

FSE598 前沿计算技术

模块 1 计算思维

单元 1 计算机系统设计

第 5 讲 Python 编程基础

本讲座的英文版内容基于教材：

Y. Chen Introduction to Programming Languages: Programming in C, C++, Scheme, Prolog, C#, and Python , 6th edition, Kendall Hunt Publishing Company, 2019.

<https://www.public.asu.edu/~ychen10/book/IntroPl.html>

本讲大纲

学习

- ☐ Python 编程环境
- ☐ Python 程序结构
- ☐ Python 函数和函数调用
- ☐ Python 编程语句

Python 主页：下载最新版本

主页：www.python.org 下载：<https://www.python.org/downloads/>



如果你的版本
早于 3.6，请更
新

安装 Python

- ❑ 如果安装程序提示你添加到系统路径，请选择“是”。
- ❑ 系统路径变量本质上是告诉计算机在哪里可以找到要从命令行运行的“默认可执行文件”。
- ❑ 这样我们只需在终端/命令行中键入“python”或“python3”即可运行 python 代码。
- ❑ 根据你的 IDE 编程环境的选择，这可能不重要或非常重要。

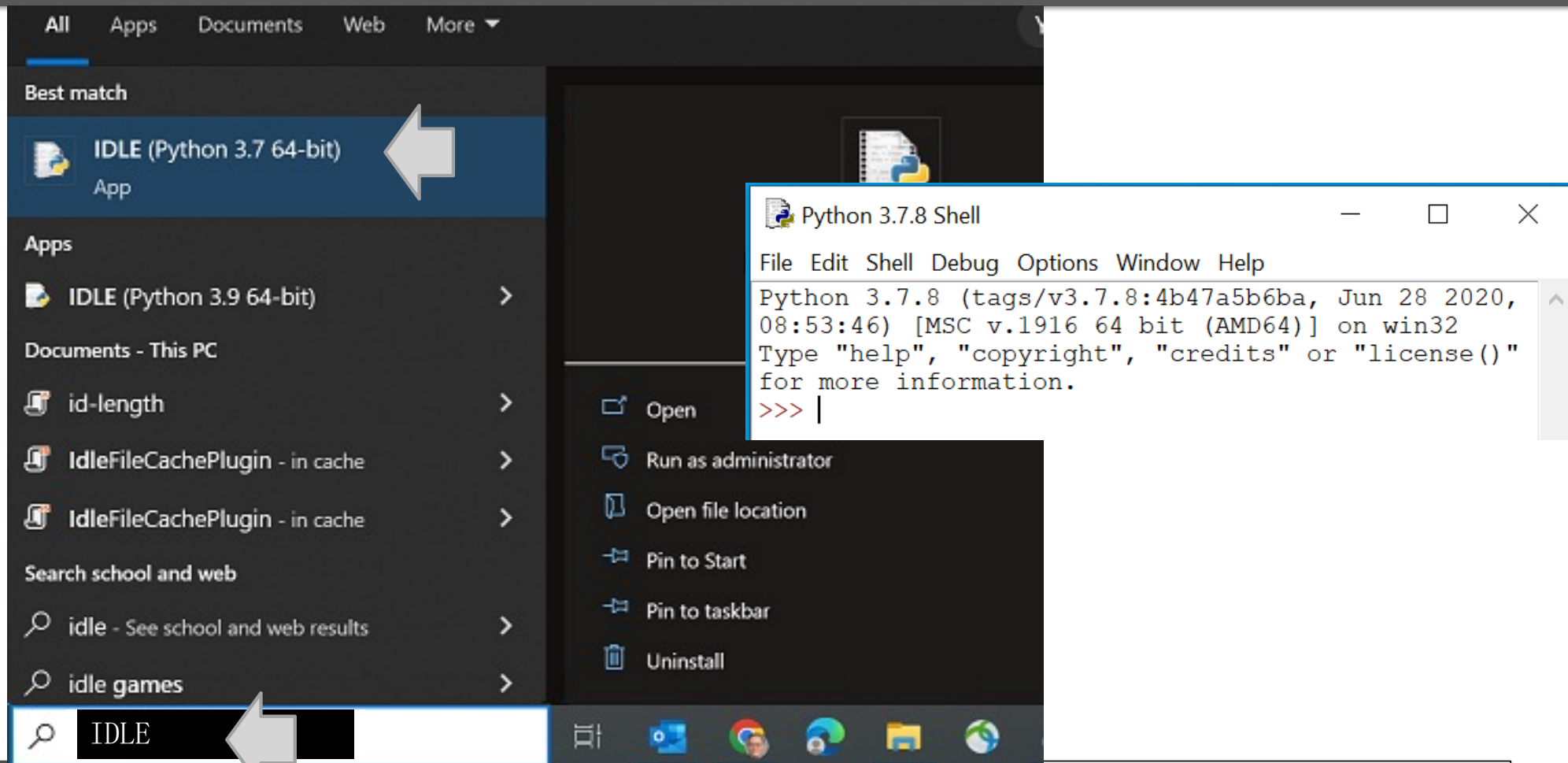
学习新编程语言的典型步骤

1. 如何进行基本输入和输出？
2. 如何创建和操作变量？
3. 如何编写 if 语句？
4. 如何编写循环？
5. 如果函数可用，我该如何编写函数？
返回值、void、参数
如何安装额外的库以重用其他人开发的代码？
6. 如果可以使用类似数组的结构，我该如何创建和使用它？
7. 如果面向对象可用，我如何编写一个类并实例化一个对象？
8. 如果面向服务可用，如何创建服务供他人使用，如何访问他人通过网络部署的服务？

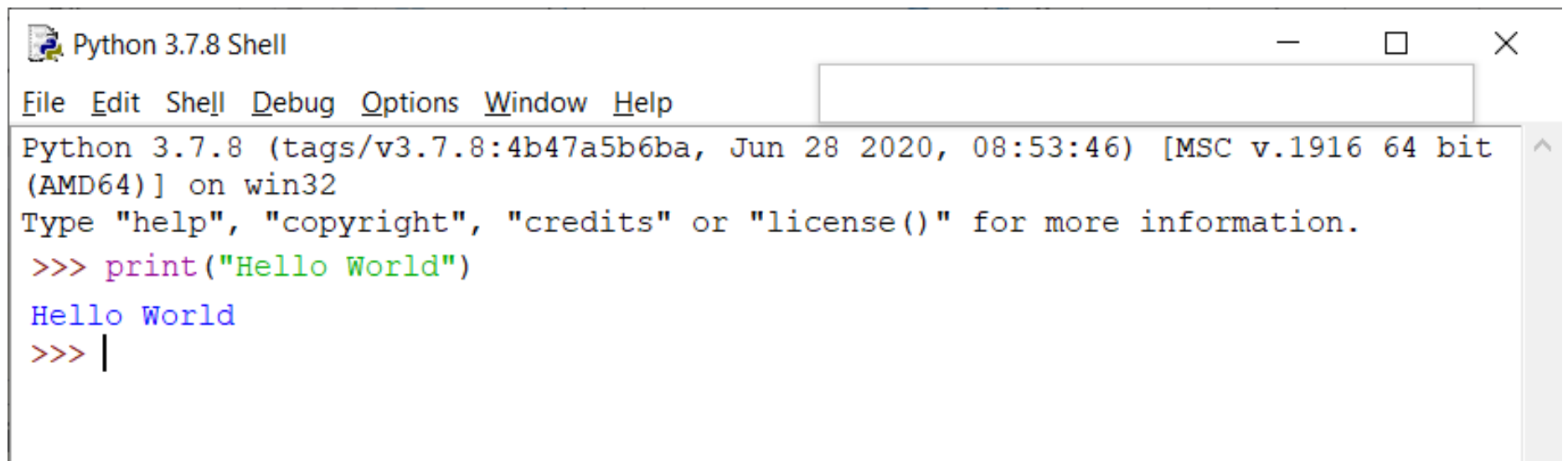
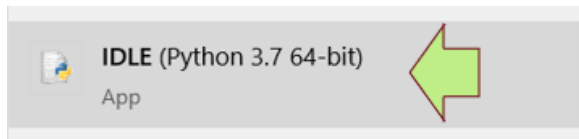
Python 和 IDLE 编程环境

