

Python 编程基础

输入、输出及IDLE环境介绍

input函数



功能

接收标准输入数据（即从键盘输入），返回为string类型（字符串）



语法格式

`input([prompt])`

- 1、prompt是一个可选参数，给用户的提示信息；
不传该参数，则没有提示信息，用户直接从键盘输入数据
- 2、本课程规定，如果一个参数写在一对方括号“[...]”中，则表示该参数是可选参数

input函数

示例

1. `name=input("请输入你的姓名：")` #输入 “张三”
2. `print(name)`

eval函数



- 功能：计算字符串所对应的表达式的值，返回表达式的计算结果
- 语法格式：eval(expression)
 - 1、expression是字符串类型的参数，对应一个有效的Python表达式
 - 2、eval函数的完整语法格式为：eval(expression, globals=None, locals=None)

eval函数

示例

1. `r=eval(input("请输入一个有效的表达式："))`
2. `print(r)`

运行结果：输入`3+5`，则输出`8`；输入`5*/3`，则报`SyntaxError`错误

print函数



功能

将各种类型的数据（字符串、整数、浮点数、列表、字典等）
输出到屏幕上



语法格式

`print(object)`

其中，object是要输出的数据

print函数

示例

1. `print("Hello World!")` #输出 "Hello World!"
2. `print(10)` #输出 "10"
3. `print(3.5)` #输出 "3.5"
4. `print([1,3,5,'list'])` #输出 "[1, 3, 5, 'list'] "
5. `print({1:'A', 2:'B', 3:'C', 4:'D'})` #输出 {1: 'A', 2: 'B', 3: 'C', 4: 'D'}



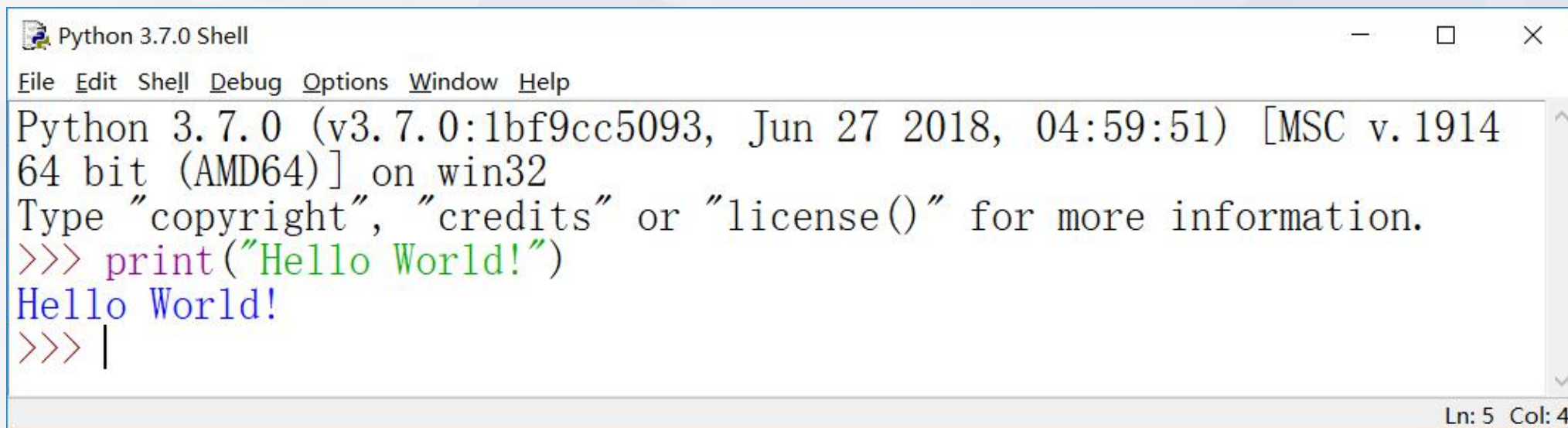
IDLE环境介绍

- IDLE (Python's Integrated Development and Learning Environment , Python集成开发和学习环境)
- 建议使用Jupyter Notebook交互式编程环境，可以大大简化开发流程
- 对于一些大型程序的编写和调试，也可以考虑使用PyCharm等集成开发环境



