



Introduction to Computer Programming in Python
A SunCam online continuing education course

Python Programming for Engineers - Part 1: Expressions, Data Types, Variables, Strings and Collections

by

Kwabena Oforu, Ph.D., P.E., PTOE



Introduction to Computer Programming in Python
A SunCam online continuing education course

Abstract

Python is a free open source computer programming language that drives some of the internet's most popular websites such as *Google*, *Youtube* and *Instagram*. *Python* can be used to perform complex mathematical calculations, handle big data, build web apps and desktop applications, and manage databases.

This course is the first of a series on the *Python* programming language. This course is tailored to practicing engineers. The course begins with a general overview of computers and computer programming, followed by an introduction to the *Python* language. The course then delves into the details of the IDLE (Python GUI) application for developing and testing *Python* scripts followed by the concepts of expressions, data types, variables, strings, lists, tuples, dictionaries and sets. Several examples of situations encountered by practicing engineers are used to demonstrate the programming concepts and methods presented in this course.

Computer programming enables practitioners to drastically increase their productivity and efficiency by automating repetitive tasks such as tedious design calculations and processing large and complex data. Programming predisposes engineers and scientists to pursue more creative and innovative solutions in their fields of specialty.



Introduction to Computer Programming in Python
A SunCam online continuing education course

TABLE OF CONTENTS

Abstract	ii
List of Tables	v
List of Figures	vi
1. INTRODUCTION	1
1.1 Computers	1
1.2 Computer Programming.....	2
1.3 Relevance of Computer Programming to Engineers	4
1.4 Planning Computer Programming Solutions	4
1.5 Python	5
2. Getting Started	7
2.1 Opening Python	7
2.2 A Quick Tour of IDLE (Python GUI).....	31
3. EXPRESSIONS, DATA TYPES AND VARIABLES.....	36
3.1 Expressions	36
3.2 Arithmetic Operators	39
3.3 Order of Operations	39
3.4 Data Types	44
3.5 Variables	47
3.6 Variable Names.....	49
3.7 Variable Name Conventions.....	50
4. STRINGS	51
4.1 Definition	51
4.2 String Indexing.....	51
4.2 String Slicing	52
4.3 String Operators	53
4.4 Built-in String Functions.....	54
4.5 Built-in String Methods	54
4.6 Modifying Strings.....	58



Introduction to Computer Programming in Python
A SunCam online continuing education course

4.7 Formatted String Literal.....	59
5. LISTS AND TUPLES.....	61
5.1 Lists.....	61
5.2 Manipulating a List.....	62
5.3 List Methods	67
5.4 Tuples.....	69
5.5 Manipulating Tuples	69
6. DICTIONARIES AND SETS.....	71
6.1 Dictionary	71
6.2 Manipulating Dictionaries	72
6.3 Dictionary Methods	75
6.4 Sets.....	77
6.5 Manipulating Sets	78
6.6 Set Operations and Methods	78
7. RUNNING THE PYTHON FILE.....	81
7.1 Code Execution.....	81
7.2 The File Editor	82
7.3 Practical Example	87
7.4 Windows Command Prompt.....	103
8. CONCLUSION.....	115
REFERENCES	116



Introduction to Computer Programming in Python
A SunCam online continuing education course

List of Tables

Table 3. 1: Arithmetic operators 39



Introduction to Computer Programming in Python
A SunCam online continuing education course

List of Figures

Figure 1. 1: Flow of instructions in a computer..... 2



A SunCam online continuing education course

1. INTRODUCTION

1.1 Computers

A computer is an electronic device that has the ability to accept, manipulate, process, store, retrieve, and output data according to programmed instructions. A computer consists of two basic parts a) the **Hardware**: the physical components of the computer, such as the monitor, keyboard, mouse, etc., and b) the **Software**: any set of instructions that tells the hardware what to do, e.g. web browsers, word processors, apps on a smartphone, etc.

The **central processing unit (CPU)** is the hardware component of a computer that carries out the instructions by performing the basic operations such as data input, arithmetic, logic, and output of results. The CPU is essentially the physical “brain” of the computer. The **memory** of a computer refers to the hardware used to store the instructions and data on a temporary or permanent basis. The **storage** of a computer refers to physical components and recording media used to retain the data in electronic format on a long term basis.

The **operating system (OS)** of a computer is software that manages and coordinates the computer's memory, storage, processes, and all of its software and hardware. Modern operating systems use a **Graphical User Interface (GUI)**, pronounced "gooey") to enable a human user to interact with the computer. It is the GUI that enables a user to click on icons, select buttons, menus, checkboxes etc., and have results displayed back to the user on a screen using graphics, text, or some combination thereof. Currently, the most common operating systems in use for personal computers are **Microsoft Windows, Apple Mac OS X, and Linux**.

The flow of data during the interaction of a human user and a personal computer can be depicted schematically, as shown in Figure 1. When a user selects a task to be performed, for example, by clicking with the mouse, using the keyboard, or by using some other input device, instructions are sent to the CPU via the memory to be processed and implemented. Computers are not designed to understand instructions in ordinary everyday language. The instructions sent to the CPU are written in a **computer programming language**, (also called a **programming language**). An element of the CPU called the **compiler**, converts these programmed instructions (commonly called **source code** or **code**) into **machine language**. Machine language consists of binary numbers (zeros and ones). Computers are built to understand and respond to machine language. The instructions can then be implemented and the results sent to an output device such as a monitor for display, or to a printer.