- ❑ Python 会自动安装一个名为 **IDLE** 的简单编辑器和解释器
- ❑ 当你运行 IDLE 时，你将获得一个命令行解释器和创建文件的能力。
- ❑ 在 IDLE 中，你可以从文件 -> 新建文件 创建一个新文件
.....
- ❑ 这将弹出一个文件编辑器窗口
 - 立即保存你的文件。你必须先保存文件，然后才能通过解释器运行它
- ❑ 如果想要在 IDLE 中运行你的代码，只需按下 F5 键
- ❑ IDLE 是一种基本的编程方式，但并不是最便捷的方式。还有更强大的环境。

从搜索栏启动 IDLE



你的第一个程序：Hello World

A screenshot of a 'Python 3.7.8 Shell' window. The window has a title bar with standard minimize, maximize, and close buttons. Below the title bar is a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the following output:

```
Python 3.7.8 (tags/v3.7.8:4b47a5b6ba, Jun 28 2020, 08:53:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print("Hello World")
Hello World
>>> |
```


Python 和更强大的 编程环境 IDE 选项

☐ Visual Studio Python

☐ PyDev for Eclipse

- https://www.pydev.org/manual_101_install.html
- 如果你很适应 Java 的 Eclipse，那么学习曲线会非常短

☐ PyCharm: 具有重构、调试器、代码补全、动态代码分析和编码生产力导向的智能 Python IDE

(<https://www.jetbrains.com/pycharm/download/>)。







☐ VSCode (Mac)

- 非常适合 Visual Studio 轻量级体验
- 缺点:
 - 将 python 解释器绑定到 VS Code 中可能很困难
 - 或者也可能在你第一次尝试时有效.....
 - 你可能必须通过命令行运行解释器

在 Visual Studio 中启动 Python Project 项目

Create a new project

Recent project templates

-  Console Application C#
-  ASP.NET Core gRPC Service C#
-  Blank Django Web Project Python
-  Python Application Python
-  Console App C++
-  Empty Project C++

Python

C#

All platforms

All project types

No exact matches found

Other results based on your search



Python Application

A project for creating a command-line application

Python

Windows

Linux

macOS

Console



From Existing Python code

Create a new project using code files that are already in a folder hierarchy

Python

Linux

macOS

Windows

Console

Web



Web Project

A project for creating a generic Python web project

Python

Windows

Linux

macOS

Web

Back

Next

如果没有找到
Python，请运行
Visual
Studio Installer
以添加

创建 Python 项目

Back

Next

Configure your new project

Python Application

PythonWindowsLinuxmacOS

Project name

Hello

Location

iers\Big Data Mining\Python Course\

...

