

# Python数据分析

(1)

殷传涛 教授  
chuantao.yin@buaa.edu.cn

# 课程说明

- ▶ Python数据分析
- ▶ 共四次课
  - 请准时上课
  - 每次：2\*（2节讲授+2节实践练习）
  - 携带个人计算机，方便进行实践
- ▶ 考核方式：大作业
- ▶ 参考教材：
  - 《Python数据分析》 阿曼多-凡丹戈著（不建议中文版）
  - 课程的PPT讲义，会分享在课程微信群
- ▶ 交流方式：
  - 微信群、邮件或通过大班长都可以



# 课程说明

## ▶ 学生背景

- Python零基础与入门者（85%）：我带你们
- Python熟练者（10%）：自由飞翔
- Python精通者（5%）：带我带我

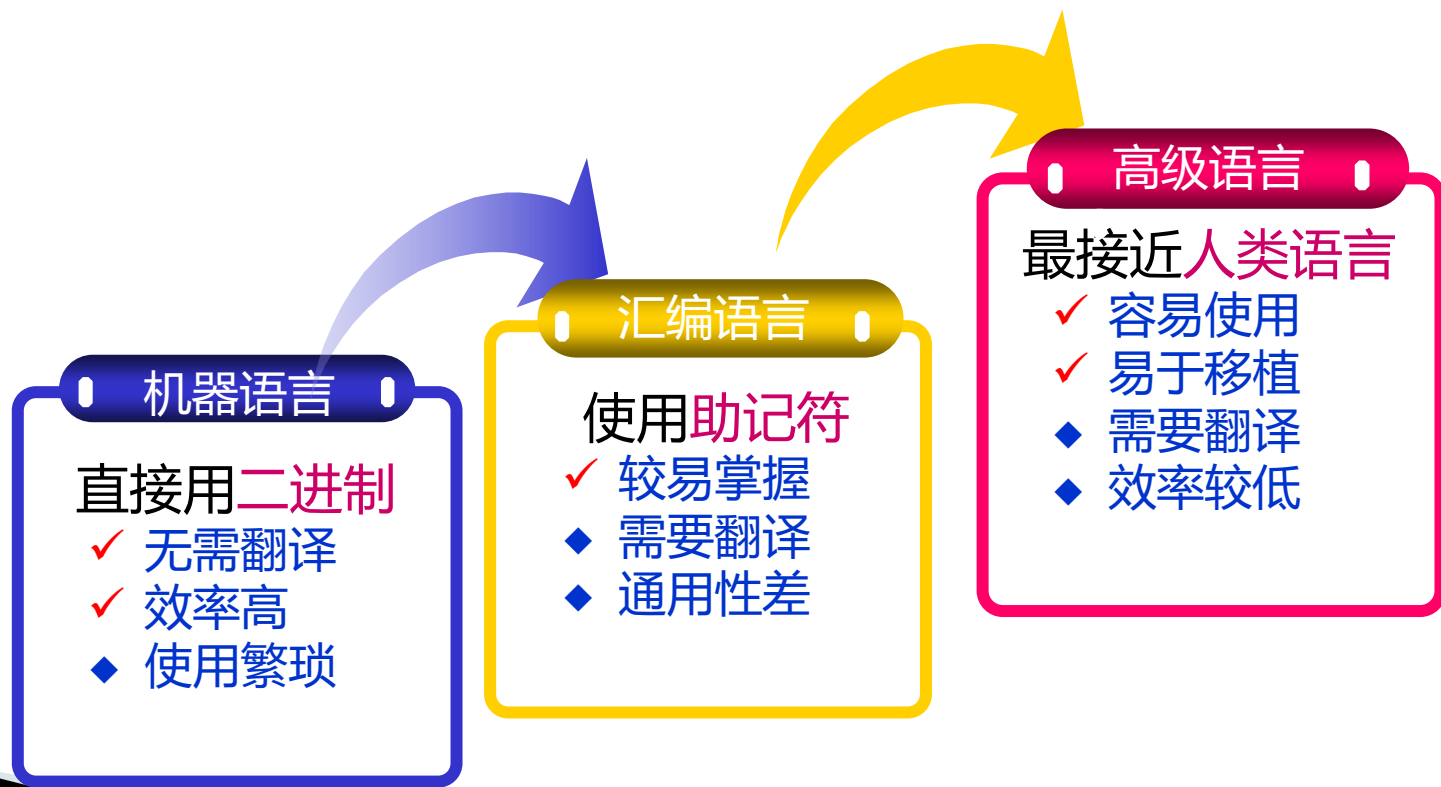