Introduction to Computer Programming in Python
A SunCam online continuing education course

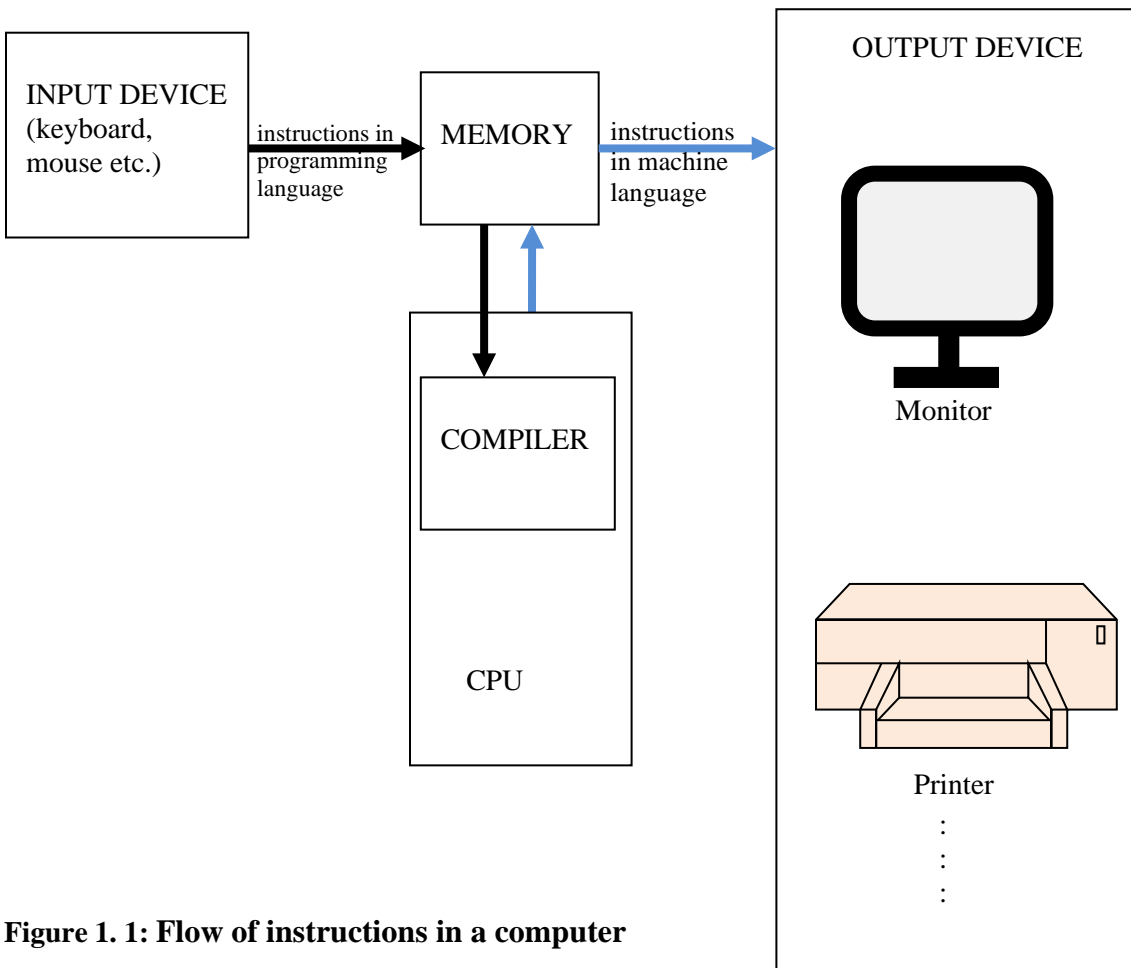


Figure 1. 1: Flow of instructions in a computer

1.2 Computer Programming

A **programming language** is a formal language specifically designed to communicate instructions to a computer. Programming languages can be used for many purposes, for example, to create programs that control the behavior of a computer or device, and/or to implement algorithms accurately and efficiently.

Currently, there are literally hundreds of programming languages in use, each one developed to address particular types of problems and projects. Programming languages can be classified in many ways. One common approach to classification is based on the paradigm, or approach to the



Introduction to Computer Programming in Python
A SunCam online continuing education course

programming. Some of the common classifications based on the approach to programming include **procedural programming**, **event-driven programming**, and **object-oriented programming**. Most modern programming languages, however, include elements from more than one classification. In procedural programming languages, the program (or programmer) specifies the sequence of operations, and program logic determines the next instruction to execute based on inputs from the user. In event-driven programming, the user can press keys or buttons, mouse clicks, check boxes etc., and these user actions can cause an event to occur which triggers some specific procedure for which code has been written. In object-oriented programming, the programmer builds the program by adding **objects** e.g. click buttons, check boxes, etc., through a GUI in the design environment. The objects have **properties** which can be manipulated, e.g., color, or caption on a click button. An action associated with the object is called a **method**. Examples of methods that can be performed on an object based on the instructions include “Move”, “Print”, “Resize”, “Clear”, etc.

Some of the more common and widely used programming languages include:

- *C* (object-oriented programming language)
- *C++* (object-oriented programming language)
- *Java* (object-oriented programming language)
- *Visual Basic* (event driven and object-oriented programming language)
- *Python* (object-oriented and procedural programming language)
- *Ruby* (object-oriented programming language)
- *Pascal* (procedural programming language)
- *Matlab* (procedural programming language)
- *Fortran* (procedural programming language), and many others.

The choice of what programming language to use on a specific project is determined by several factors such as:

- the operating system of the end user(s)
- the programming approach that is relevant, procedural programming or event-driven, etc.
- how well the structure and features of the language are compatible with the project goals
- the level to which the program will be used to manipulate hardware components e.g. in video games, smartphone apps, etc.
- the ease with which new features can be added to the existing program
- the skill set of the programming team
- support for, and community standing behind the language



Introduction to Computer Programming in Python

A SunCam online continuing education course

Computer programming (or programming) is a comprehensive process of formulating a computing problem, developing a methodology to solve the problem, writing code in a specific programming language to implement the solution methodology, testing, debugging, maintaining the code, verifying and validating the results, and monitoring the consumption of computer resources over the entire process. The objective of programming, therefore, is to develop a series of instructions that can automate the performance of a specific task, or to solve some specific problem formulation in a timely and efficient manner. Therefore, programming involves a multidisciplinary as well as an interdisciplinary approach.

