



PYTHON程序设计

第一章：Python概述

阿里巴巴商学院

程序设计基础教研组



1.1 下载与安装

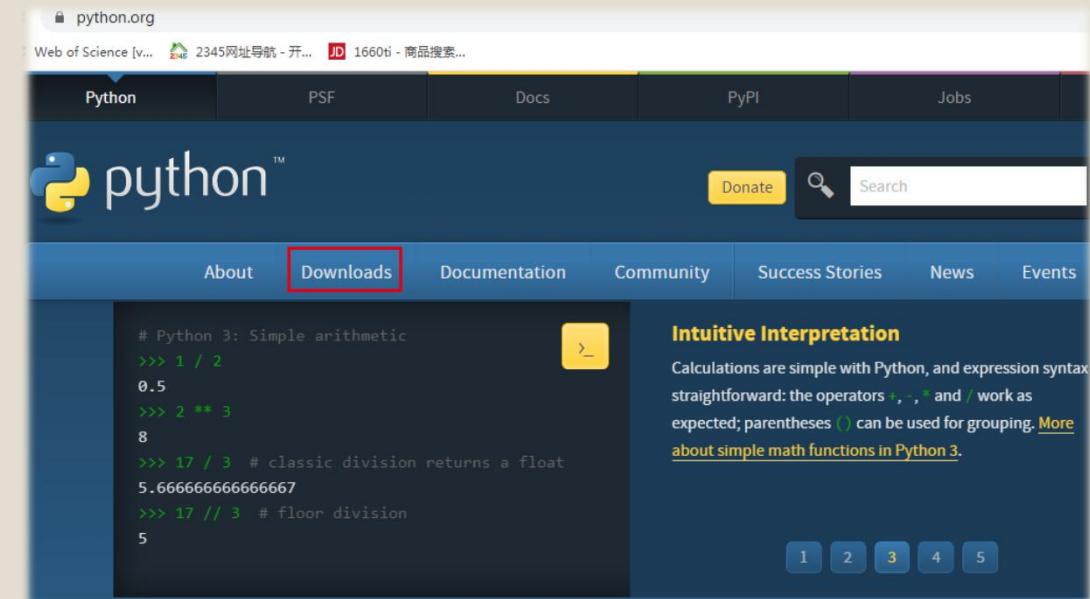
Python采用编译/解释混合方式

- 先编译成字节码，再解释执行

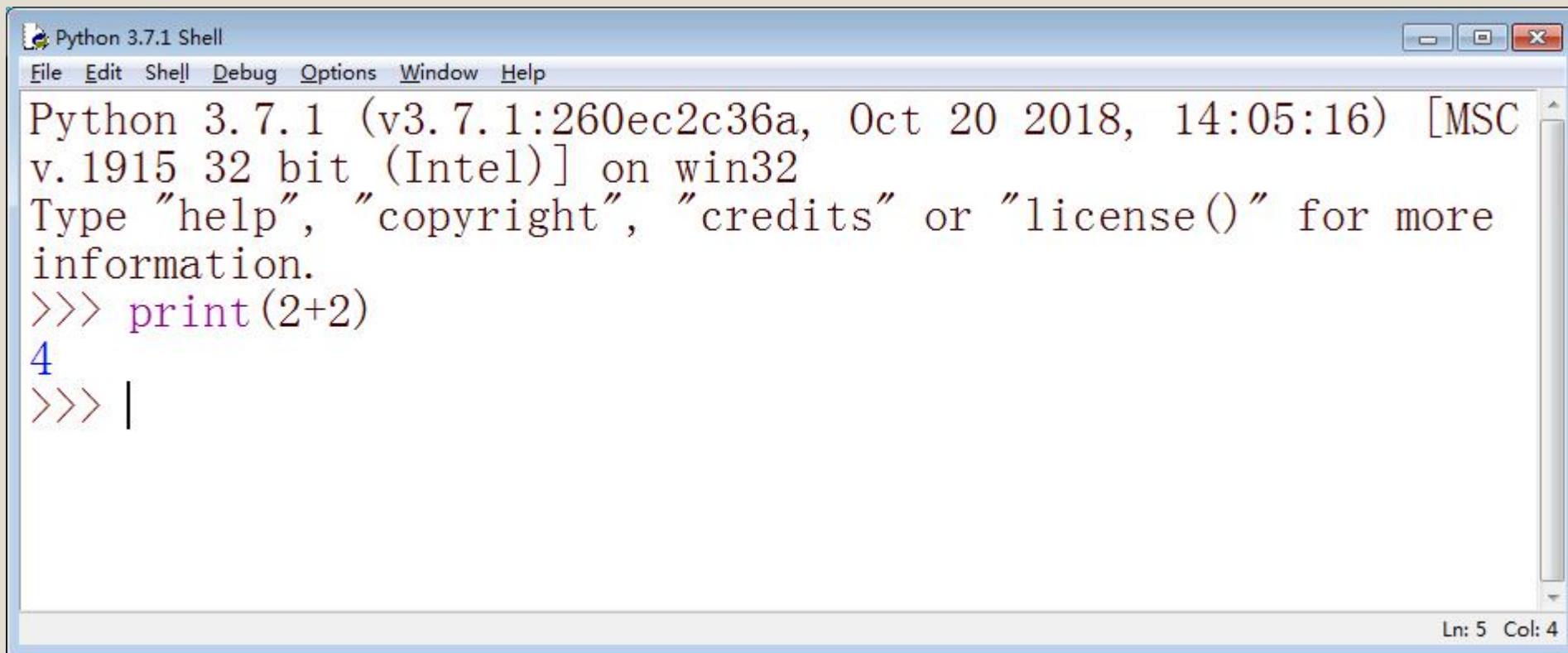
安装Python 3.X

- <http://www.python.org/>
- 下载相应的程序
- 与Python 2.x有不兼容的地方

启动Python



1.2 代码执行：交互式



The screenshot shows the Python 3.7.1 Shell window. The title bar reads "Python 3.7.1 