## ▶ 主要内容

- Python入门与基础
- Numpy
- Pandas
- Matplotlib
- 数据分析案例与实践

# 一、Python语言入门与基础

# 程序设计语言

- ◆ **程序设计语言** (Programming Language) 是用于书写计算机程序的语言。它是软件的基础和组成



# Python

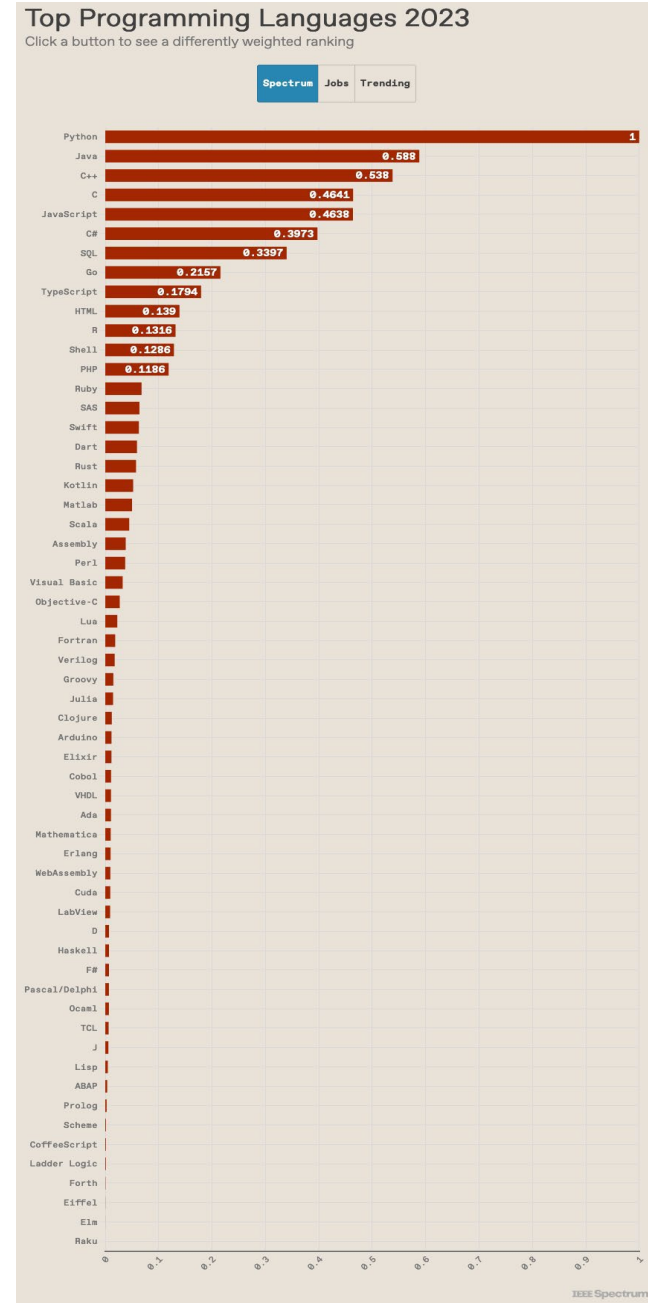
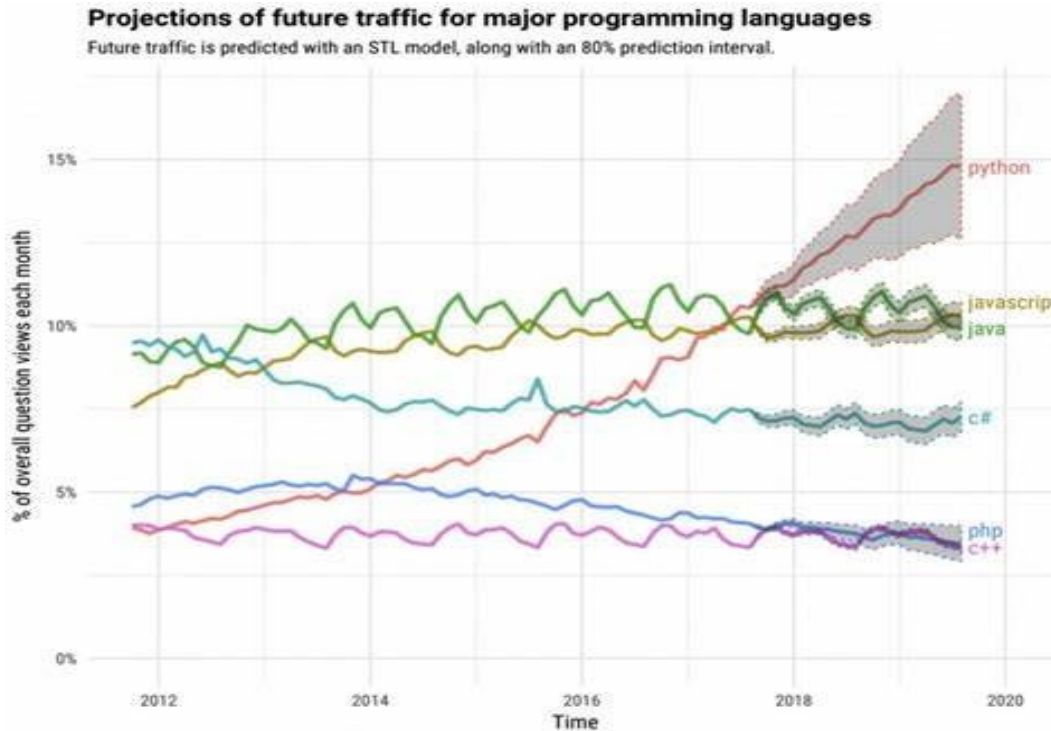
- ▶ **Python: 面向对象**的高级计算机语言
  - 创始人: Guido Van Rossum
  - 1989年发明, 目前最新版本3.12.3 (2024.4.9)

