

PYTHON程序设计

第一章：Python概述

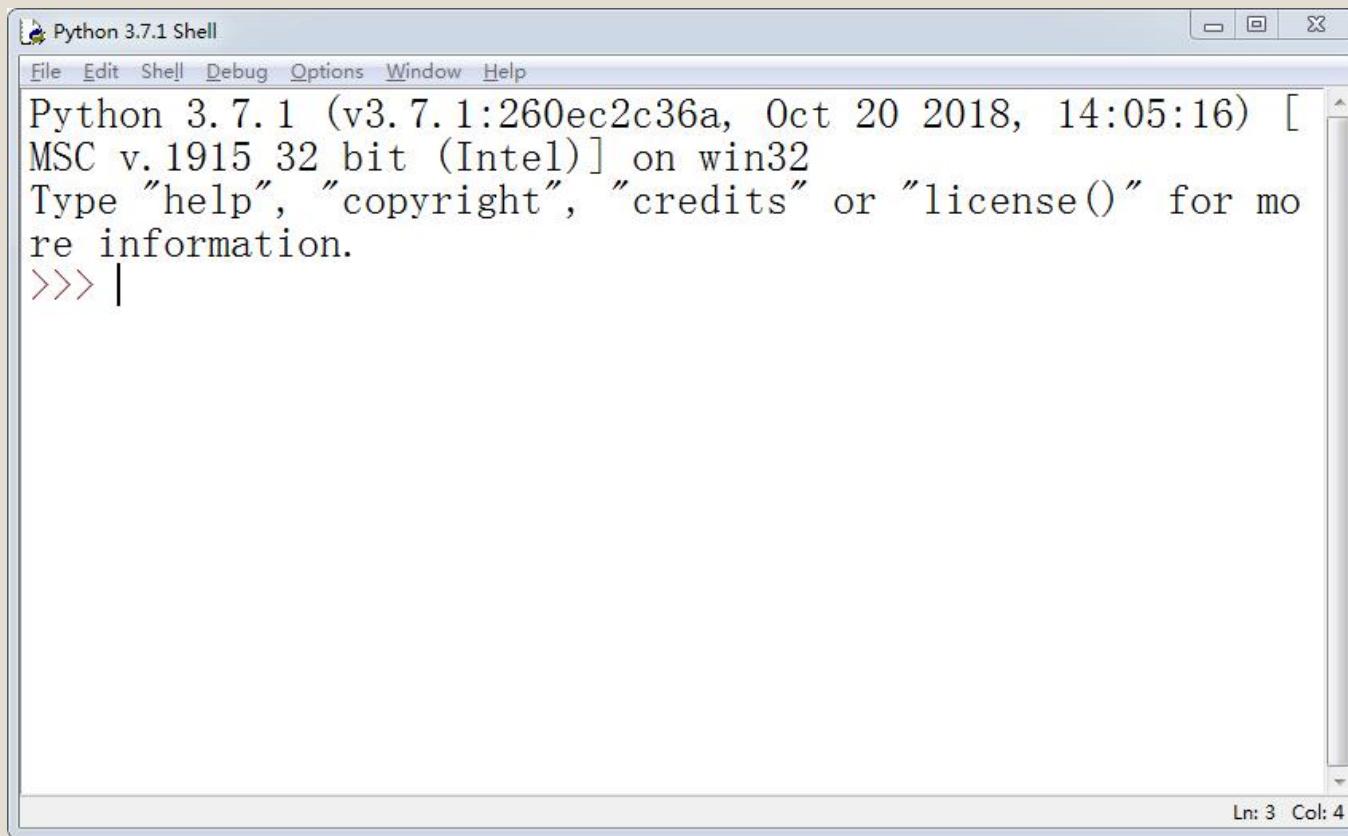
阿里巴巴商学院

程序设计基础教研组

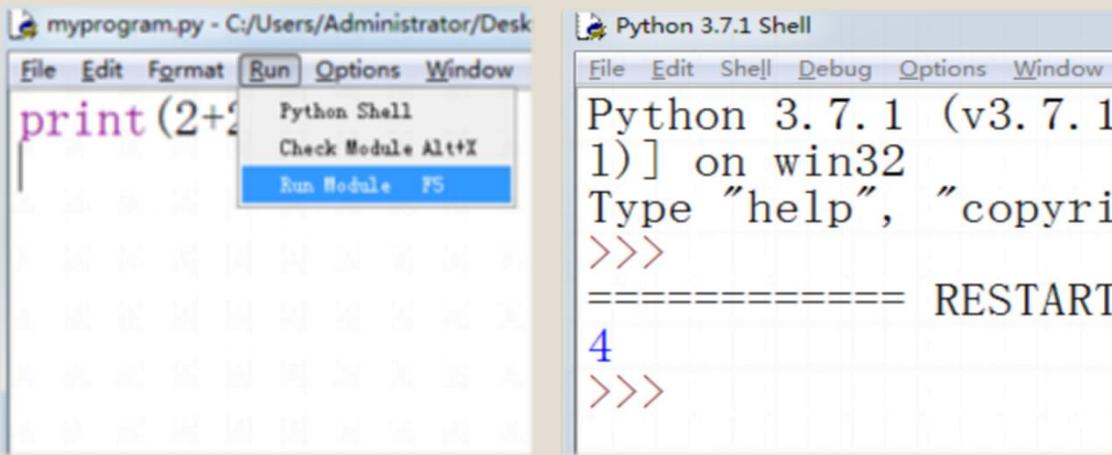
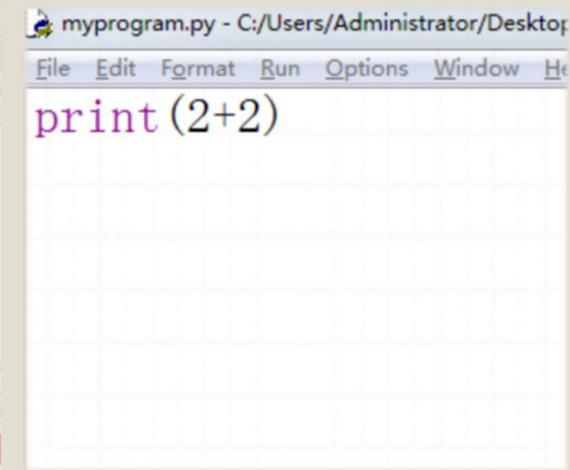
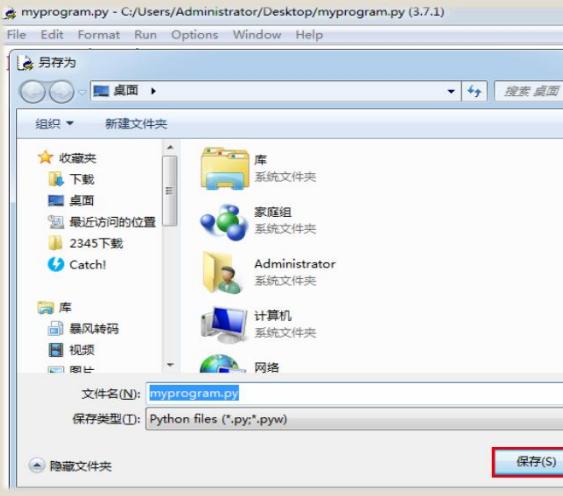
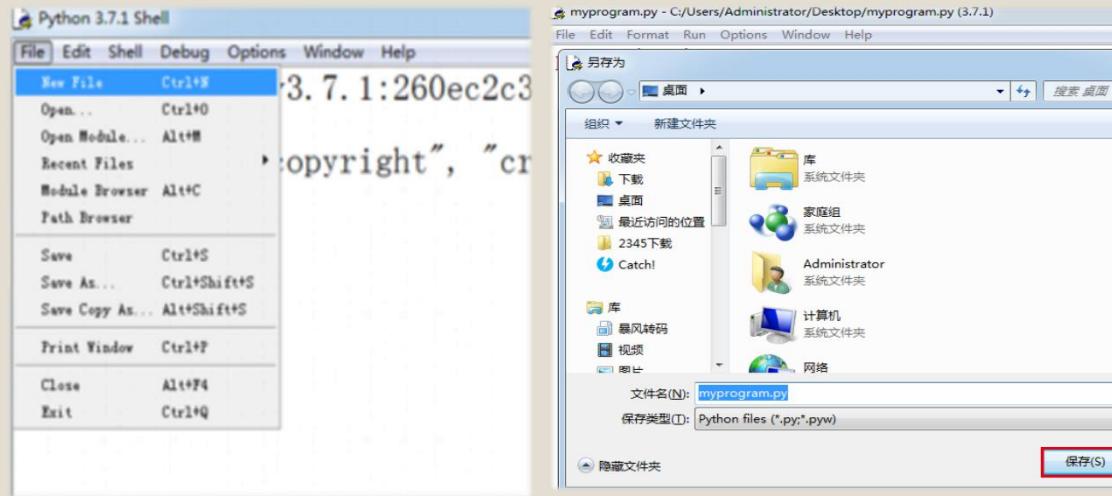
1.1 下载与安装

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

1.2 代码执行：交互式



1.2 代码执行：脚本式



- 1：创建文件
- 2：保存文件
- 3：写入代码
- 4：运行代码
- 5：查看结果

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)
- Python 3.7.X可以表示任意大小整数

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

```
>>> 123**123  
1143743679346171900998802  
9522806627674621807845185  
0229775887975052369504785  
6668964466065683652015421  
6964997472773062884234534  
3196581134895919942820874  
4498372120994766489583590  
2379607854904194900780722  
0625356526926729664064846  
6857583828037071007667402  
20839267  
>>>
```

1.3 基本数据类型：整数

- 可以通过内置函数bin,oct,hex实现10进制与其转换

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

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  
0.30000000000000004  
>>> 0.2+0.3  
0.5  
>>> 0.6+1.2  
1.7999999999999998
```

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，也可以用转义自符\'；
- 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

1.3 基本数据类型：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	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			

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

+ =

/ =

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！几点具体建议：

- 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")
```

謝 謝 !