LV5

基础语法

BBCLOUD 林凡

https://github.com/lvancer/course_python

lin029011@163.com

LV5

大纲

注释

打印

变量

数字

字符串

布尔值

条件

练习

lin029011@163.com

LV5

注释

编程过程中除了开发代码，还需要编写注释。

```
# 输出Hello World
print("Hello World")
```

Pycharm里会使用不同的颜色来表示注释很好识别

Python中以 # 开头后面的文字都是注释，常见的注释方式还有：

```
print("Hello World") # 输出Hello World
```

直接写在后面的

注释的作用代码解释，是给人看的，方便其他人理解代码。

注释另外一个作用就是注释代码，使代码无效化

```
# print('hehe')
print('xixi')
```

这样输出hehe的那句话就无效了，在调试过程中常用。

lin029011@163.com

LV5

打印

```
print('hello world')
```

print是python的内置方法，括号里面是这个方法的参数。

hello world被单引号包围，表示这是个字符串。

也可以用双引号：

```
print("HELLO WORLD")
```

二者大部分情况下没有区别。使用哪个都可以。

主要的区分就在于单引号和双引号的互相包裹上

```
print('hello world "Baby"')
print("HELLO WORLD 'baby'")
```

结果：

```
hello world "Baby"
HELLO WORLD 'baby'
```

在单引号中打印双引号，在双引号中可以输出单引号

我大部分使用单引号，因为不用多按一个shift

lin029011@163.com

LV5

打印

特殊字符的处理：使用反斜杆 `\` 来打印一些特殊字符，成为转义字符。

常用的几个转义字符

名称	代码	输出
换行符 \n	`print('第一行\n第二行')`	第一行 第二行
制表符 \t	`print('1\tmale')`	1　male
反斜杠 \\	`print('反斜杆\\')`	反斜杆\
单引号 \’	`print('单引号\'')`	单引号'
双引号 \”	`print("双引号\"")`	双引号"

这样单双引号也无所谓了

尝试输出一大段文字。

lin029011@163.com

LV5

打印

用户输入：从用户的输入获得信息，input方法

```
# 用户输入
name = input('请输入用户名: ')
print(name)
```

input方法会输出文字，并暂停程序，等待用户输入内容。

我们输入名字，回车。

结果会输出我们输入的名字。

这里name是一个变量，print打印了变量的值。

```
请输入用户名:
```

```
请输入用户名: 张三
```

```
请输入用户名: 张三
张三
```

lin029011@163.com

LV5

变量

```
print('hello world')
```

里面的hello world是常量，即不变的量。

与之相对的就是变量，即可改变的量。下面的name就是变量：

```
name = 'hello world'
print(name)
```

```
# 用户输入
name = input('请输入用户名: ')
print(name)
```

其意义就是一个用于存储数据的被命名的空间。上面的name就是变量名。

中间的等号表示：用等式右边的值赋值给左边的变量。

赋值之后，就可以用name来做为print的参数了。

变量还可以修改，修改后就是新的赋值了。

修改

```
name = 'hello world'
print(name)
# 变量修改
name = '你好啊'
print(name)
```

lin029011@163.com

LV5

变量

变量的命名法

1、不要使用没有意义的名字，如
```
a = '张三'
name = '张三'
```
用name就可以知道这变量的意义。

2、大小写区分，较长的名字我们一般使用小写加下划线的组合。如 `last_name = 'Green'`

3、不能用作名字开头的字符：数字。 `错误： 3name = 'Green'  正确： name3 = 'Green'`

4、python内置的关键字不可用：

变量数据	False	True	None	global	nonlocal	del	with	as
逻辑处理	if	elif	else	and	or	is	in	not
循环语句	for	while	continue	pass	break			
方法与类	def	class	return	import	from	lambda		
异常处理	try	except	raise	finally	assert			

这些关键字就是我们后面会陆续学到的语法。

lin029011@163.com

LV5

变量

变量不仅仅只有字符串一种类型。

最基础类型有字符串、整数、浮点数、布尔值、空值。

名称	类型	代码	取值
字符串	string	`var_int = 123`	所有文字
整数	int	`var_float = 1.23`	1，2，3，4….所有整数
浮点数	float	`var_string = '123'`	0.1，0.2…. 所有小数
布尔值	boolean	`var_boolean = True`	True, False
空值	None	`var_none = None`	None