吉多·范罗苏姆  
Guido van Rossum



# Python

- ▶ 2017年成为最受欢迎的编程语言





# Python

- ▶ Python主要用户
  - 数据科学家/分析师
  - 安全工程师
  - 云计算工程师
  - 大数据工程师
  - 科学家/研究员

- ▶ AI领域首选语言



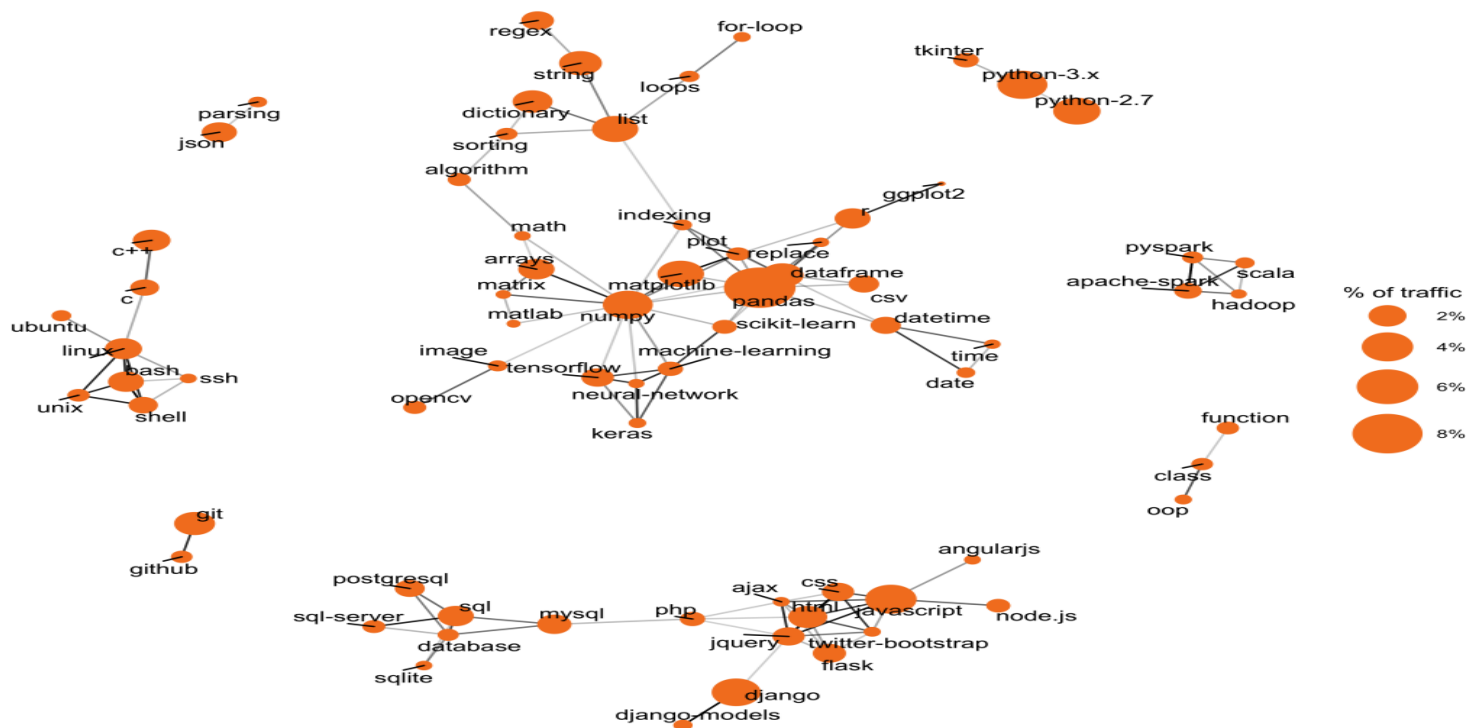


# Python

## Python相关标签网络

### Network of Correlated Tags Visited by Python Visitors

Connections are between tags with a greater than .2 Pearson correlation in visits across Python visitors, which are defined as someone with  $\geq 50$  total visits whose most visited tag is Python.



# Python

## ▶ 特点

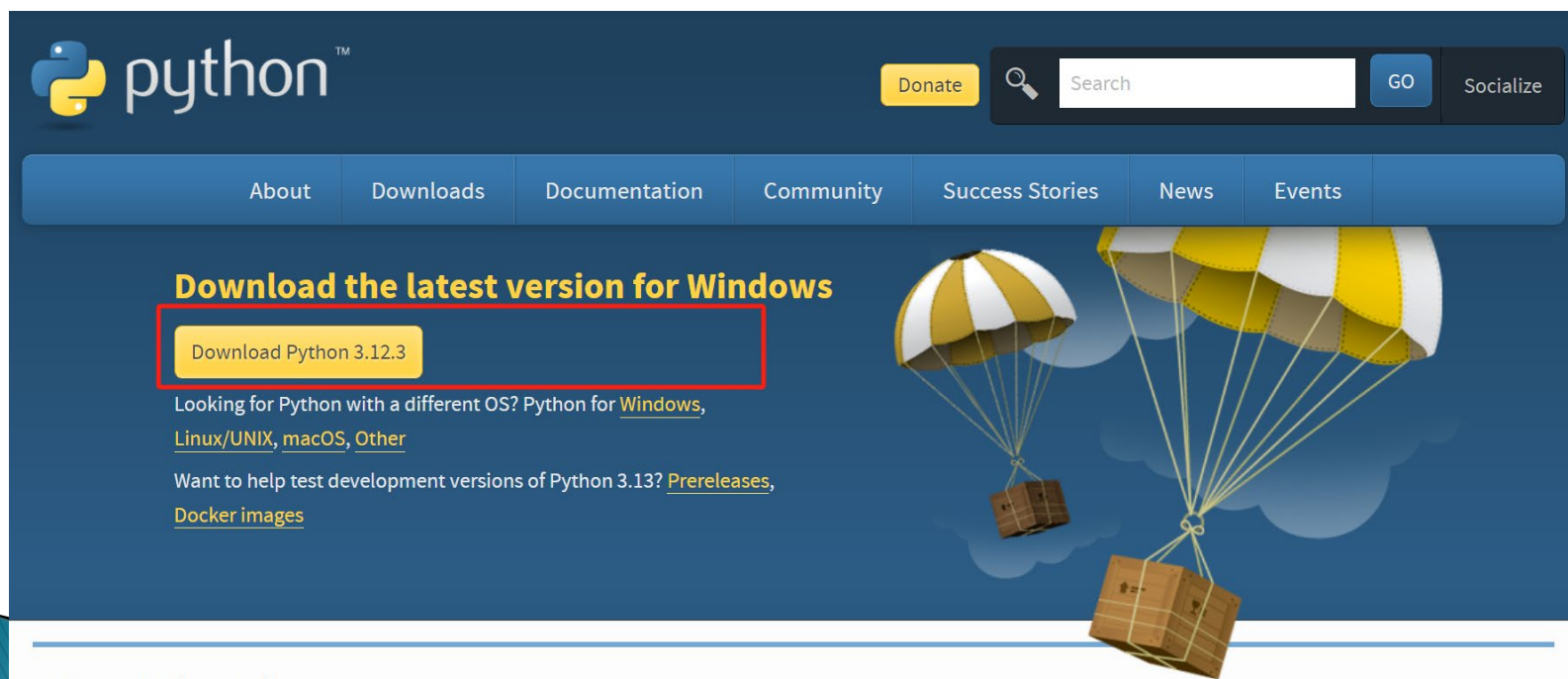
- 简单易懂
  - 提供交互环境
  - 语法简洁
    - 高级数据结构简洁地表达复杂的操作
    - 语句组织依赖于缩进
    - 参数或变量不需要声明
- 功能强大
  - 网络互联、图形处理、科学计算、实时控制等
  - 跨平台
  - 面向对象
- 开发快速
  - 内建的数据结构，适合于快速应用开发
  - 支持模块和包，鼓励程序和代码重用