1.3 Relevance of Computer Programming to Engineers

Computer programming has become an increasingly advantageous and necessary skill for engineers and scientists competing in the global economy. By writing computer programs to automate tedious and repetitive tasks such as design calculations, or preparing, processing and analyzing large amounts of data which would otherwise be done by hand, engineers and scientists can drastically increase their productivity and efficiency. Competence in computer programming predisposes engineers and scientists to pursue and develop more creative and innovative solutions than their peers. Knowledge, and some level of experience in computer programming enables engineers and scientists to communicate more effectively with full-time programmers and information technology professionals they may work with and collaborate with on various projects.

1.4 Planning Computer Programming Solutions

A 6-step procedure for planning and implementing computer programming solutions to a problem or research question is as follows:

1. Study the problem in detail and gain a good working understanding of the fundamental concepts and principles, and underlying theories. This may involve knowledge from a wide variety of fields such as engineering, the physical sciences, mathematics, statistics, business, economics, government and many others.
2. Develop an algorithm for solving the problem. A flow chart of the algorithm will provide a detailed visual representation of the process from start to finish, including all inputs, calculations, data processing, automated reasoning, outputs, reports, and feedback procedures.
3. Write **pseudo code** for your algorithm. Pseudo code is the instructions needed to implement the algorithm in ordinary everyday language. Test your algorithm by performing manual calculations to verify your results.



Introduction to Computer Programming in Python
A SunCam online continuing education course

4. Write code to implement the algorithm in the target programming language following all syntax rules and requirements.
5. Test and debug the code intermittently to identify and resolve glitches and errors.
6. Test the program. Validate and verify the results. Make changes and updates as necessary where needed.

1.5 Python

Python is an interpreted, high-level, general purpose programming language. An **interpreted language** is one in which most of the programming structures and implementations execute the instructions directly without pre-compiling into machine language. An **interpreter** is a software that directly executes instructions written in a programming language, without the need to pre-compile the instructions into machine language.

In this course series, the font and lettering style “Python” shall be used to refer to tools, applications, interpreters, software etc. that enable instructions written in the *Python* programming language to be executed.

A **high-level** computer programming language is one that enables development of a program in a more user-friendly programming context and is generally independent of the computer's hardware architecture. A high-level language is closer to ordinary human language when compared to machine language (and the converse is true for a **low-level** programming language). *Python* supports multiple programming paradigms, including the object-oriented and procedural approaches to programming.

There are many “versions” or **implementations** of *Python*. The **reference implementation** of *Python* (called *CPython*), is a free, open source software managed by the Python Software Foundation. *Python interpreters* come free and pre-installed with some operating systems. In this course, all examples will be presented using Microsoft Windows.

Python can be used to perform complex mathematical and engineering calculations, and to handle big data. *Python* can be used for building GUIs and desktop applications. *Python* can be used on a server for web development and to build web apps. *Python* can be used to connect to database systems and can read and modify files. Due to the fact that *Python* runs on an interpreter system, the code is executed rapidly which enables quick prototyping or production-ready software development.



Introduction to Computer Programming in Python
A SunCam online continuing education course

As a high-level language, *Python* has a simpler syntax similar to the English language. The syntax of *Python* enables code to be written with fewer lines than some other programming languages. Due to its ease of use, *Python* is amenable for use by beginners as well as seasoned programmers. *Python* is increasingly popular as a first programming language for beginners.

As a result of its user-friendliness and versatility, *Python* is the programming language driving some of the internet's most popular websites, namely:

- *Google*
- *Youtube*
- *Quora*
- *Dropbox*
- *Yahoo!*
- *Yahoo Maps*
- *Reddit*
- *Bitly*
- *Instagram*
- *Spotify*
- *SurveyMonkey*
- *Pintrest*
- *Eventbrite*
- *Firefox*
- *and many others*



Introduction to Computer Programming in Python
A SunCam online continuing education course

2. Getting Started

2.1 Opening Python

Let's see if you have the Python interpreter on your computer, and what version of the program you have, if you have it.

From your Desktop,

Click on Windows **Start**.