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main window displays the Python startup message:

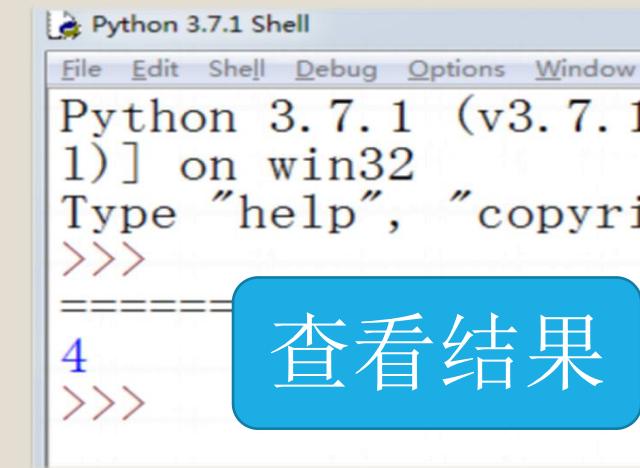
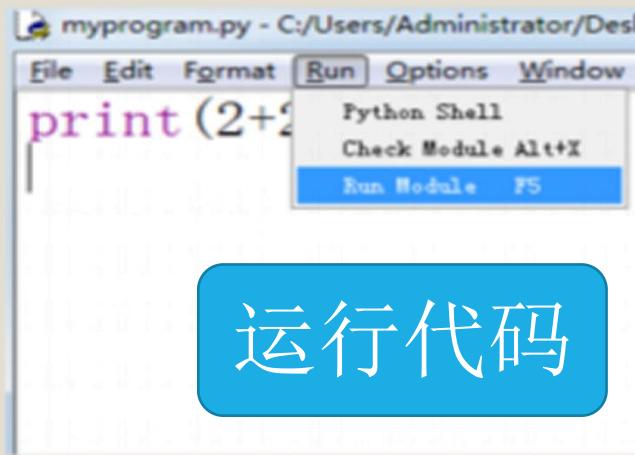
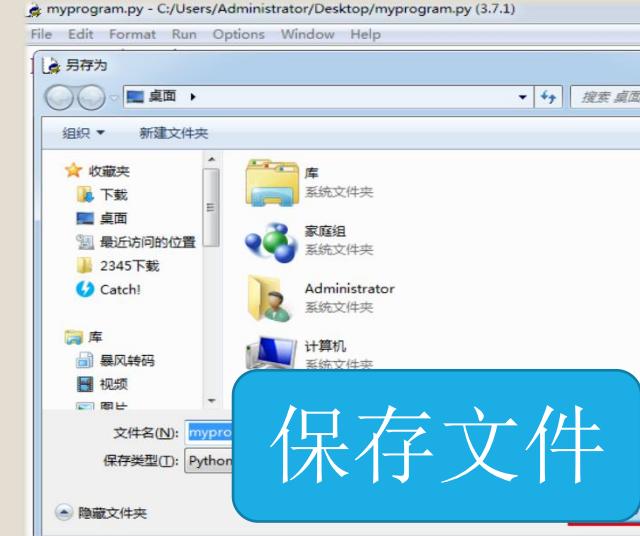
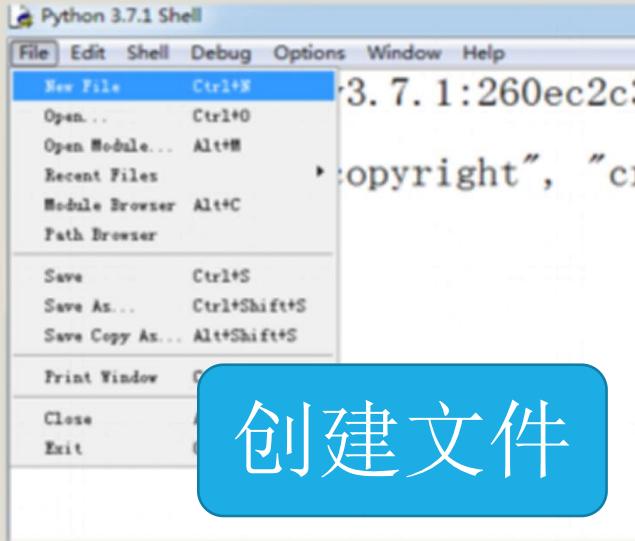
```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v. 1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
```

