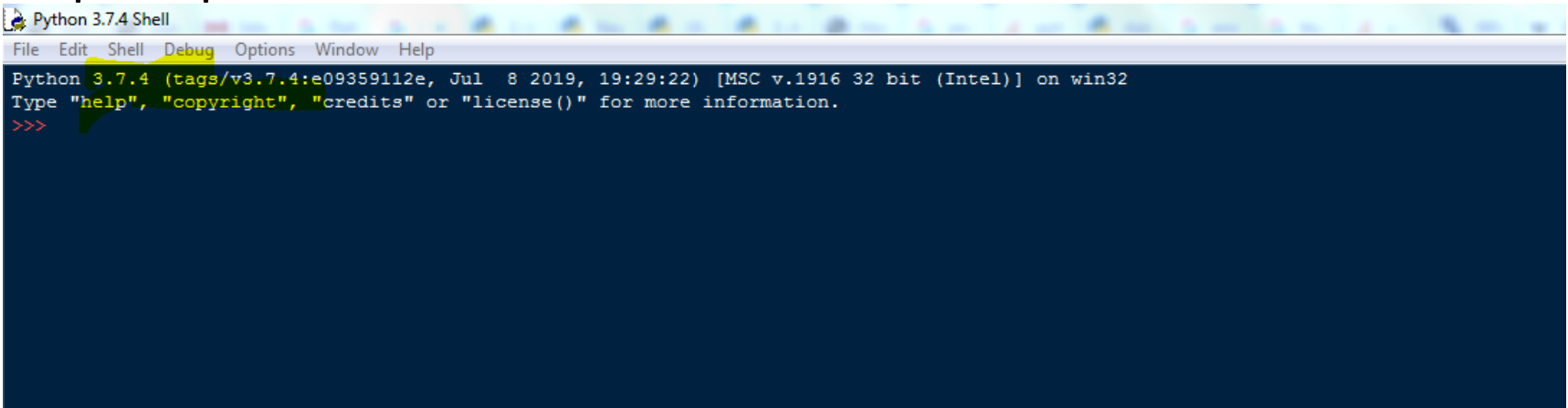


# IDLE

- IDLE is Python's Integrated Development and Learning Environment.
- IDLE has two main window types, the Shell window and the Editor window.

# Python Interpreter

- When commands are read from a tty, the **interpreter is said to be in *interactive mode***. In this mode it prompts for the next command with the **primary prompt, usually three greater-than signs (>>>)**; for continuation lines it prompts with the **secondary prompt**, by default three dots (...).
- The interpreter prints a welcome message stating its version number and a copyright notice before printing the first prompt:

A screenshot of a Python 3.7.4 Shell window. The window title is "Python 3.7.4 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main content area shows the following text: "Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32", "Type \"help\", \"copyright\", \"credits\" or \"license()\" for more information.", and the primary prompt ">>>".

```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 19:29:22) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

# A typical python program

## A typical Python program

```
def function_1(..., ...):  
    ...  
def function_2(..., ...):  
    ...  
    ...  
    ...  
def function_k(..., ...):  
    ...  
  
statement_1  
statement_2  
    ...  
statement_n
```

- Interpreter executes statements from top to bottom
- Function definitions are “digested” for future use
- Actual computation starts from statement\_1

# Using Python as a calculator

- The **interpreter acts as a simple calculator**: you can type an expression at it and it will write the value.
- Expression syntax is straightforward: the operators  $+$ ,  $-$ ,  $*$  and  $/$  work just like in most other languages (for example, Pascal or C); parentheses  $(())$  can be used for grouping.
- For example:

```
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5*6) / 4
5.0
>>> 8 / 5 # division always returns a floating point number
1.6
```

# Using Python as a calculator

- With Python, it is possible to use the **\*\* operator to calculate powers**

```
>>> 5 ** 2 # 5 squared
25
>>> 2 ** 7 # 2 to the power of 7
128
```

- The equal sign (=) is used to assign a value to a variable. Afterwards, no result is displayed before the next interactive prompt:

```
>>> width = 20
>>> height = 5 * 9
>>> width * height
900
```

# Using Python as a calculator

- In interactive mode, the last printed expression is assigned to the variable `_`. This means that when you are using Python as a desk calculator, it is somewhat easier to continue calculations, for example:

```
>>> tax = 12.5 / 100
>>> price = 100.50
>>> price * tax
12.5625
>>> price + _
113.0625
>>> round(_, 2)
113.06
```

# Indentation

- **Whitespace is used for indentation** in Python. All statements with the **same distance to the right belong to the same block of code.**
- If a block has to be more deeply nested, it is simply indented further to the right.
- Most of the programming languages like C, C++, Java use braces { } to define a block of code. One of the distinctive features of Python is its use of indentation to highlight the blocks of code.
- To **indicate a block of code** in Python, you **must indent each line of the block by the same whitespace.**

# Indentation

- The lines `print('Logging on to gcbhoranj...')` and `print('retype the URL.')` are **two separate code blocks**.
- The two blocks of code in our example if-statement are both indented four spaces.
- The **final `print('All set!')` is not indented, and so it does not belong to the else-block.**



# Comments

- A comment in a computer program is text that is intended only for the human reader — it is **completely ignored by the interpreter**.
- In Python, **the # token starts a comment**. The rest of the line is ignored.
- As programs get bigger and more complicated, they get more difficult to read. Formal languages are dense, and it is often **difficult to look at a piece of code and figure out what it is doing, or why**. For this reason, it is a good idea to add notes (comments) to your programs to explain in natural language what the program is doing.