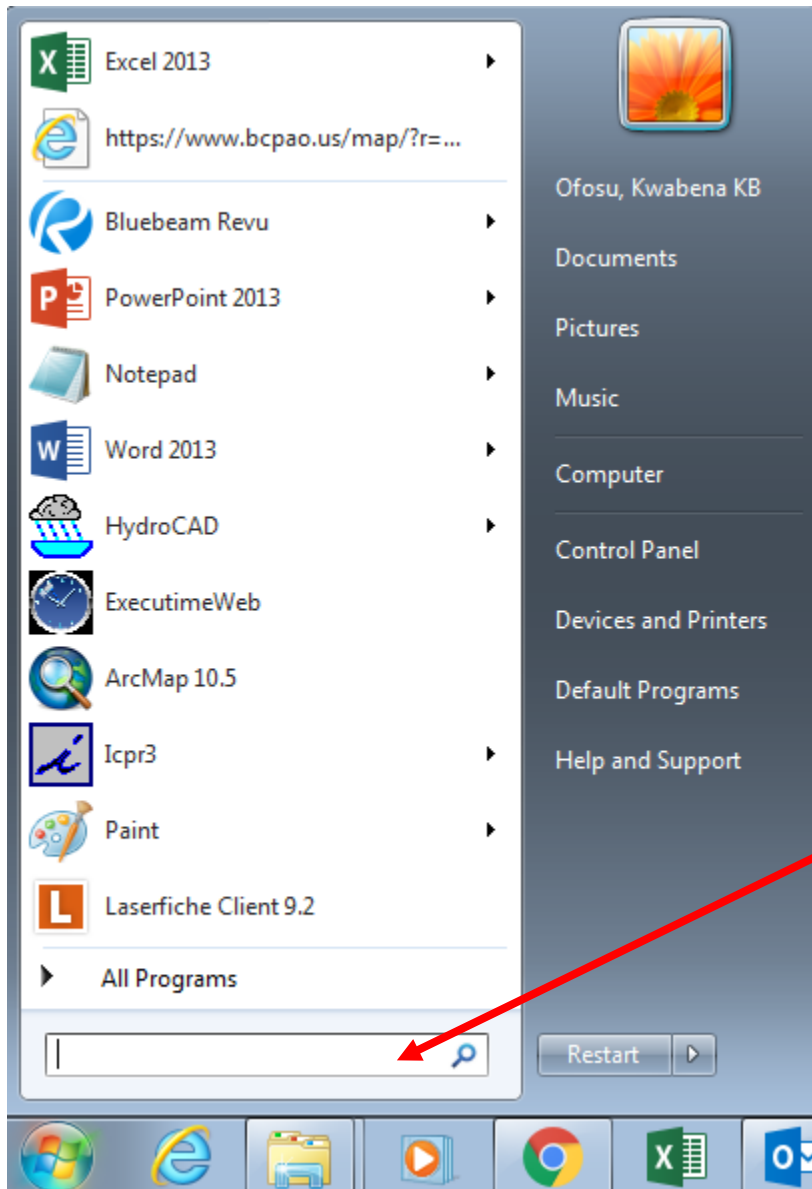


Introduction to Computer Programming in Python
A SunCam online continuing education course

In the Search bar, type “Python” or “IDLE”.

IDLE is the desktop **interactive shell** or GUI that comes with Python.

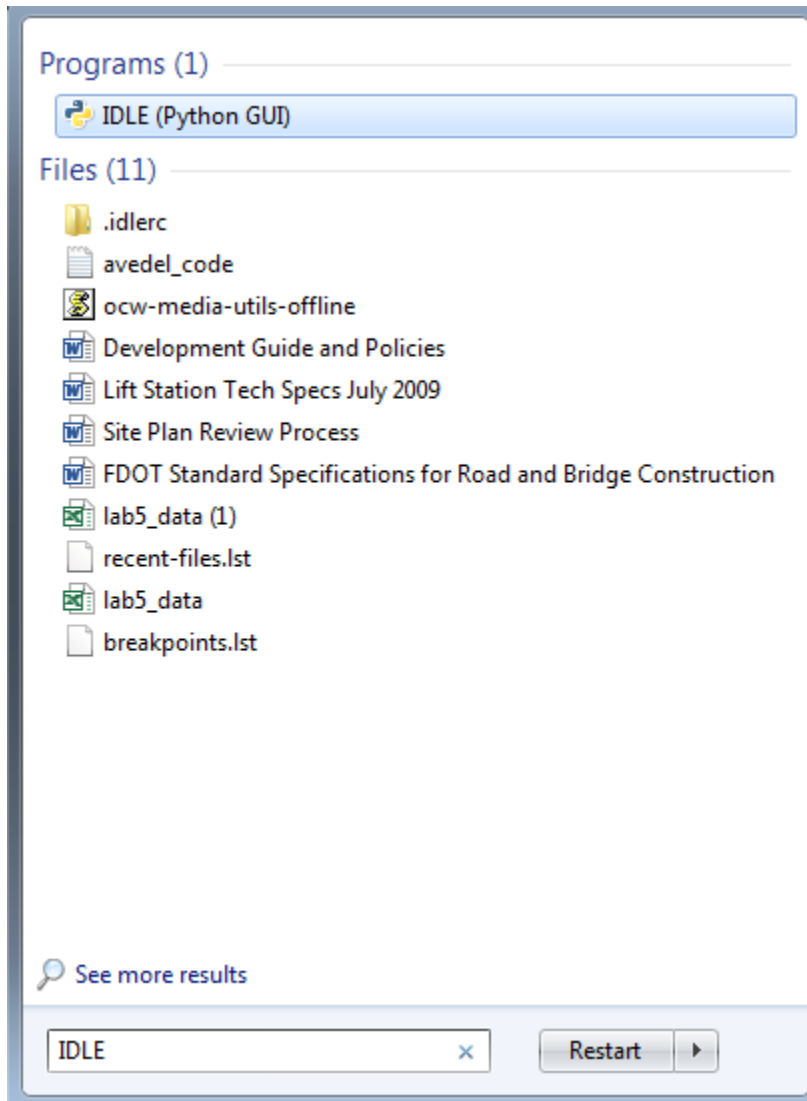
IDLE is a text editor program (like a *Notepad* or a *Wordpad*).





Introduction to Computer Programming in Python
A SunCam online continuing education course

The **Start** window refreshes.