Below the message, an interactive session is shown:

```
>>> print(2+2)
4
>>> |
```

The status bar at the bottom right indicates "Ln: 5 Col: 4".

1.2 代码执行：脚本式



1.3 基本数据类型:整数

有不同的进制类型

- 默认十进制
- 二进制(0b,0B)
- 八进制(0o,0O)
- 十六进制(0x,0X)

```
>>> 0b11110101  
245  
>>> 0o1234567  
342391  
>>> 0x12E45F  
1238111
```

Python 3.7.X可以表示任意大小整数

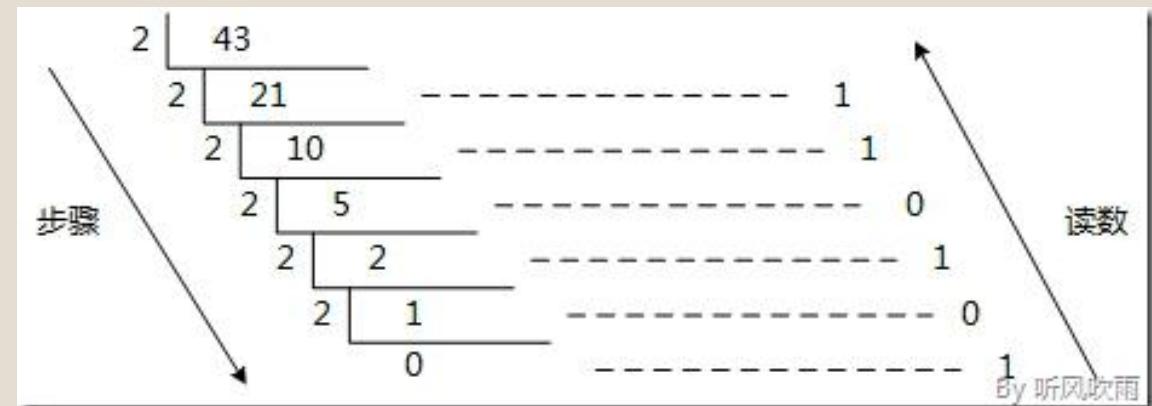
```
>>> 123**123  
11437436793461719009988029522806  
62767462180784518502297758879750  
52369504785666896446606568365201  
54216964997472773062884234534319  
65811348959199428208744498372120  
99476648958359023796078549041949  
00780722062535652692672966406484  
66857583828037071007667402208392  
67
```

1.3 基本数据类型:整数

● 十进制 (Decimal) 转二进制 (Binary) (0-1)

方法: 除2取余法, 即每次将整数部分除以2, 余数为该位权上的数, 而商继续除以2, 余数又为上一个位权上的数, 这个步骤一直持续下去, 直到商为0为止, 最后读数时候, 从最后一个余数读起, 一直到最前面的一个余数。