"Life is short  
(You need Python)"  
-- Bruce Eckel  
ANSI C++ Comittee member



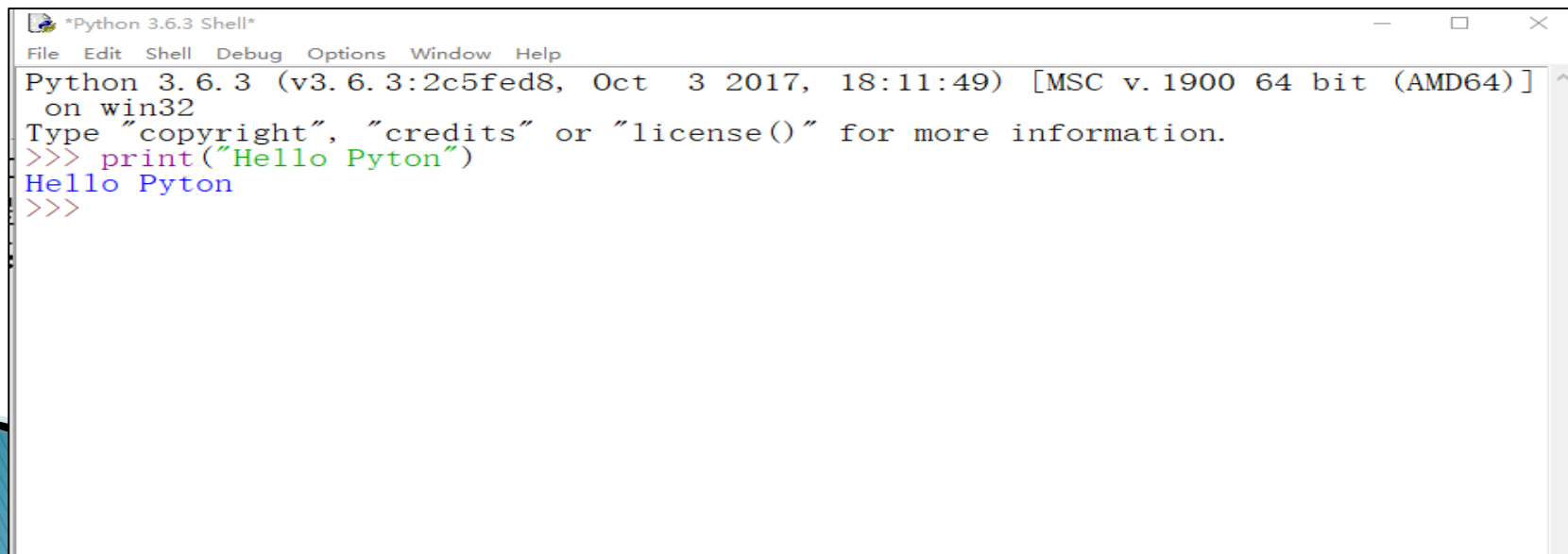
# Python的安装

- ▶ 官网: <https://www.python.org>
- ▶ 最新版本: Python 3.12.3 (2.X与3.X有所区别)
- ▶ 下载安装正确的版本 (系统, 位数, 安装方式)