IDLE环境介绍

- 两种窗口模式：Shell和Editor（编辑器）
- 交互式运行方式



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914
64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello World!")
Hello World!
>>> |
```

Ln: 5 Col: 4

创建Python脚本



选择Shell窗口中的File->New File创建Python脚本文件并打开Editor窗口

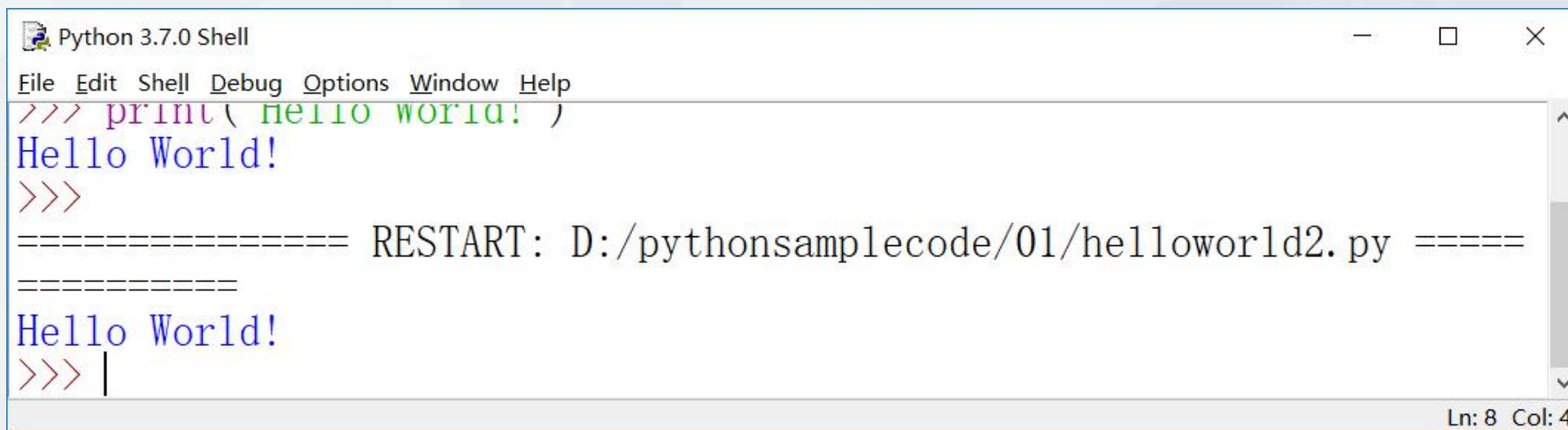
A screenshot of a code editor window titled '*Untitled*'. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The code is written in a monospaced font with syntax highlighting: green for strings and comments, and purple for the print function. The code includes a multi-line string with author and date information, and a print statement to output 'Hello World!'.

```
*Untitled*
File Edit Format Run Options Window Help
'''
This is my first Python program
Author: Kai Wang
Create Date: 07/29/2018
'''
print("Hello World!") #在屏幕上输出 "Hello World!"
```

创建Python脚本



保存（File->Save）后，在Editor窗口选择Run->Run Module菜单项，可运行当前脚本文件，并在Shell窗口输出运行结果



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
>>> print( 'hello world!' )
Hello World!
>>>
===== RESTART: D:/pythonsamplecode/01/helloworld2.py =====
Hello World!
>>> |
```

Ln: 8 Col: 4



Python程序的错误类型

- 在编写Python程序时，主要会遇到两类错误：语法错误和逻辑错误
- 当执行到有语法错误的代码时，Python解释器会显示出错信息，开发者可根据提示信息分析错误原因并解决
- 然而，Python解释器并无法发现逻辑错误，当执行有逻辑错误的代码时，解释器不会报任何错误，但最后的执行结果会与预期不一致

Python程序的调试



- 为了能够分析执行结果错误的原因，所有编程语言的集成开发环境都会提供调试的功能。
- 通过调试可以逐条语句执行程序并查看每条语句执行后各变量的状态，也可以设置断点让程序执行时遇到断点就暂停执行、停在断点所在的代码处。
- 在IDLE的Shell窗口中有一个Debug菜单，该菜单中的菜单项就是用来调试Python程序。
- 目前编写的程序都比较简单，不容易出现逻辑错误；编写复杂程序时如果遇到逻辑错误，可参考网上材料尝试通过调试解决问题。