1. 将商43除以2, 商21余数为1;
2. 将商21除以2, 商10余数为1;
3. 将商10除以2, 商5余数为0;
4. 将商5除以2, 商2余数为1;
5. 将商2除以2, 商1余数为0;
6. 将商1除以2, 商0余数为1;
7. 读数字从最后的余数向前读, 101011, 即 $(43)D=(101011)B$ 。

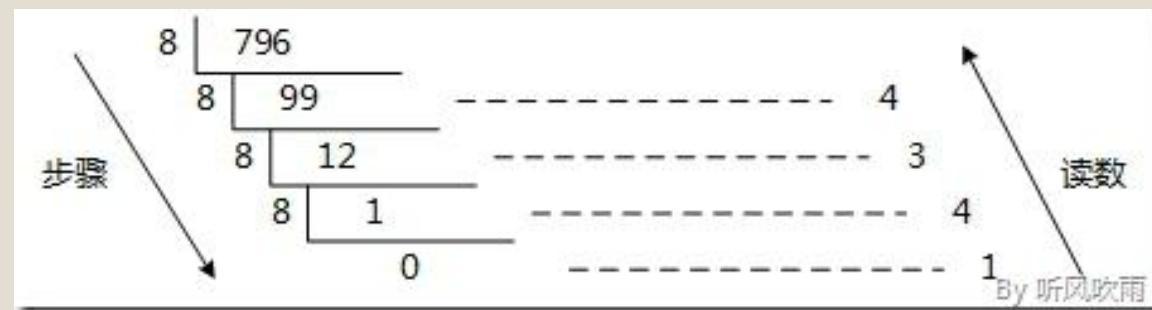


1.3 基本数据类型:整数

● 十进制转八进制 (Octal) (0-7)

方法: 除8取余法, 即每次将整数部分除以8, 余数为该位权上的数, 而商继续除以8, 余数又为上一个位权上的数, 这个步骤一直持续下去, 直到商为0为止, 最后读数时候, 从最后一个余数起, 一直到最前面的一个余数。

1. 将商796除以8, 商99余数为4;
2. 将商99除以8, 商12余数为3;
3. 将商12除以8, 商1余数为4;
4. 将商1除以8, 商0余数为1;
5. 读数字从最后的余数向前读, 1434, 即 $(796)_D = (1434)_O$ 。



1.3 基本数据类型:整数

- 十进制转十六进制 (**Hexadecimal**) (取值范围: **0-9ABCDEF**)

方法: 除16取余法, 即每次将整数部分除以16, 余数为该位权上的数, 而商继续除以16, 余数又为上一个位权上的数, 这个步骤一直持续下去, 直到商为0为止, 最后读数时候, 从最后一个余数起, 一直到最前面的一个余数。

1. 将商796除以16, 商49余数为12, 对应十六进制的C;
2. 将商49除以16, 商3余数为1;
3. 将商3除以16, 商0余数为3;
4. 读数字从最后的余数向前读, 31C, 即 $(796)D=(31C)H$ 。



1.3 基本数据类型:整数

● 二进制、八进制、十六进制转十进制

方法：二进制数从低位到高位（即从右往左）计算，第0位的权值是2的0次方，第1位的权值是2的1次方，第2位的权值是2的2次方，依次递增下去，把最后的结果相加的值就是十进制的值了。

八进制、十六进制转十进制的方法与二进制转十进制类似。

1. 第0位 $1 \times 2^0 = 1;$

2. 第1位 $1 \times 2^1 = 2;$

3. 第2位 $0 \times 2^2 = 0;$

4. 第3位 $1 \times 2^3 = 8;$

5. 第4位 $0 \times 2^4 = 0;$

6. 第5位 $1 \times 2^5 = 32;$

7. 读数，把结果值相加， $1+2+0+8+0+32=43$ ，即
 $(101011)B=(43)D.$

1. 第0位 $3 \times 8^0 = 3;$

2. 第1位 $5 \times 8^1 = 40;$

3. 读数，把结果值相加， $3+40=43$ ，即 $(53)O=(43)D.$

1. 第0位 $B \times 16^0 = 11;$

2. 第1位 $2 \times 16^1 = 32;$

3. 读数，把结果值相加， $11+32=43$ ，即 $(2B)H=(43)D.$

1.3 基本数据类型：整数

可以通过内置函数bin,oct,hex实现10进制转换为其他进制。

```
>>> bin(2021)  
'0b11111100101'  
>>> oct(2021)  
'0o3745'  
>>> hex(2021)  
'0x7e5'
```