# IDLE

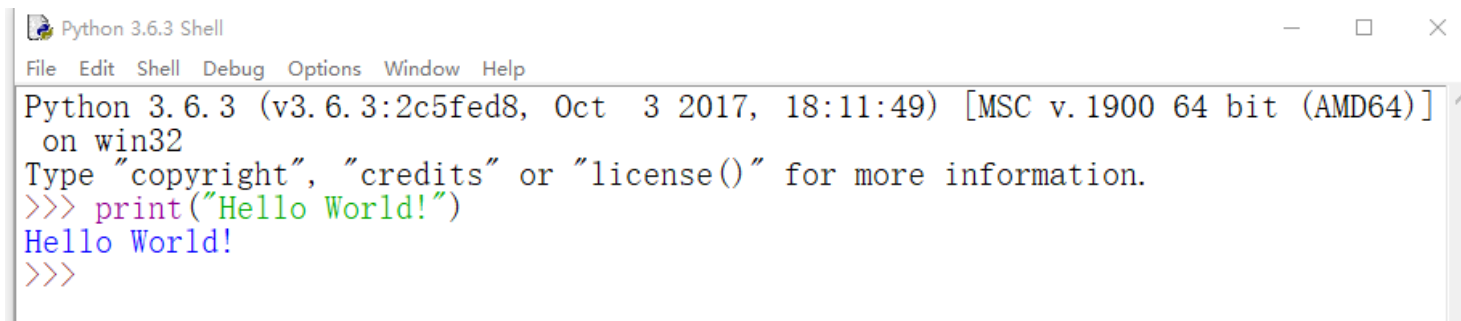
- ▶ **IDLE** : Integrated Development and Learning Environment, Python自带的集成开发环境
  - **IDLE的交互界面窗口 (shell)**
  - 编写和调试程序
  - 语法词汇高亮显示
  - 创建、编辑程序文件



```
*Python 3.6.3 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello Python")
Hello Python
>>>
```

# 第一个python语句

在shell窗口里输入: `print("Hello World")`



```
Python 3.6.3 Shell
File Edit Shell Debug Options Window Help
Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello World!")
Hello World!
>>>
```

- ▶ **命令提示符:** `>>>` , 等待指令
- ▶ **`print()`:** 打印输出函数, 完成将数据打印在屏幕上的功能
- ▶ **字符串:** `" "` , 表示字符集合的一种数据类型, 可以用 `+` 进行连接

# 第一个python语句



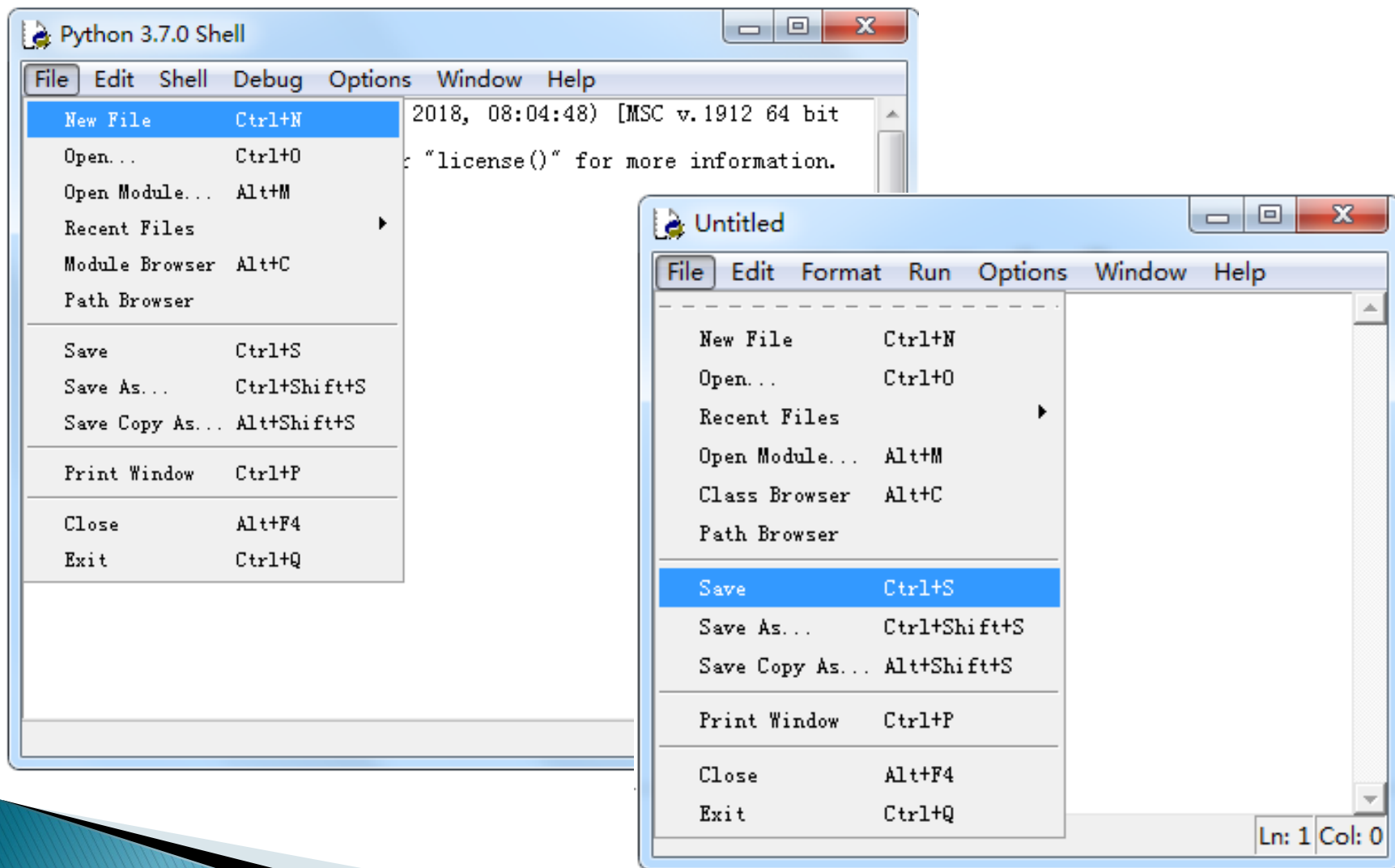
```
*Python 3.6.3 Shell*
File Edit Shell Debug Options Window Help
Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello World!")
Hello World!
>>> print("AAA")
AAA

>>> print("I will tell you a story")
I will tell you a story

>>> print("Long long ago,")
Long long ago,
>>> print("There is a beautiful girl")

Python 3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> a=3
>>> b=4
>>> a+b
7
>>> 256*487
124672
>>> 10**2
100
>>>
```