Solution name ⓘ

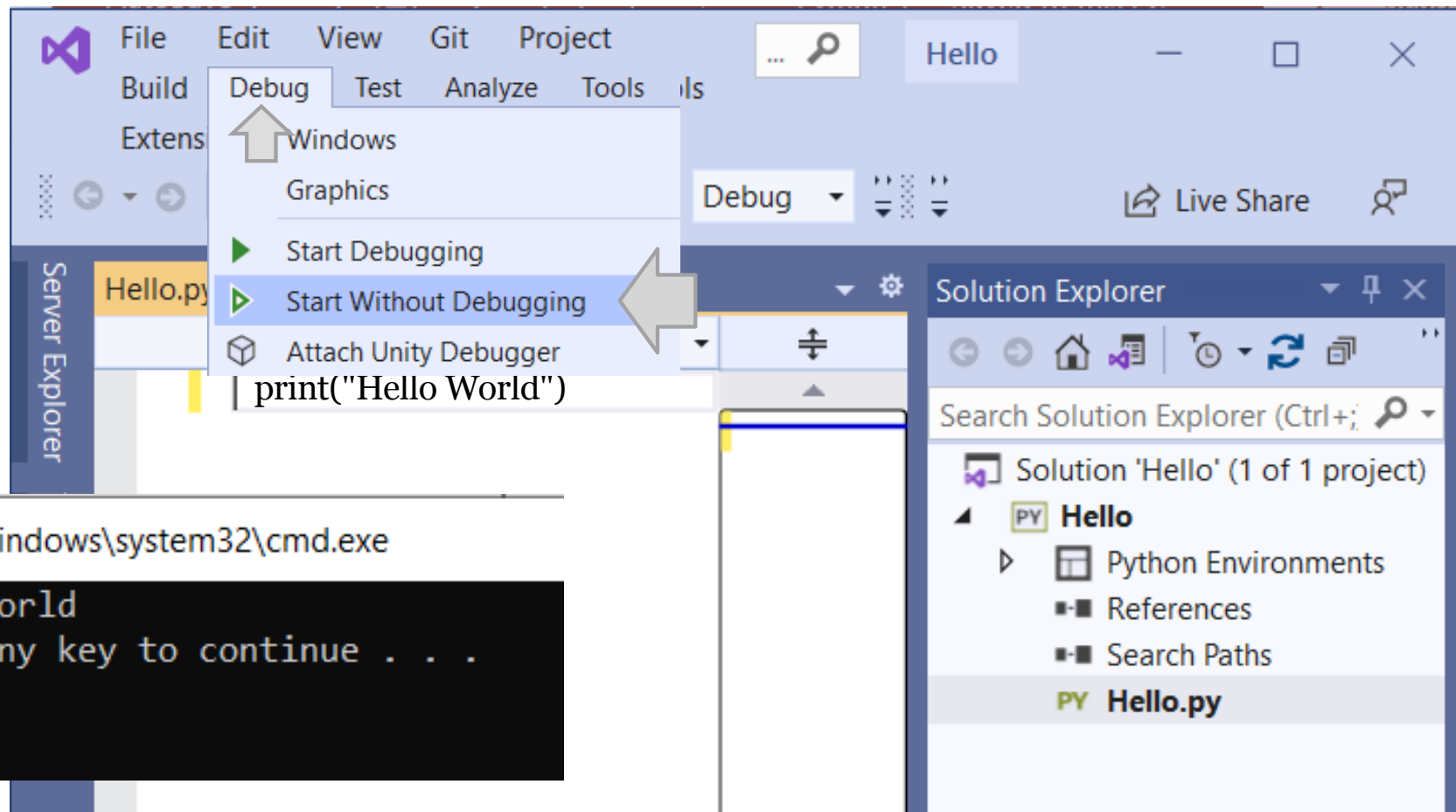
Hello

☒ Place solution and project in the same directory

Back

Create

已创建项目



Python 中的函数

❑ Python允许键入代码并按顺序运行代码，例如：

变量

`i = 5;`

变量

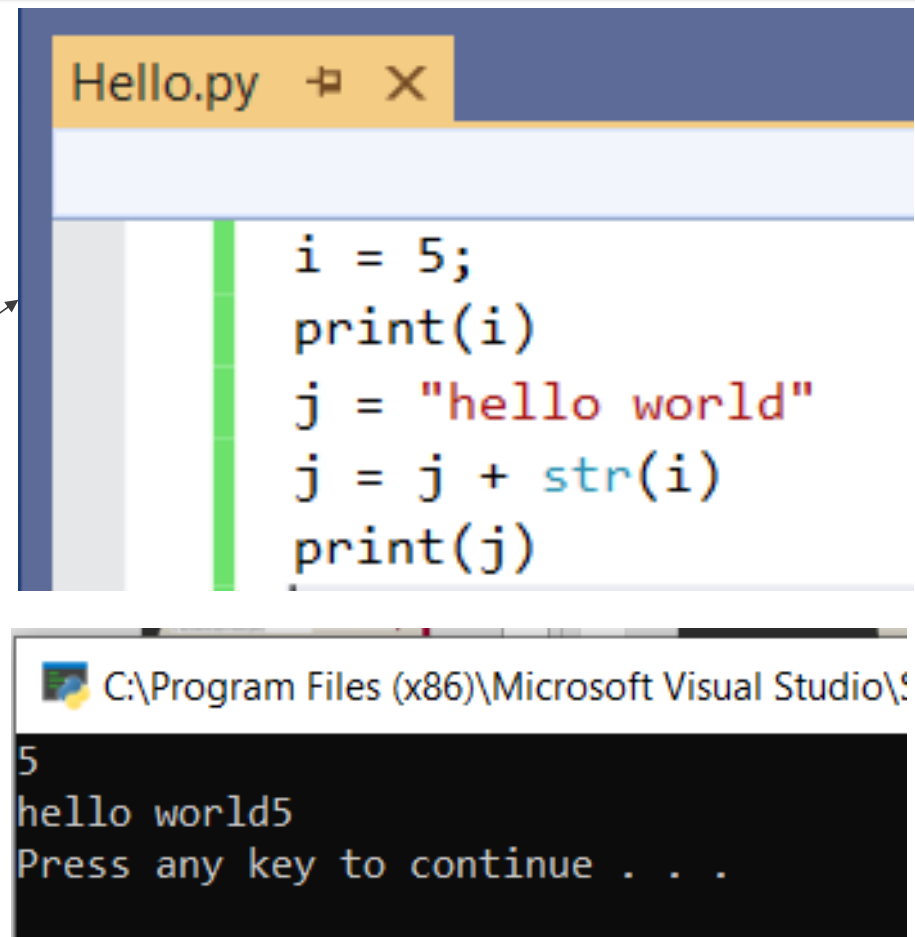
`print(i)`

`j = "hello world"`

`j = j + str(i)`

`print(j)`

将整数转换为
字符串类型



```
Hello.py  + X

i = 5;
print(i)
j = "hello world"
j = j + str(i)
print(j)

C:\Program Files (x86)\Microsoft Visual Studio\
5
hello world5
Press any key to continue . . .
```

Python 程序结构

- ❑ 当代码变长时，我们需要对代码进行结构化
- ❑ 我们将为我们的代码定义一个主函数。
- ❑ Python 不遵循的另一个主要约定是 { } 代码分块
- ❑ 与大多数其他编程语言不同，Python 通过制表符 Tab 和空格结构来识别代码块。它不使用 { } 来分开代码块。

```
def main():
```

```
    i = 5
```

```
    j = 10
```

```
    print("in main i = ", i, "j = ", j)
```

```
if __name__ == "__main__": #required in Python 2, optional in Python 3
```

```
    main()
```

```
print("I can still put other code outside main")
```

“def” 引出一个函数。在被调用前都不会执行该函数

调用 main()

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\
in main i = 5 j = 10
I can still put other code outside main
Press any key to continue . . .
```

对于 Python 3

这一行代码不是必须的：

```
if __name__ == "__main__":    #required in Python 2, optional in Python 3
```

```
def main():  
    i = 5  
    j = 10  
    print("in main i = ", i, "j = ", j)
```

#for Python 3

```
main()
```

```
print("I can still put other code outside main")
```

构建主函数

- ❑ 接下来我们将定义一个主函数
- ❑ `def main():`
 - `print("Hello World")`
- ❑ 函数语法是：
 - `def <function name>(<parameters>):`
 - `[tab] <function body>`
 - 注意：签名行（第一行）不需要指定返回类型。它由函数体中的 `return` 语句自动指定。

```
def main():  
    print("Hello World")
```


调用 main()

- ❑ 定义 main() 后，我们需要调用这一函数。
- ❑ 所以我们可以设置一个 if 语句，确认文件是否直接执行，如果是，则运行主函数
- ❑ 这意味着构建一个 if 语句
- ❑ 这绝对是一个“只复制代码，稍后再理解”的场景
 - 在完全分解之前，让我们来看看函数和 if 语句