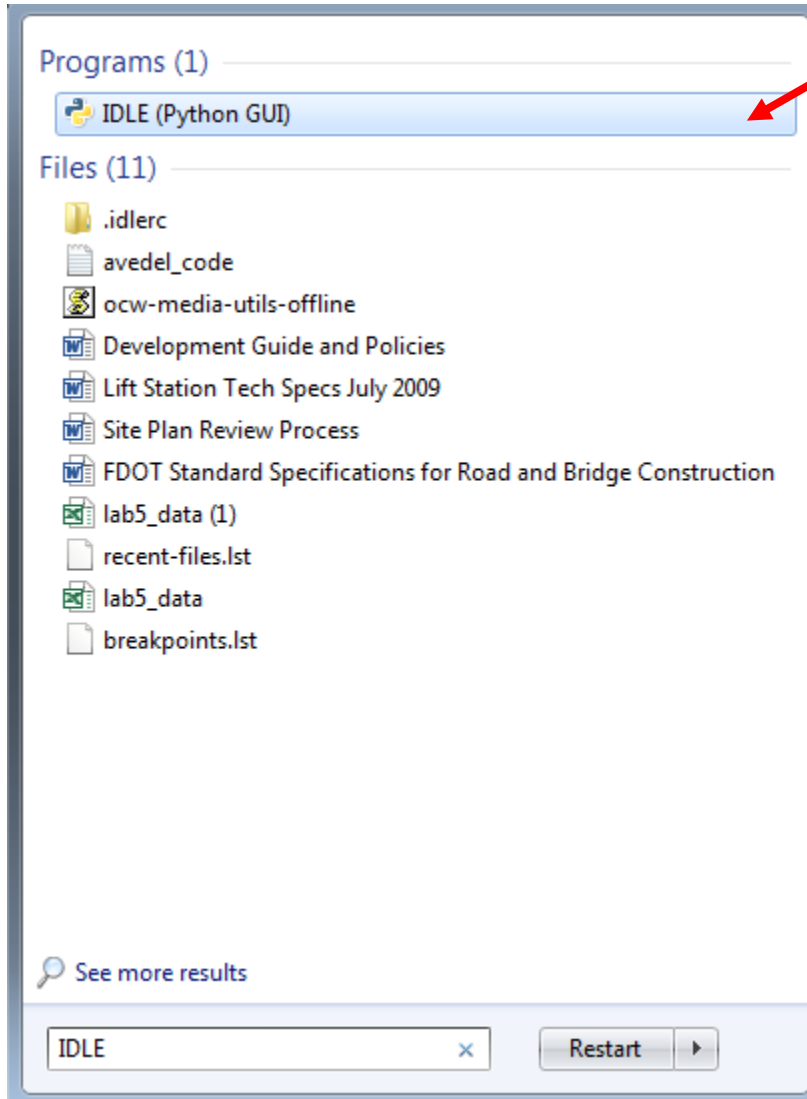


If you do not have the program, no worries! Please follow along, in the next few pages we shall demonstrate where and how to download a Python interpreter.



Introduction to Computer Programming in Python
A SunCam online continuing education course

If you have the program, click on **IDLE (Python GUI)** to open it.





Introduction to Computer Programming in Python
A SunCam online continuing education course

The interactive shell IDLE (Python GUI) opens.

A screenshot of the Python 2.7.12 Shell window. The window title is "Python 2.7.12 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following text:

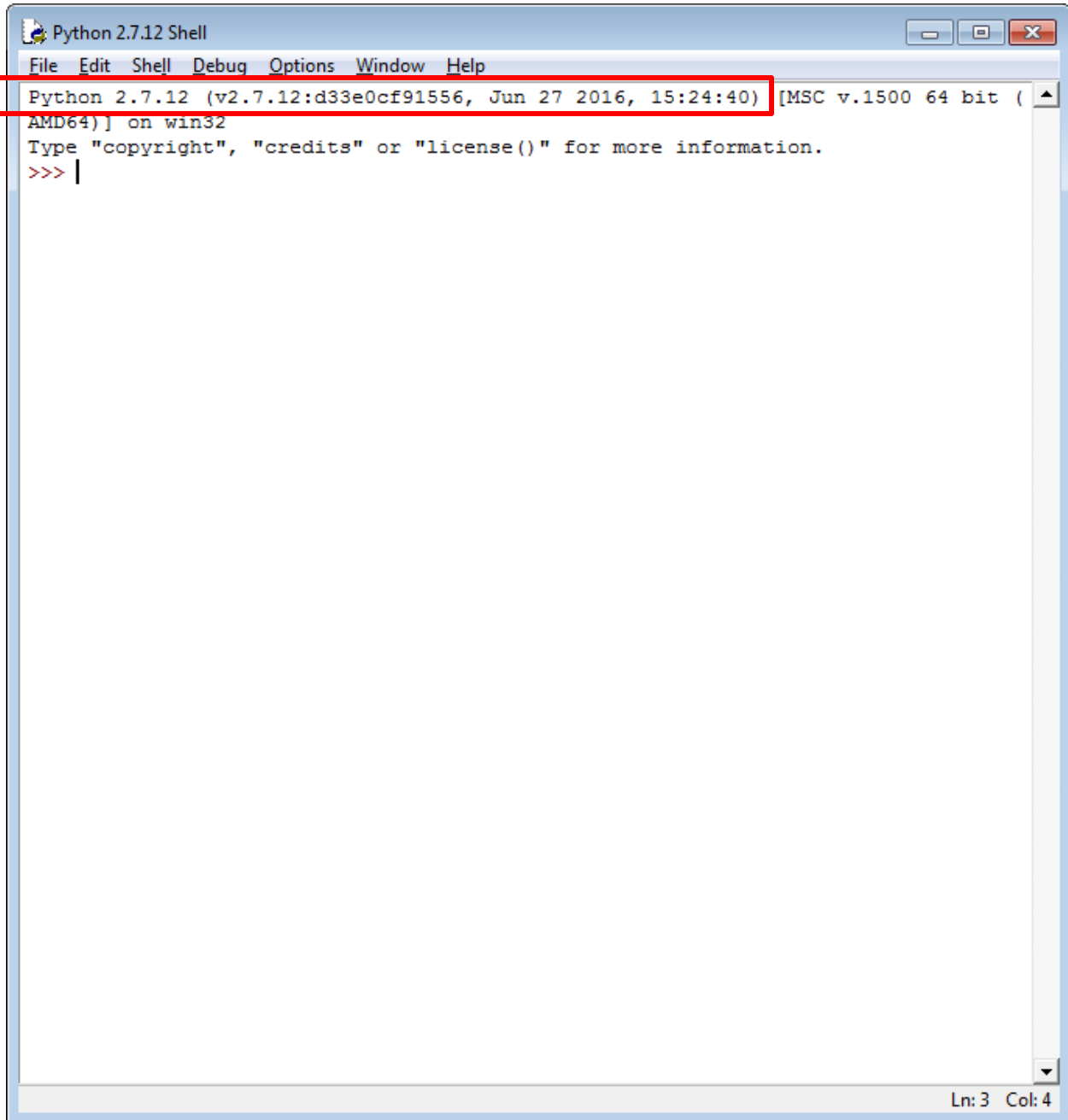
```
Python 2.7.12 (v2.7.12:d33e0cf91556, Jun 27 2016, 15:24:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> |
```

The status bar at the bottom right shows "Ln: 3 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

Note the edition and date of the program that you have.

A screenshot of a Windows-style application window titled "Python 2.7.12 Shell". The window has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main content area displays the following text:

```
Python 2.7.12 (v2.7.12:d33e0cf91556, Jun 27 2016, 15:24:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> |
```

The first line of the output is enclosed in a red rectangular box. The status bar at the bottom right of the window shows "Ln: 3 Col: 4".