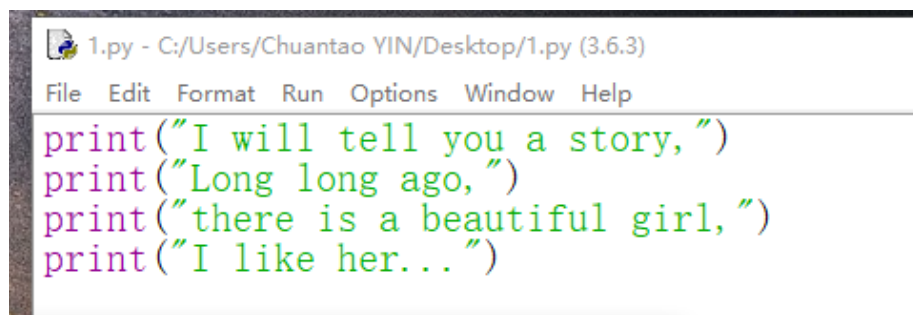
# 创建python源程序





# 创建python源程序

- ▶ 创建一个python源程序的文件
  - 在IDLE中,工具栏-文件-新文件 (ctr+N)
  - 弹出IDLE的**文件编辑器**
  - 在**文件编辑器**中, 输入多行代码
  - 存储文件为\*.py的python源代码文件



```
1.py - C:/Users/Chuantao YIN/Desktop/1.py (3.6.3)
File Edit Format Run Options Window Help
print('I will tell you a story,')
print('Long long ago,')
print('there is a beautiful girl,')
print('I like her...')
```

```
===== RESTART: C:/Users/Chuantao YIN/Desktop/1.py =====
I will tell you a story,
Long long ago,
there is a beautiful girl,
I like her...
>>>
```

# 文件编辑器

## ▶ IDLE的文件编辑器

- 能够自由地编辑多行代码
- 以文本的方式编辑代码
- 代码可以存储为一个程序源文件
- 可以打开一个程序源文件，进行修改和再次编辑
- 可以一次性地运行代码（Run-Run Module, F5）
- 运行结果在shell中显示

# 语法

- ▶ **语法**：编写编程语言的规则集合
- ▶ **语法错误**（bug）：不符合语法的语句而导致错误
- ▶ **调试**（debug）：找到，并且修复bug的过程
- ▶ 常见的语法错误：
  - 关键字拼写错误
  - 大小写错误
  - 标点符号错误
  - 空格错误

```
prnt("I will tell you a story,")  
Print("Long long ago,")  
print"there is a beautiful girl,"  
print("I like her...")
```

# 对象

- ▶ **对象**是Python程序操作的核心内容，每个对象具有类型的定义，用于说明程序可用该类型对象完成的工作
- ▶ 对象的类型可以是标量以及非标量
  - **标量对象**：单个值或数据元素，不能进行分解，如整数等
  - **非标量对象**：多个值或元素的集合组成，如列表等

# 标量对象

- ▶ Python含有四类标量对象：
  - 整数 (int) : 16, -5, 1024;
  - 浮点数 (float) : 16.0, -5.25, 1024E3 ( $=1024*10^3$ ) ;
  - 布尔值 (bool) : True或False;
  - 字符串 (str) : “hello” , “GOOD”;

# 变量

```
>>> print("this is a girl")
this is a girl
>>> print(3)
3
>>> print(3+5)
8
>>> x="this is a girl"
>>> print(x)
this is a girl
>>> x=3
>>> print(x)
3
>>> x=3+5
>>> print(x)
8
```

```
>>> x=3+5
>>> print(x)
8
>>> y=6
>>> z=x+y
>>> print(z)
14
```

x,y,z是一个变量

# 变量

## ▶ 变量

- 引用一个数据对象，通过赋值语句将变量名与值关联起来
- 变量名：
  - 可以包括字母，数字和下划线（数字不能开头）
  - 不能包括除下划线之外的符号
  - 不能是python保留的关键字
  - 大小写有区分
    - x, X, y, yourNumber, your\_Number



# 变量

## ▶ python中保留的**关键字**

|          |         |        |
|----------|---------|--------|
| and      | exec    | not    |
| assert   | finally | or     |
| break    | for     | pass   |
| class    | from    | print  |
| continue | global  | raise  |
| def      | if      | return |
| del      | import  | try    |
| elif     | in      | while  |
| else     | is      | with   |
| except   | lambda  | yield  |

# 对变量进行赋值

- ▶ **变量赋值**：利用**等号“=”**把一个数据对象(object)赋给一个变量名(variable name)
  - $x=5$ ,  $x=10.3$ ,  $x=\text{"this is a long long story..."}$
  - $x=y$ ,  $a=b=c=1$
  - $\text{pi}=3.14159$
  - $\text{radius}=10.5$
  - $\text{circumference}=2 * \text{pi} * \text{radius}$
  - $\text{radius}=12.6$
- ▶ **注意**：必须**先赋值，再计算**
- ▶ **注意**：等号“=”是一个动作，不是代表两边相等

