Python编程基础
-- 基础语法

万永权

- 基本语法（注释、行与缩进）
- 保留字和标识符
- 变量和类型
- 运算符
- 输入和输出

注释

Python中的单行注释以#开头！

```
# 第一个注释
print (“Hello, Python!”)  # 第二个注释
```

注释

多行注释可以使用三引号作为开头和结束符号

```
"""
print(value, ..., sep=' ', end='\n',
file=sys.stdout, flush=False)
"""
```

行与缩进

python最具特色的就是使用缩进来表示代码块

```
if True:
        print ("True")
else:
        print ("False")
        print ("False")
```

```
if True:
        print ("True")
else:
        print ("False")
  print ("hello")
```

保留字与标识符

[概念]

保留字是Python语言中已经被赋予特定意义的一些单词，开发程序时，不可以作为变量、函数、类、模块和其他对象的名称来使用。

and	as	assert	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			

标识符

若希望在程序中表示一些事物，需要开发人员自定义一些符号和名称，这些符号和名称叫做标识符。

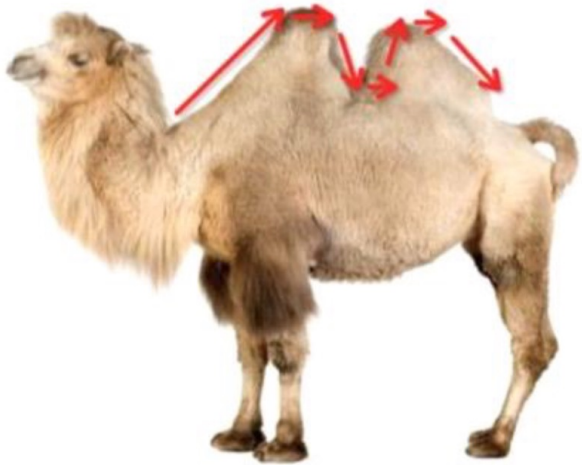
命名规则

- 标示符由字母、下划线和数字组成，且数字不能开头。
- Python中的标识符是区分大小写的。
- python中的标示符不能使用关键字

标识符

为了规范命名标识符，关于标识符的命名提以下建议：

1. 见名知意
2. 建议使用驼峰式



如：userName　　userLoginFlag


变 量

变量和赋值

Python中的变量用来存储数据，变量可以理解为去超市购物的菜篮子，其类型和值在赋值的那一刻被初始化。

```
num1 = 100
num2 = 87
result = num1 + num2
```

num1和num2变量就好比一个小菜篮子，它们分别存储的数据是100和87。result变量存储的数据是num1和num2这两个“菜篮子”中的数据累计之和。

变量

- 变量在第一次赋值的时候被创建，不需要事先声明；
- 变量仅仅是名称，本身没有类型，与它们绑定在一起的对象拥有类型；
- 每个变量在使用前都必须赋值，变量赋值以后该变量才会被创建。
- 等号（=）用来给变量赋值。

```
counter = 100 # 赋值整型变量
miles = 1000.0 # 浮点型
name = "John" # 字符串

miles=“1.6km”
name = counter
```

Python 中变量可以被赋值成不同的数据类型

Python变量

- 在Python中，**不需要事先声明变量名及其类型**，直接赋值即可创建各种类型的对象变量。这一点适用于Python任意类型的对象。

例如语句

```
>>> x = 3
```

凭空出现一个整型变量x

创建了整型变量x，并赋值为3，再例如语句

```
>>> x = 'Hello world.'
```

创建了字符串变量x，并赋值为’Hello world.’。

新的字符串变量，再也不是原来的x了

[常量]

在程序运行过程中，**值不能改变的量**。

基本数据类型

[概念]

数据类型就是数据的类型。


数字
类型
1
布尔
类型
3
数据类
型转换
4
字符串
类型
2

1 整数

2 浮点数

3 复数

字符串类型

字符串

- 字符串是连续的字符序列，可以是计算机所能表示的一切字符的集合。
- 用单引号、双引号或三引号界定的符号系列。
- 单引号、双引号、三单引号、三双引号可以互相嵌套，用来表示复杂字符串。

字符串

- 字符串属于不可变序列
- 空字符串表示为''或""
- 三引号'''或"""表示的字符串可以换行，支持排版较为复杂的字符串；三引号还可以在程序中表示较长的注释。

```
'abc'
'123'
'中国'
"Python"
'''Tom said, "Let's go"'''
```

[概念]

转义字符是指使用“\”对一些特殊字符进行转义。

转义字符	说 明
\	续行符
\n	换行符
\0	空
\t	水平制表符，用于横向跳到下一制表位
\"	双引号
\'	单引号
\\	一个反斜杠
\f	换页
\0dd	八进制数，dd代表的字符，如\012代表换行
\xhh	十六进制数，hh代表的字符，如\x0a代表换行

布尔类型

数据类型转换

int() → 整型

float() → 浮点型

str() → 字符串类型

hex() → 整数转换为十六进制字符串

oct() → 整数转换为八进制字符串

运算符


算术运算符
1
比较运算符
3
位运算符
5
赋值运算符
2
逻辑运算符
4

运算符的优先级

类　型	运算符	说　明
单目运算符	~、+、-	取反、正号和负号
算术运算符	*、/、%、//	乘、除、求余
	+、-	加、减
位运算符	<<、 >>	左移、右移
	&	位与
	^	位异或
	|	位或
比较运算符	<、<=、>、>=、!=、==	小于、小于等于、大于、大于等于、不等于、等于