Introduction to Computer Programming in Python A SunCam online continuing education course

It is always recommended to use the latest and greatest edition of the software. To find out what that version is we shall go to the Python Software Foundation website.

Google “Python Software Foundation.”

The screenshot shows a Google search interface. The search bar contains the text "python software foundation". Below the search bar, there are navigation tabs for "All", "News", "Maps", "Images", "Shopping", "More", "Settings", and "Tools". The search results show "About 52,600,000 results (0.47 seconds)". The top result is "Python Software Foundation - Python.org" with the URL "https://www.python.org/psf/". Below this, there is a brief description: "The Python Software Foundation (PSF) is a 501(c)(3) non-profit corporation that holds the intellectual property rights behind the Python programming language." There are several sub-links: "About", "Grants program", "Sponsors", "Membership", and "Donate to the PSF". At the bottom, there is a link for "Python Software Foundation - Wikipedia".

Python Software Foundation - Python.org
<https://www.python.org/psf/> ▼
 The **Python Software Foundation (PSF)** is a 501(c)(3) non-profit corporation that holds the intellectual property rights behind the Python programming language.

About
 The mission of the Python Software Foundation is to ...

Grants program
 If you have questions about the Grants program, please read ...

Python Software Foundation
 The Python Software Foundation is an organization devoted to ...

Sponsors
 Python Software Foundation Sponsors. The PSF would not ...

Membership
 PSF Membership FAQ. The mission of the Python Software ...

Donate to the PSF
 Donors. The PSF sends a big thank you to all of the donors ...

[More results from python.org »](#)

[Python Software Foundation - Wikipedia](#)



Introduction to Computer Programming in Python A SunCam online continuing education course

Follow the links to the website.

The PSF

The Python Software Foundation is the organization behind Python. Become a member of the PSF and help advance the software and our mission.

Tweets by @ThePSF

Python Software @ThePSF
Berlin Python Pizza - A micro conference organized by the Python Berlin Community. Feb 23, 2019 in Berlin, Germany. berlin.python.pizza. In practical sessions developers share their experience and

Python Software Foundation

The mission of the Python Software Foundation is to promote, protect, and advance the Python programming language, and to support and facilitate the growth of a diverse and international community of Python programmers.

—from the Mission Statement page

The Python Software Foundation (PSF) is a 501(c)(3) non-profit corporation that holds the intellectual property rights behind the Python programming language. We manage the open source licensing for Python version 2.1 and later and own and protect the trademarks associated with Python. We also run the North American PyCon conference annually, support



Introduction to Computer Programming in Python A SunCam online continuing education course

Click on **Python** to open the Python page.

The screenshot shows the Python Software Foundation website. A red arrow points to the 'Python' link in the top navigation bar. The page content includes a header with the Python logo and 'SOFTWARE FOUNDATION', a search bar, and a secondary navigation bar with links like 'About', 'Sponsorship', etc. The main content area features a section titled 'The PSF' with a description of the foundation's mission, a 'Tweets by @ThePSF' section, and a 'Python Software Foundation' section with a mission statement and a brief description of the organization.

The PSF
The Python Software Foundation is the organization behind Python. Become a member of the PSF and help advance the software and our mission.

Tweets by @ThePSF

Python Software @ThePSF
Berlin Python Pizza - A micro conference organized by the Python Berlin Community. Feb 23, 2019 in Berlin, Germany. berlin.python.pizza. In practical sessions developers share their experience and

Python Software Foundation

The mission of the Python Software Foundation is to promote, protect, and advance the Python programming language, and to support and facilitate the growth of a diverse and international community of Python programmers.

—from the *Mission Statement* page

The Python Software Foundation (PSF) is a 501(c)(3) non-profit corporation that holds the intellectual property rights behind the Python programming language. We manage the open source licensing for Python version 2.1 and later and own and protect the trademarks associated with Python. We also run the North American PyCon conference annually, support



Introduction to Computer Programming in Python A SunCam online continuing education course

The Python page opens.

The screenshot shows the Python.org website homepage. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is the Python logo and a search bar. A secondary navigation bar includes links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a code editor with Python code demonstrating list comprehensions and the enumerate function. To the right of the code is a section titled "Compound Data Types" explaining lists. Below the code and text is a promotional message: "Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)". At the bottom, there are four columns of links: "Get Started", "Download", "Docs", and "Jobs", each with a brief description.

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in
fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Compound Data Types

Lists (known as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)

- Get Started**
Whether you're new to programming or an experienced
- Download**
Python source code and installers are available for download for all
- Docs**
Documentation for Python's standard library, along with tutorials
- Jobs**
Looking for work or have a Python related position that you're trying to



Introduction to Computer Programming in Python
A SunCam online continuing education course

Hover over **Downloads**.

The screenshot shows the Python.org website. A red arrow points to the 'Downloads' menu item in the navigation bar. Below the navigation bar, there is a search bar and a 'Socialize' button. The main content area features a code editor with Python code and a 'Compound Data Types' section. At the bottom, there are four columns: 'Get Started', 'Download', 'Docs', and 'Jobs'.

Python PSF Docs PyPI Jobs Community

python™ Search GO Socialize

About Downloads Documentation Community Success Stories News Events

```
# Python 3: List comprehensions
>>> fruits = ['Banana', 'Apple', 'Lime']
>>> loud_fruits = [fruit.upper() for fruit in fruits]
>>> print(loud_fruits)
['BANANA', 'APPLE', 'LIME']

# List and the enumerate function
>>> list(enumerate(fruits))
[(0, 'Banana'), (1, 'Apple'), (2, 'Lime')]
```

Compound Data Types

Lists (known as arrays in other languages) are one of the compound data types that Python understands. Lists can be indexed, sliced and manipulated with other built-in functions. [More about lists in Python 3](#)

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. [>>> Learn More](#)