# 赋值语句

- ▶ `x,y = 1,2`
- ▶ `print ('x=',x)`
- ▶ `print ('y=',y)`
  
- ▶ `x,y = y,x`
- ▶ `print ('x=',x)`
- ▶ `print ('y=',y)`
  
- ▶ 注意：分隔符只能是英文半角逗号

# 合理使用变量名以增强程序可读性

- ▶  $x = 3.14159$
- ▶  $y = 10.5$
- ▶  $z = 2 * x * y$

尽可能使用英语（拼音）使变量具有可读性意义

- ▶  $pi = 3.14159$
- ▶  $radius = 10.5$
- ▶  $circumference = 2 * pi * radius$

# 基本数据类型

- ▶ 数字 (Number)
- ▶ 字符串 (String)

# 数字

## ▸ 数字的类型

- 整数 (**int**): 0, 23, -33
- 浮点数(**float**): 2.3, -0.1, 2E+3
- 布尔型(**bool**): True和False (1与0)
- 复数(**complex**): 5+3j, complex(5,3)

## ▸ 可由type( )和isinstance( )来判断数字类型

```
>>> type(5)
<class 'int'>
>>> type(5.0)
<class 'float'>
>>> type(True)
<class 'bool'>
>>> type(5.0+3.0j)
<class 'complex'>
```

```
>>> x=5
>>> y=4
>>> type(x)
<class 'int'>
>>> type(x/y)
<class 'float'>
>>>
```

```
>>> isinstance(5, int)
True
>>> isinstance(5, float)
False
>>> isinstance(5, str)
False
```

# 数字类型转换

- ▶ **int(x)**将x转换为一个整数，以**截尾**的方式
  - 利用**round(x)**可将x进行四舍五入
- ▶ **float(x)**将x转换为一个浮点数
- ▶ **complex(x)**将x转换为一个复数  $x+0j$
- ▶ **complex(x,y)**将x和y转换为 $x+yj$

```
>>> int(3.1)
3
>>> int(3.5)
3
>>> int(3.9)
3
>>> int(3.9999999999)
3
>>> int(3.9999999999999999999999999999)
4
>>> float(3.9999999999999999999999999999)
4.0
>>> 3.9999999999999999999999999999
4.0
>>> complex(3, 4)
(3+4j)
>>> complex(3)
(3+0j)
```



# 数字的计算

## ▶ 由表达式进行计算

- 表达式由对象和运算符组合而成
  - 对象是被创建的具有实际数据类型的一个数据
  - 对象可以赋值给变量，对象实际存在，变量只是标识
  - 变量在使用前必须定义（赋值）
- 表达式被创建后，具有自己的值，也是对象
  - 例：表达式 $4+5$ 表示int类型的对象9
  - 例：表达式 $4.0+5.0$ 表示float类型的对象9.0

# 算数运算符

以下假设变量a为10，变量b为21：

| 运算符 | 描述                        | 实例                              |
|-----|---------------------------|---------------------------------|
| +   | 加 - 两个对象相加                | a + b 输出结果 31                   |
| -   | 减 - 得到负数或是一个数减去另一个数       | a - b 输出结果 -11                  |
| *   | 乘 - 两个数相乘或是返回一个被重复若干次的字符串 | a * b 输出结果 210                  |
| /   | 除 - x 除以 y                | b / a 输出结果 2.1                  |
| %   | 取模 - 返回除法的余数              | b % a 输出结果 1                    |
| **  | 幂 - 返回x的y次幂               | a**b 为10的21次方                   |
| //  | 取整除 - 返回商的整数部分            | 9//2 输出结果 4 , 9.0//2.0 输出结果 4.0 |

int类型与float类型混合运算的结果为float类型

# 逻辑运算符

## ▶ 逻辑运算符

- **and**, x and y, 进行逻辑运算, 若为真则返回最后一个真值
- **or**, x or y, 进行逻辑运算, 若为真则返回第一个真值
- **not**, not x, 进行逻辑运算, 返回True 或 False

```
>>> 1 and 0
0
>>> 1 or 0
1
>>> not 1
False
>>> not 0
True
>>> True and False
False
>>> 10 and 20
20
>>> 10 or 20
10
>>> bool(10)
True
```

# 比较运算符

- ▶ 比较运算符，返回True或False
  - ==, != 等于，不等于
  - <, > 小于，大于
  - >=, <= 大于等于，小于等于

```
>>> x=3;y=4
>>> x==y
False
>>> x!=y
True
>>> y!=y
False
>>> x>y
False
>>> x<=y
True
```

# 更多数字计算

- ▶ 使用python中的内置math模块
  - import math, 导入math模块
  - 使用math模块中定义的函数

```
>>> import math
>>> dir(math)
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acosh',
'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign', 'cos', 'cosh',
'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'floor', 'fmod',
'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf',
'isnan', 'ldexp', 'lgamma', 'log', 'log10', 'log1p', 'log2', 'modf', 'nan',
'pi', 'pow', 'radians', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc']
>>> math.pi
3.141592653589793
>>> math.pow(2, 3)
8.0
>>> math.sqrt(25)
5.0
>>> math.floor(1.8)
1
>>> math.ceil(1.8)
2
>>> math.sin(2)
0.9092974268256817
```

# 数学运算的优先级

- ▶  $x + y * 3$ 
  - 首先计算  $y * 3$ ，之后加上  $x$
- ▶ 通过括号可以改变优先顺序
  - $(x + y) * 3$ ，将首先计算  $x + y$ ，结果值再乘以 3
- ▶ 注意：没有中括号、没有大括号、只有小括号
- ▶ 注意：没有全角括号、中文括号、只有半角英文小括号

# 文本型数据的表示

- ▶ 计算机应用
  - 科学计算
  - 信息管理
- ▶ 信息管理中大量的数据都是文本数据
  - 如：姓名、地址、简历等等
  - 小测验：身份证号码、电话号码等是数值？
- ▶ 计算机中用字符串来表示文本数据
- ▶ 字符串类型str： **字符序列**