```
def main():  
    print("Hello World")  
  
if __name__ == "__main__":  
    main()
```

详细看一下主函数

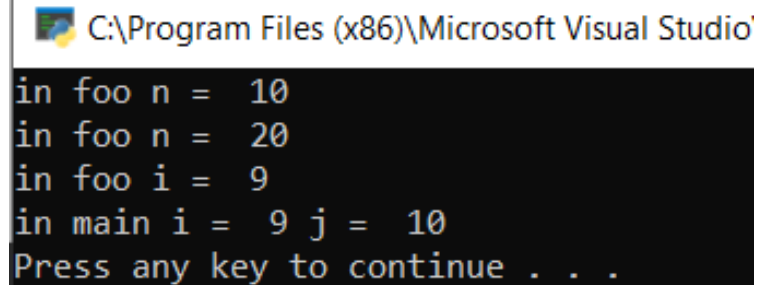
- ❑ 现在有一个主函数
- ❑ 请记住，我们必须使用制表符（Tab）来维持我们的代码块
- ❑ 一旦不用制表符，代码块就结束
- ❑ 会产生一些后果
 - 你可以看到 `print("I'm not in main!!")` 不在主函数中
 - 取消制表符结束 `main()`
 - 此外，它会在主函数前被调用，因为 python 会执行可执行代码（即使我们为自己创建了 `main()`）

```
def main():  
    print("Hello World")  
    print("I'm in main")  
  
print("I'm not in main!!")  
  
if __name__ == "__main__":  
    main()
```

```
PS C:\Users\jselgrad\Documents\Python\HelloWorld> python hello.py  
I'm not in main!!  
Hello World  
I'm in main
```

定义更多函数

```
def foo(n):  
    m = 10  
    print("in foo n = ", n)  
    n = n+m  
    print("in foo n = ", n)  
    global i # indicate we use the global variable i defined outside  
    i += 4  
    print("in foo i = ", i)  
  
def main():  
    j = 10 # is a local variable for main() only  
    foo(j)  
    print("in main i = ", i, "j = ", j)  
  
i = 5 # global variable  
main()
```



C:\Program Files (x86)\Microsoft Visual Studio'
in foo n = 10
in foo n = 20
in foo i = 9
in main i = 9 j = 10
Press any key to continue . . .

Print()

- ❑ `print()` 函数是 Python 中的基本输出语句
- ❑ `print()` 输出对象
- ❑ 它可以通过 , 将多个对象分隔来输出多个对象
 - `print("this", "is", "a", "test")`
 - `print(var1, var2, var3)`

Python 条件和循环语句

❑ Python 能够满足你对标准流量控制套件的期望

- If
- Else
- Else If
- While
- Foreach

❑ 但它缺少某些语句, 例如

- Do While
- For

If 语句

- ❑ if 语句演示了标准的 python-tab 格式

if <conditional>:

do

this

if true

else:

do this if false

- ❑ 制表符切换代码块，取消制表符结束代码块
- ❑ 注意 Python 中的条件不必像在 C/C++/Java/etc 中那样位于()中。
 - 但是，出于好习惯,你还是可以将它们置于()中

If..else.. elif?

- ❑ elif 用于“else if”，与 Java 中的用法相同