Get Started
Whether you're new to programming or an experienced

Download
Python source code and installers are available for download for all

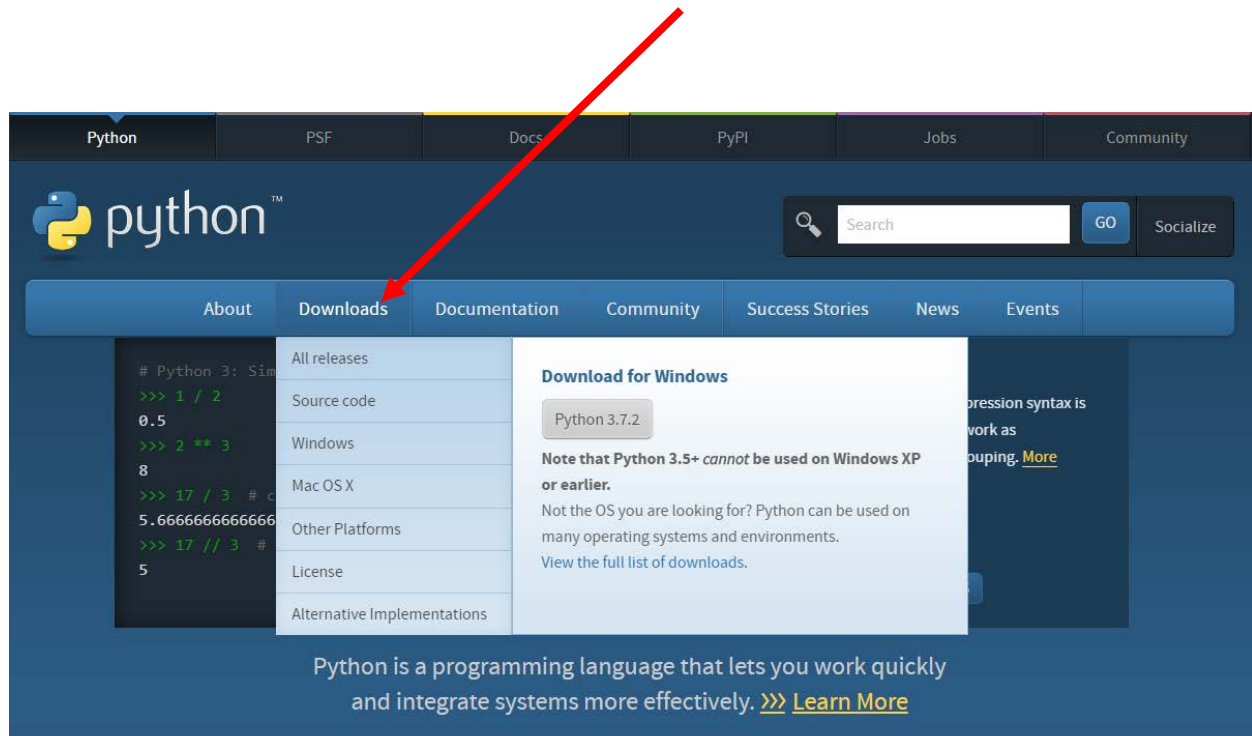
Docs
Documentation for Python's standard library, along with tutorials

Jobs
Looking for work or have a Python related position that you're trying to



Introduction to Computer Programming in Python
A SunCam online continuing education course

Hovering over **Downloads** will enable you to see the latest and greatest packages available for your specific operating system.



If the current version listed on the webpage differs from the version on your computer, or if you did not have the program to begin with, we shall shortly demonstrate how to download the latest and greatest edition of the software compatible with your computer's operating system, from the Python webpage.

If you prefer not to download the program you may launch and use the interactive shell available on the webpage as described in the next two pages.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Hover away from the **Downloads** tab to observe the interactive shell launch button.

The screenshot shows the Python.org website interface. At the top, there are navigation tabs for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is the Python logo and a search bar. A secondary navigation bar includes links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a code snippet on the left and an article titled 'All the Flow You'd Expect' on the right. A red arrow points to a yellow button with a terminal prompt symbol (>_) located next to the code snippet. Below the main content, there are four columns of links: 'Get Started', 'Download', 'Docs', and 'Jobs', each with a brief description.

Click on **Launch Interactive Shell**.



Introduction to Computer Programming in Python A SunCam online continuing education course

The web interactive shell launches as follows.