通过内置函数int实现其他进制转十进制。

```
>>> int('123', 8)  
83  
>>> int('123', 16)  
291  
>>> int('101', 2)  
5
```

1.3 基本数据类型：浮点数

普通和科学计数法（用 E或e 表示10）

```
>>> 0.1233333445  
0.1233333445  
>>> 1E-3  
0.001  
>>> 1e-5  
1e-05  
>>> 12e3  
12000.0
```

存在不定尾数，有些浮点数无法精确表达

```
>>> 0.1+0.2  
0.3000000000000004  
>>> 0.2+0.3  
0.5  
>>> 0.6+1.2  
1.799999999999998
```

1.3 基本数据类型：复数

复数(有实部和虚部)

```
>>> 1+2j
(1+2j)
>>> 3j
3j
>>> 1+j
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    1+j
NameError: name 'j' is not defined
```

注意：当虚部为1时，1不可省略！

1.3 基本数据类型：字符串

在解释器中高亮显示

```
>>> '123'  
'123'  
>>> "123"  
'123'  
>>> "I'm a boy"  
"I'm a boy"  
>>> 'I\'m a boy'  
"I'm a boy"
```

- 1: 单引号内包含的字符；
- 2: 也可以用双引号；
- 3: 使用分割符'的时候可以用双引号区分，也可以用转义字符\'；
- 4: 字符串允许有空（不含任何字符）。

1.3 基本数据类型：布尔值

布尔值

- `True(1, 非零值, 非空值), False(0, 零值, 空值)`

```
>>> 1+True  
2  
>>> 1-True  
0  
>>> 3*False  
0
```

布尔值可以直接参与运算
`True`相当于1
`False`相对于0

1.3 基本数据类型：空值

空值 None

```
>>> 3*None
Traceback (most recent call last):
  File "<pyshell#15>", line 1, in <module>
    3*None
TypeError: unsupported operand type(s) for *: 'int' and 'NoneType'
```

空值不能直接参与运算！

1.4 变量：变量的创建

一个数据在计算机内需要一个对应的内存空间，每个内存空间存在一个地址。通过地址程序可以访问内存中的数据。

```
>>> x = 100  
>>> y = 1000
```

变量名和变量的地址进行关联，从而可以通过变量名来访问数据。

```
>>> x  
100  
>>> y  
1000
```

1.4 变量：变量的命名规则

命名规则：

- 包含合法的字符：字母， 数字和下划线
- 不能以数字开头
- 不能与python关键字重复
- 大小写敏感

and	as	assert	break	class	contin e
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			

附表： **Python**关键字

1.5 操作符：运算操作符

+ : 加

- : 減

* : 乘

**: 乘方

/ : 除

//: 地板除

%: 向左取余

```
>>> 1 + 5/2 + 5**2 + 2*3 + 5 // 2  
36.5  
>>> 7%2  
1  
>>> -7%2  
1  
>>> 5.2%2  
1.2000000000000002  
>>> -5.2%2  
0.7999999999999998
```

1.5 操作符：比较操作符

<

>

<=

>=

==

!=

<>

```
>>> 5==2
```

```
False
```

```
>>> 5<2
```

```
False
```

```
>>> 5>2
```

```
True
```

```
>>> 5!=2
```

```
True
```

```
>>> 5<>2
```

```
True
```

1.5 操作符：逻辑操作符

and

not

or

```
>>> 5<2 and 4>6  
False  
>>> 5<2 and 4<6  
False  
>>> 5<2 or 4<6  
True  
>>> not 5<6  
False  
>>>
```

1.5 操作符：身份和成员关系

身份运算符：is 或 is not

```
>>> x = -5
>>> y = -5
>>> x is y
True
>>> a = 256
>>> b = 256
>>> a is b
True
>>> e = 257
>>> f = 257
>>> e is f
False
```

成员关系运算符：in 或 not in

```
>>> '1' in '123'
True
>>> 1 in [1, 2, 3]
True
>>> 1 in (1, 2, 3)
True
>>> 1 in {1, 2, 3}
True
>>> 1 in {1:2, 2:5}
True
```

1.6 语句和表达式

表达式是某事，而语句就是做某事。

```
>>> 2+2  
4  
>>> print(2+2)  
4
```

在交互式解释器中执行上述两行代码，结果都是一样的。交互解释器会把所有表达式的值输出。

```
>>> x=2  
>>> y=3  
>>> x+y  
5  
>>> z=x+y
```

