()函数有3种主要的使用方法

Python程序的输入输出

print()用法1：将单一字符串或变量直接输出

```
echo = "这是一个字符串"
print(echo)
print("这是一个字符串")
```

```
这是一个字符串
这是一个字符串
```

Python程序的输入输出

print()用法2：将多个字符串或变量直接输出

```
echo1 = "字符串A"
echo2 = "字符串B"
print(echo1, echo2)
```

```
字符串A 字符串B
```

Python程序的输入输出

print()用法3：字符串和变量的混合输出

```
echo = "A"
print("这是变量{ }的输出".format(echo))
```

这是变量A的输出

Python程序的输入输出

回声程序：最短输入输出程序

```
print(input())
```

```
(输入)>你好
你好
```

Python快速入门

"温度转换"代码分析

Python123

“温度转换”代码分析


```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

注释

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

获得输入

输入形式为:

摄氏度:

28C

或

华氏度:

82F

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

多分支语句

冒号+缩进

if-elif-else

TempStr字符串

TempStr[]索引

列表类型[,]

保留字in

in保留字：成员判断

```
print('F' in ['F', 'f'])
print('C' in ['F', 'f'])
```

True

False

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

赋值语句

数值运算

eval()

TempStr[0:-1]

评估函数eval()

评估函数eval()：去掉参数最外侧引号并执行余下语句的函数

```
print(eval('1+2'))
eval('print('Hello')')
```

```
3
Hello
```

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

输出语句

混合输出用法

.format()

{:.2f}

保留小数点2位

“温度转换”代码分析

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```


Python 123
Thank you