空值是一个特殊的类型，设为空值的变量相当于没有赋值。

经常用来做逻辑上的非空判断，保证变量是有数据的。

lin029011@163.com

LV5

数字

数字包括整数和浮点数。

```
x = 23          # 整数
y = 3.16        # 浮点数
```

数学运算

```
x = 1
y = 1
z = x + y + 3
print(z)
```

运算优先级（小学数学）

```
x = 6
x = x + 5 * x - (x - 2)
```

运算符	说明
+	加法
-	减法
*	乘法
/	除法
**	次方
%	取余

浮点数与整数进行运算时，会先将整数转换成浮点数再进行运算

其他类型转换数字（一个常用的方法）

```
x = int('123')
y = float('1.23')
```

lin029011@163.com

LV5

数字

编写一个固定的计算器：用户输入两个整数，输出两个数字的乘积。

```
x = input('输入第一个整数: ')    # 输入字符串1
y = input('输入第二个整数: ')    # 输入字符串2
x_int = int(x)  # 转换成int类型
y_int = int(y)  # 转换成int类型
answer = x_int + y_int
print(x + ' * ' + y + ' = ' + str(answer))
```

输出：

```
输入第一个整数: 123
输入第二个整数: 234
123 * 234 = 28782
```

input输入的内容是字符串，所以我们要使用int方法把字符串转换为整数类型。

如果输入的内容无法转换为整数，如1.23，程序就会出错，这时就要用float方法。

print里面使用了字符串的加法，其含义就是将几个字符串串起来。

lin029011@163.com

LV5

字符串

转换：对于不是字符串类型的变量，我们要使用str方法进行转换。

```
age = str(12)
```

字符串拼接：使用加号（+）将几个字符串变量连接起来。

```
age = str(12)
name = 'Green'
s = name + ' is a ' + age + ' years old'
print(s)
```

输出：

```
Green is a 12 years old
```

这样我们就可以拼接出我们需要的任意字符串了。

但我们会觉得这样的拼接很难受也不好理解，这时候我们就可以用格式化输出了。

lin029011@163.com

LV5

字符串

格式化输出：使用占位符来表示要写入的变量，然后替换。

```
name = 'Green'
s = '{} is a {} years old'.format(name, 12)
print(s)
```

上面的代码同样也可以实现前面一个的输出。

python中使用{}来表示占位符，后面的format是字符串变量的的一个方法，用 . 调用。

format方法里依次填上占位符的内容，就可以用一种直观的方式进行字符串的拼接。

尝试下面的代码：

```
name = input('输入名字: ')
age = input('输入年龄: ')
print('{} is a {} years old'.format(name, age))
```

lin029011@163.com

LV5

字符串

获取字符串长度：len方法。

```
s = 'abcdefg'
l = len(s)
print('length is ' + str(l))
```

输出：

```
length is 7
```

截取字符串：从字符串中取出其中一段。观察变量s的内容：

a	b	c	d	e	f	g
0	1	2	3	4	5	6

数字表示了每个字的编号，从0开始，一直到6。

最小的编号一定是0，最大的编号是长度-1。接着我们就可以来用中括号[]来取字了。

```
print(s[1])      # 取某一位
print(s[3:5])    # 取某一段，3到5但不包含5
print(s[2:])     # 从第2位取到最后
print(s[:4])     # 从第1位取到第4位
```

输出：

```
b
de
cdefg
abcd
```

lin029011@163.com

LV5

字符串

大小写变化：upper，lower，capitalize

```
word = 'i love you'
print(word.upper())          # 大写
print(word.lower())          # 小写
print(word.capitalize())     # 首字母大写
```