语句改变了事物，但没有返回值，也不会有输出。

算术表达式

- 包含各种算数运算符的计算表达式

```
>>> 2+3*5**2-10  
67  
>>> (2+3)*5**2-10  
115  
>>> 2+3*5%4/2  
3.5
```

赋值语句

- 赋值运算符=，增强赋值运算符(部分位运算符[不包含取反]和全部算数运算符加 =，之间不能有空格)，如+=， /=

```
>>> x = 5  
>>> x += 5  
>>> x  
10  
>>> x *= 5  
>>> x  
50
```

1.7 函数

函数就好像就可以用来实现特定功能的小程序一样。

使用函数的方式叫做调用，调用的时候需要提供参数。

函数调用的可以看成另外一种表达式。

```
>>> 2**3
8
>>> pow(2, 3, 5)
3
>>> 2**3%5
3
>>> abs(-5)
5
>>> abs(3+4j)
5.0
>>> round(3.14159265, 3)
3.142
```

常用内置函数

- bool(),chr()分别表示转成布尔值， ASCII码转单字符字符串
- complex() 表示转成复数， 包含类似 ‘1+2j’的字符串
- float(),int()转成浮点数或整数
- str(),ord()转字符串和ASCII码

```
>>> bool('o')
True
>>> bool('')
False
>>> bool(0)
False
>>> chr(65)
'A'
>>> chr(0)
'\x00'
```

```
>>> complex(5)
(5+0j)
>>> complex('5')
(5+0j)
>>> complex('5+1j')
(5+1j)
>>> float('5')
5.0
>>> float(5)
5.0
>>> int(3.14)
3
```

```
>>> bin(65)
'0b1000001'
>>> oct(65)
'0o101'
>>> hex(65)
'0x41'
>>> str(4.124)
'4.124'
>>> ord('a')
97
```

1.8 简单输入与输出

有时候程序需要输出结果(`print`)，有时候也需要获取数据(`input`)

```
>>> print('123456', end="结束")
123456结束
>>> x = input("请输入一个数字: ")
请输入一个数字: 123
>>> print(x)
123
>>> y = input("请输入一个字符串: ")
请输入一个字符串: abc
>>> print(y)
abc
```

`print()`将参数值打印出来，可选参数`end`表示结束标志符

`input()`获取用户输入，并转换为字符串返回

1.9 模块

1.导入模块

- import module_name

2.访问模块函数或者变量

```
>>> from math import sin  
>>> sin(4)  
-0.7568024953079282  
>>> from math import e  
>>> e  
2.718281828459045  
>>> import math  
>>> math.cos(2)  
-0.4161468365471424  
>>> math.pi  
3.141592653589793
```

1.10 简单条件语句

```
if V==102 and V>=98:  
    print('合格')  
else:  
    print('不合格')
```

判断某种条件是否成立。
牢记冒号，缩进，对齐三原则



1.11 简单循环语句

while 循环

- while后面表达式为真时，持续执行循环体内语句

```
>>> result = 0  
>>> x = 1  
>>> while x<=5:  
    result += x  
    x += 1  
  
>>> print(result, x)  
15 6
```

0		result=0; x = 1
1	result += x x += 1	result=1; x = 2
2	result += x x += 1	result=3; x = 3
3	result += x x += 1	result=6; x = 4
4	result += x x += 1	result=10; x = 5
5	result += x x += 1	result=15; x = 6

牢记冒号，缩进，对齐三原则

1.14 注意事项

写程序基本建议--认真！多练！几点具体避坑建议：

- 1：避免拼错标志符，如变量名，函数，语句等
- 2：避免使用中文符号，如引号，逗号，括号等
- 3：左边一个引号和右边的引号一定有对应的匹配（括号也一样）
- 4：注意书写格式（冒号，缩进，对齐）

1.15 体质指数的代码实现

```
#BMI指数计算
```

```
Height = int(input("请输入身高(m):"))
Weight = int(input("请输入体重(kg):"))

BMI = Weight/Height**2

if BMI>=23.9:
    print("BMI指数为", BMI, "体质偏重")
elif BMI<=18.5:
    print("BMI指数为", BMI, "体质偏轻")
else:
    print("BMI指数为", BMI, "正常")
```

谢 谢 !