# 字符串

'hello world'

"~!@#\$%^&\*"

"汉字也是字符"

"line one,  
line two"

"""line one,  
line two"""

'hello world'

"~!@#\$%^&\*"

"汉字也是字符"

'line one,  
line two'

"""line one,  
line two"""

# 字符串

- ▶ 由双（单）引号来创建
  - 例：'hello, world' “good! @#” “汉字串”
  - 注：由三引号来创建的是跨行字符串
- ▶ 字符串中有引号的情况
  - 含有单引号，由双引号括住
  - 含有双引号，由单引号括住
  - 两者有有，使用转义字符 \', \"

```
>>> print("I' am good")
I' am good
>>> print(' "OK" ')
"OK"
>>> print("\"OK\"", I' am good")
"OK", I' am good
>>>
```

```
>>> str='good'
>>> str1="morning"
>>> del str
>>> str1='good'
>>> str2="morining"
>>> str3="""i am very
hungry,
today"""
>>> print(str1)
good
>>> print(str2)
morining
>>> print(str3)
i am very
hungry,
today
>>>
```

# 关于转义字符

- ▶ 以反斜杠(\)开始，后面跟一个或多个字符，表示一些特殊的字符
  - 换行符: `\n`
  - 制表符: `\t`
  - 反斜杠本身、单引号、双引号: `\\, \', \"`
  - Unicode字符: `\uXXXX` , `\UXXXXXXXX`

```
>>> print("Hello\nWorld")
Hello
World
>>> print("Hello\tWorld")
Hello    World
```

```
>>> print("\'Hello\\World\"")
'Hello\\World'
>>> print("\u03A3")
Σ
>>> print("\U0001F602")
😂
```

# 输入函数

- ▶ 如何在程序运行过程中让用户给变量赋值呢？
- ▶ 内置输入函数 `input()`
  - 具体格式：`input(“提示语”)`
  - 执行时，先打印输出自定义提示语，然后等待控制台窗口的输入数据，并且数据保存在内存中
  - 返回值为从控制台窗口输入的字符串
  - 返回值赋给其他变量，例如`x=input(“xxx”)`。

```
>>> x=input("please enter the value of x:")
please enter the value of x:5
>>> print(x)
5
>>>
```

# 输入函数

注意：语句返回值为字符串，可以通过类型转换，实现其它类型数据的输入

```
>>> x=input()  
56  
>>> x/4  
Traceback (most recent call last):  
  File "<pyshell#12>", line 1, in <module>  
    x/4  
TypeError: unsupported operand type(s) for /: 'str' and 'int'  
>>> int(x)/4  
14.0  
>>> |
```

```
ID = input("Please enter your Student ID: ")  
age = int(input("Please enter your age: "))  
height = float(input("Please enter your height(m): "))
```

# 输出函数

- ▶ 内置输出函数 **print()**
- ▶ 具体格式 `print(*object, sep=' ', end='\n', file=sys.stdout)`
- ▶ `object`可以是多个要输出的对象，用逗号隔开
- ▶ `sep`参数默认为一个空格，`end`参数默认为换行符，可以更改
- ▶ `file`参数表示输出到屏幕（或文件）
- ▶ 没有返回值

```
a=45
print(a)
print("xyz", a, 55)
print("xyz", a, 55, sep=" ", end="\n")
print("xyz", a, 55, sep="***", end="!")
print("end")
```

```
===== RESTART: C:/Users/lenovo/Desktop/1.py ==
45
xyz 45 55
xyz 45 55
xyz***45***55! end
>>>
```

# 输入和输出函数的使用

2.py - C:/Users/Chuantao YIN/Desktop/2.py (3.6.3)

File Edit Format Run Options Window Help

```
personName=input("please enter a person's name: ")
animalName=input("please enter a animal's name: ")
place=input("plese enter a place: ")
time=input("please enter a time:")
story=time + ", " + personName + " is walking in " + place + " with a " + animalName
print(story)
```

===== RESTART: C:/Users/Chuantao YIN/Desktop/2.py =====

```
please enter a person's name: Jone
please enter a animal's name: dog
plese enter a place: school
please enter a time:today
today, Jone is walking in school with a dog
>>>
```

===== RESTART: C:/Users/Chuantao YIN/Desktop/2.py =====

```
please enter a person's name: xiaoming
please enter a animal's name: cat
plese enter a place: airport
please enter a time:this morning
this morning, xiaoming is walking in airport with a cat
>>>
```

# 输入函数与输出函数的使用

1.py - C:/Users/chuan/Desktop/1.py (3.12.0)

File Edit Format Run Options Window Help

# 获取用户输入

name = input("请输入你的姓名: ")

age\_str = input("请输入你的年龄: ")

# 将年龄转换为整数

age = int(age\_str)

# 获取当前年份（这里简化为假设今年是2023年）

current\_year = 2023

# 计算出生年份

birth\_year = current\_year - age

# 打印问候语和计算结果

print("\n你好!", name, "!")

print("根据你的年龄, 你大约出生于", birth\_year, "年。")

===== RESTART: C:/Users/chuan/Desktop/1.py =====

请输入你的姓名: chuantao

请输入你的年龄: 41

你好! chuantao !

根据你的年龄, 你大约出生于 1982 年。



# 函数与参数

- ▶ **函数**：能够完成一个计算功能的可重复使用的代码，可以通过函数的名字和参数来调用它。
  - `int()`, `type()`, `input()`, `print()`都属于内置函数
  - **参数**：传递到函数内部的值，可以改变函数的运行方式。

谢谢！