输出：

```
I LOVE YOU
i love you
I love you
```

去头尾空格：strip

```
print(' abc '.strip())       # 去空格
```

输出：

```
abc
```

替换：find，replace

```
print('I love you'.find('you'))      # 找到返回位置
print('I love you'.find('me'))       # 未找到返回-1
print('I love you'.replace('you', 'me'))    # 将you替换成me
```

```
7
-1
I love me
```

lin029011@163.com

LV5

布尔值

布尔值只有True和False两种，主要用于条件的判断。

对比运算符：

==	!=	<	>	<=	>=
等于	不等于	小于	大于	小等于	大等于

注意等于==要与赋值号=区分

条件表达式：

```
a = 5
b = a == 5   # True
b = a != 5   # False
b = a > 2    # True
b = a < 4    # False
```

```
s = 'abc'
b = s == 'abc'   # True
b = s.find('a') > 0      # 判断是否存在a, True
b = s.find('e') > 0      # 判断是否存在e, False
b = s.startswith('b')    # 判断是否以b开头, False
b = s.endswith('c')      # 判断是否以c结尾, True
```

lin029011@163.com

LV5

布尔值

布尔值运算：and，or，not。

```
a = True
b = False
print(a and b)  # False
print(a or b)   # True
print(not a)    # False
```

and运算需要计算两边都为True，才为True，其他情况为False。含义为并且。

or运算只要有一个是True，就为True，两边都是False时为False。含义为或者。

not运算是将当前的布尔值变换成另一个。含义为否。

这个表达式是True还是False？ `a or b and (a or b) and not b`

lin029011@163.com

LV5

条件

理解了布尔值，我们就可以开始进行条件判断了。

```
x = 5
if x > 10:                    # if引导条件语句，用冒号结尾
   print('x > 10')          # 当条件语句为True时，执行该语句
   print('end')
```

if 关键字表示如果，当后面的条件语句为True时，执行冒号下面的语句。

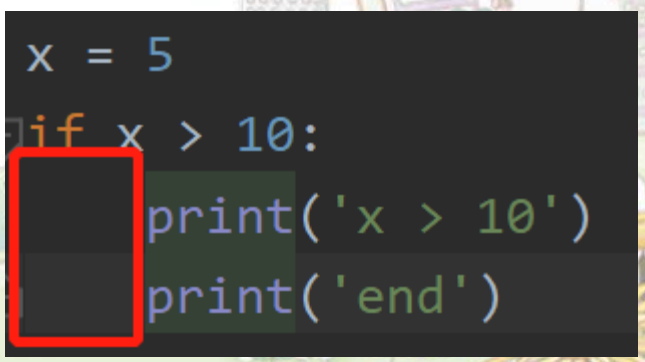
python语法预警

当出现冒号时，说明下一行是一个代码块。

在代码块里面的内容都必须比冒号所在行缩进4个空格。

实际操作中一般使用tab键即可。（pycharm会自动处理）

离开代码块只要在下一行去除该缩进。



lin029011@163.com

LV5

条件

条件嵌套

```
x = 5
if x < 10:
    print('x < 10')
    if x > 4:
        print('x > 4')
```

很多时候可以将上面的条件嵌套用and改写成一层的条件

```
if (x < 100) and (x > 4):
    print('ok')
```

这样就比较简单了。

熟练使用and、or、not运算符，是编写程序的关键。

lin029011@163.com

LV5

条件

添加一个默认的执行语句：else 。

```
if x < 10:
    print('yes')
else:
    print('default')
```

当满足条件时执行yes，不满足时默认执行default。

多条件语句：elif。

elif后面也是条件语句，程序会逐层判断条件执行。

都不满足条件时，执行else的默认语句。

```
if x == 1:
    print('x == 1')
elif x == 2:
    print('x == 2')
elif x == 3:
    print('x == 3')
else:
    print('out of range')
```

lin029011@163.com

LV5

条件

编写一个计算器：用户输入两个整数，并输入运算符，返回结果。

```
x = int(input('输入第一个整数: '))
op = input('输入运算符:')
y = int(input('输入第二个整数: '))
answer = 0
if op == '+':
    answer = x + y
elif op == '-':
    answer = x - y
elif op == '*':
    answer = x * y
elif op == '/':
    answer = x / y
else:
    answer = '错误的运算符'
print(answer)
```

lin029011@163.com

LV5

练习

编写一个个人所得税计算器：

用户输入月薪，程序返回”您需要交税xx元，剩余xx元。”

最新版个人所得税计算方式：
https://jingyan.baidu.com/article/a3761b2b10f4861577f9aa5c.html

lin029011@163.com