优先级从高到底

输入和输出

```
variable = input("提示文字")
```

print(输出内容)

流程控制语句

- 分支结构
  - 简单的if else语句
  - 多分支if elif else语句
- 循环结构
  - for语句
  - while语句

3.2 分支结构

3.2.1 单分支结构

```
#coding:utf-8
ss = input("请输入你的成绩(0-100):")
si= int(ss)
if si < 60:
    print("不及格")
if si>=60 and si<70:
    print("及格")
if 80> si >=70:
    print("中")
if 90 > si >= 80:
    print("良")
if si >= 90:
    print("优")
```

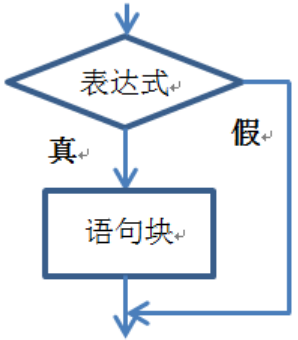
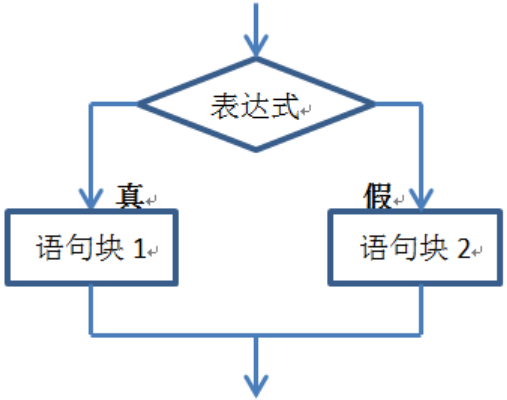


图 3.1 单分支结构

3.2.2 双分支结构

```
#coding:utf-8
ss=input("请输入分数(0-100):")
si=int(ss) if si>=60:
    print("及格")
else:
    print("不及格")
```



3.2 双分支结构

3.2.3 多分支结构

- 用if…elif…语句实现多分支结构

```
#coding:utf-8
ss = input("请输入你的成绩（0-100）： ")
si=int(ss)
if si<60:
    print("不及格")
elif si <70:
    print("及格")
elif si<80:
    print("中")
elif si<90:
    print("良")
elif si<=100:
    print("优")
```

3.3 循环结构

- 3.3.1 While循环
- While的语法结构为:
- 【格式一】
- While 条件表达式:
- 　　循环体
- 【格式二】
- While 条件表达式:
- 　　循环体
- else:
- 　　语句体

1.次数不确定的while循环

```
#coding:utf-8
sum=0
numstr = input("请输入一个整数(‘#’结束):  ")
while numstr != "#":
    numint = int(numstr)
    sum = sum + numint
    numstr = input("请输入一个整数(‘#’结束):  ")
print("sum = ", sum)
```

2. 次数确定的While循环

```
#codign:utf-8
sum = 0
i=1
while i<=100:
    sum = sum+i
    i=i+1
print("1+2+…+100=",sum)
```

3.3.2 for循环

◆1. 基于序列的for循环

```
#coding:utf-8
sum = 0
lista = [1,6,34,26,56,2,9,86,23]
for i in lista:
    sum = sum +i
print("sum=",sum)
```

2. 基于字典的for循环

```
#coding:utf-8
dicta= {'体育':78,'英语':86,'操作系统':93,'网络安全':63,'网络编程':74}
sum=0
avr=0
for key in dicta.keys():
    sum = sum + dicta[key]
avr = sum / len(dicta)
print("总成绩：",sum)
print("平均成绩：",avr)
```

3. 基于for迭代访问集合

```
#coding:utf-8
set_a = {1,3,34,31,67,98,-12,0,65}
sum = 0
for i in set_a:
    sum = sum + i
print("sum=",sum)
```

4. 基于range()函数的计数循环

```
#coding:utf-8
sum = 0
for i in range(2,101,2):
    sum = sum + i
print("sum=",sum)
```

3.3.3 循环嵌套

```
#coding:utf-8
for i in range(1,10):
    for j in range(1,i+1):
        print("%d×%d=%d"%(i,j,i*j),end = ' ')
    print()
```

3.3.4 break和continue语句

◆执行break语句会退出循环。

```
#coding:utf-8
from random import randint
numa= randint(1,100)
numstr = input("请猜一猜，这个数是多少(1 ~ 100)?")
while True:
    numint = int(numstr)
    if numint > numa:
        print("猜大了")
    elif numint < numa:
        print("猜小了")
    else:
        break
    numstr = input("再猜一次吧：")
print("你猜对了")
```

continue语句

- continue语句在while和for循环中起到提前结束本次循环的作用，并忽略continue后面的语句，然后回到循环的顶端，继续执行下一次循环。

```
#coding:utf-8
sum = 0
i=0
while i<100:
    i=i+1
    if i%2==0:
        continue
    print("i=%d"%i)
    sum=sum+i
print("1 ~ 100之间奇数的和为%d"%sum)
```

总结

- 变量与常量
- 基础数据类型与类型转换
- 运算符
- 分支结构
- 循环结构