```
if x == 0:  
    print “0”  
elif x == 1:  
    print “1”  
else:  
    print “-1”
```

- ❑ 与任何其他 else-if 结构一样，else 从句在所有其他条件都没有满足时执行

While 循环

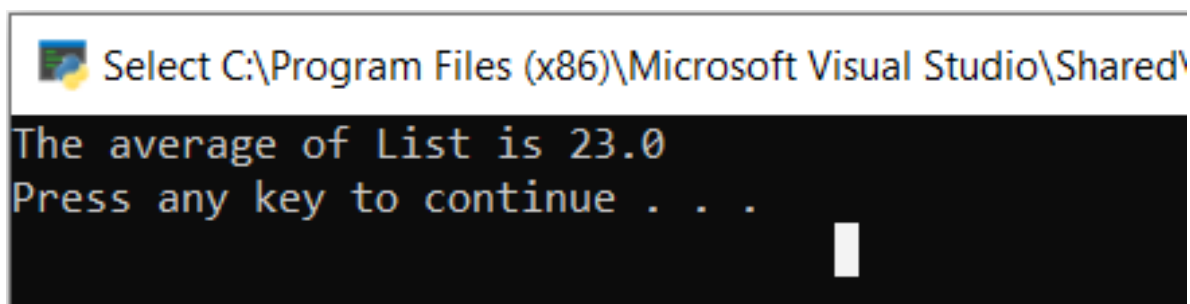
- ❑ While 循环与所有其他语言完全相同
- ❑ 当条件是真，则执行循环主体
 - While 继续遵循冒号-制表符-代码块结构

while <conditional>:
 [do stuff]

```
count = 0
while count < 10:
    print(count)
    count+=1
```


示例：计算平均值

```
def averageOfList(num):  
    sumOfNumbers = 0  
    for t in num:  
        sumOfNumbers = sumOfNumbers + t  
    avg = sumOfNumbers / len(num)  
    return avg  
def main():  
    print("The average of List is", averageOfList([19, 21, 46, 11, 18]))  
main()
```

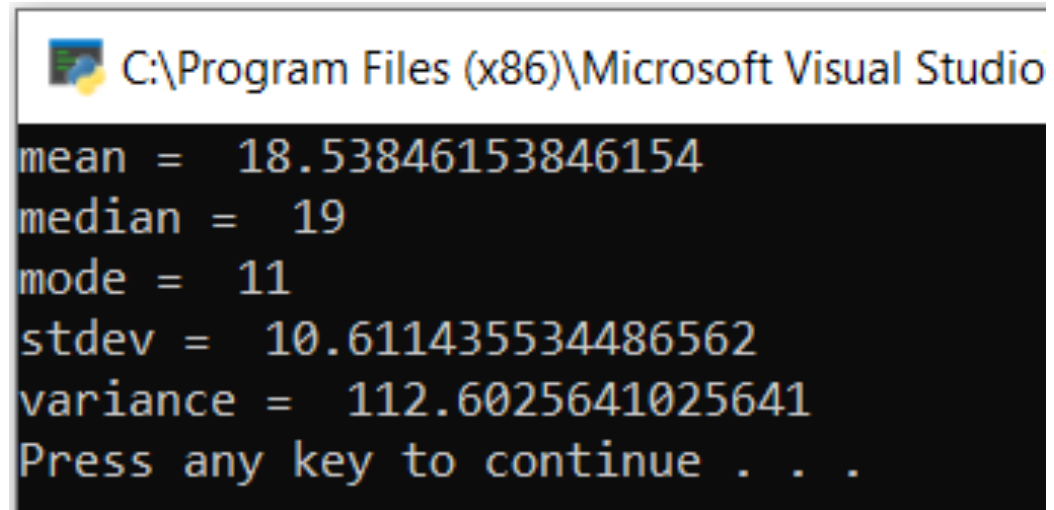


示例：使用statistics库函数

```
import statistics
```

```
data = [11, 21, 11, 19, 46, 21, 19, 29, 21, 18, 3, 11, 11]
```

```
def main():  
    a = statistics.mean(data)  
    print("mean = ", a)  
    b = statistics.median(data)  
    print("median = ", b)  
    c = statistics.mode(data)  
    print("mode = ", c)  
    d = statistics.stdev(data)  
    print("stdev = ", d)  
    e = statistics.variance(data)  
    print("variance = ", e)  
main()
```



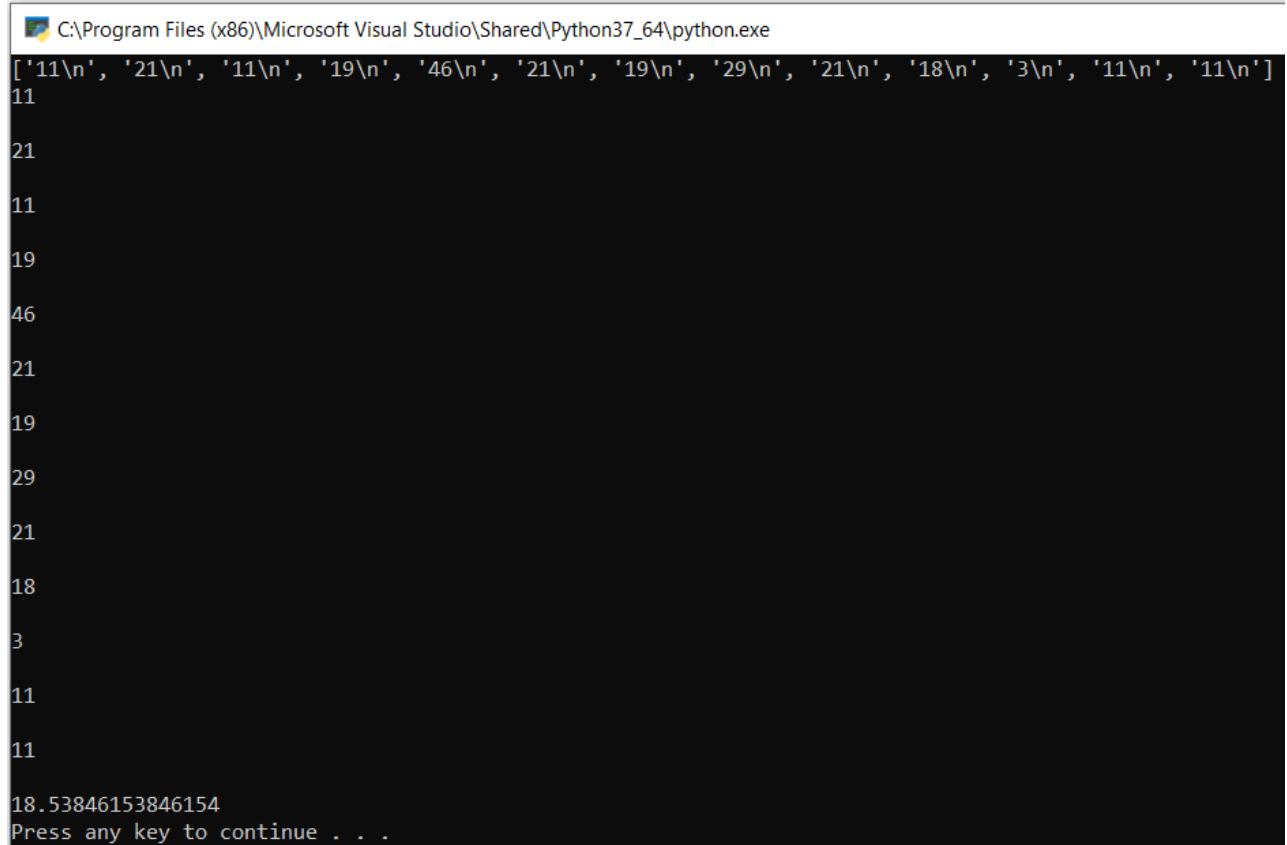
C:\Program Files (x86)\Microsoft Visual Studio

```
mean = 18.53846153846154  
median = 19  
mode = 11  
stdev = 10.611435534486562  
variance = 112.6025641025641  
Press any key to continue . . .
```

示例：从文件读取数据

打开文件

```
data = []
sum = 0
n = 0
with open('mydata.txt') as f:
    data = f.readlines()
    print(data)
    for d in data:
        if (d):
            sum = sum + int(d)
            n += 1
            print(d)
f.close()
average = sum/n
print (average)
```



```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37_64\python.exe
['11\n', '21\n', '11\n', '19\n', '46\n', '21\n', '19\n', '29\n', '21\n', '18\n', '3\n', '11\n', '11\n']
11
21
11
19
46
21
19
29
21
18
3
11
11
18.53846153846154
Press any key to continue . . .
```