The screenshot displays the Python.org homepage. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is the Python logo and a search bar. A secondary navigation bar includes links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a terminal window with the following text:

```
Python 3.7.0 (default, Aug 22 2018, 20:50:05)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Below the terminal window, there is a promotional message: "Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)". At the bottom of the page, there are four main navigation buttons: "Get Started" (with a power icon), "Download" (with a download icon), "Docs" (with a document icon), and "Jobs" (with a briefcase icon). Each button has a short description below it.

It must be noted, however, that the webpage interactive shell [and in fact the desktop interactive shell IDLE (Python GUI)], is of limited use and on its own will not be capable of completing all of the tasks required to complete this course. However, as we shall see, the desktop interactive shell IDLE (Python GUI) that we shall download (if you do not have the latest and greatest version already) connects to other modules and applications that provide the requisite capabilities needed to complete this course and to develop real-life projects. Therefore, in this course, all methods will be demonstrated using the desktop interactive shell IDLE (Python GUI).



Introduction to Computer Programming in Python
A SunCam online continuing education course

Before we conduct a download of the Python interpreter, let's find out what operating system you have running on your computer. (Seasoned computer users bear with us, or skip to page 31).

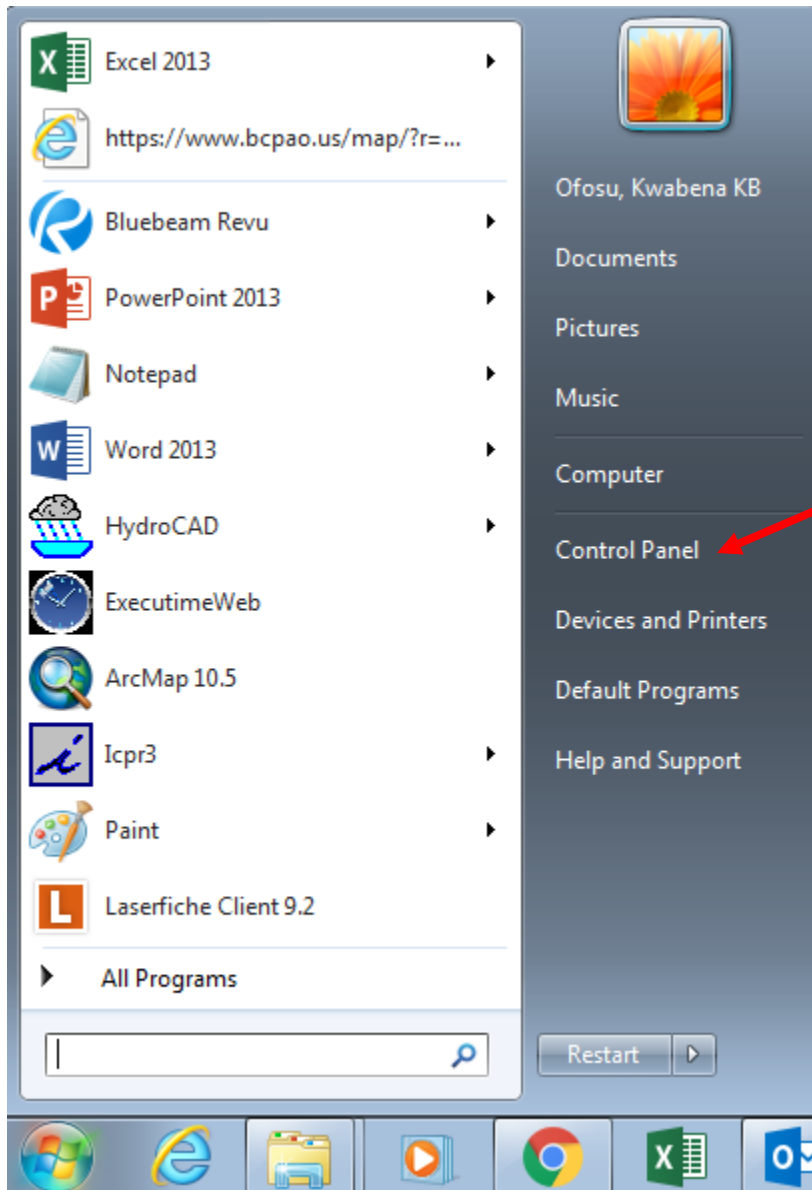
From your Desktop,
Click on **Start**.





Introduction to Computer Programming in Python
A SunCam online continuing education course

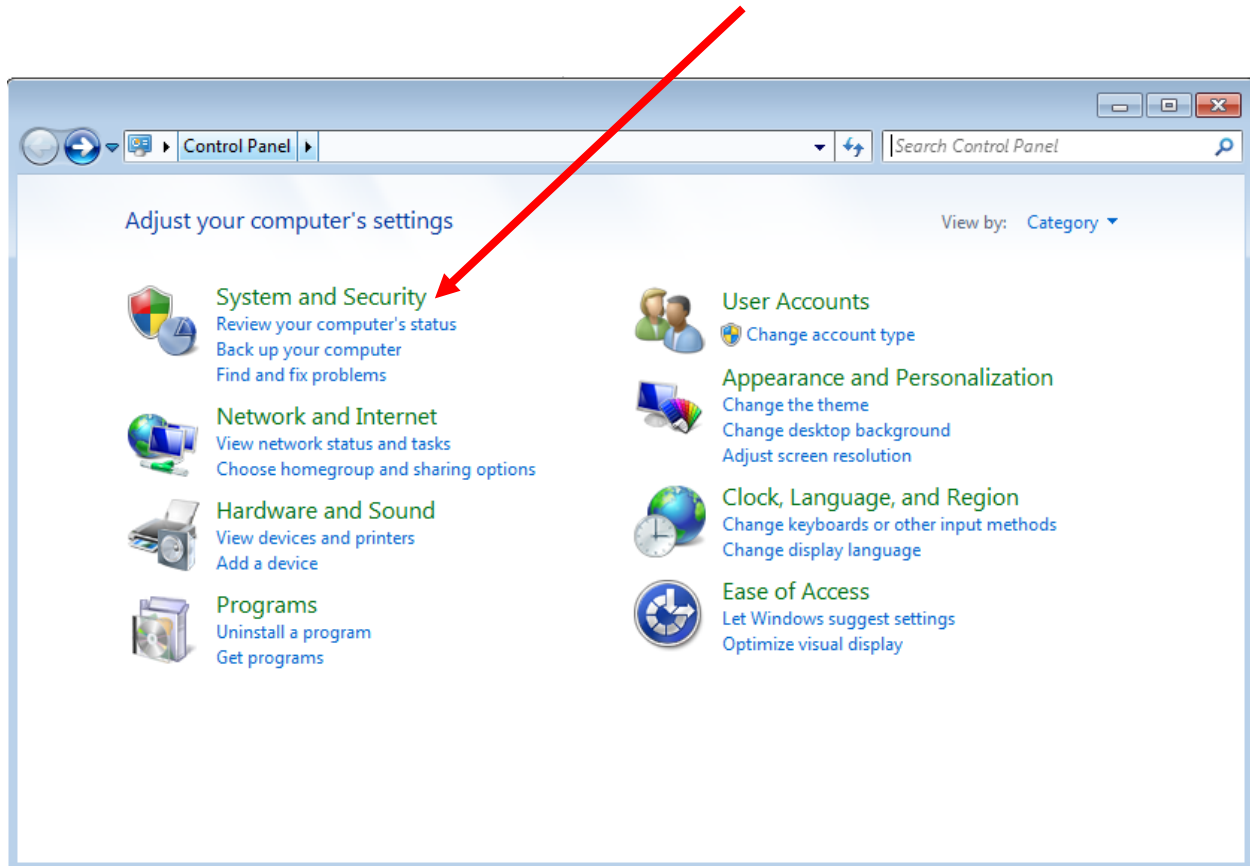
Click on **Control Panel**.





Introduction to Computer Programming in Python
A SunCam online continuing education course

The Control Panel opens.
Click on **System and Security**.

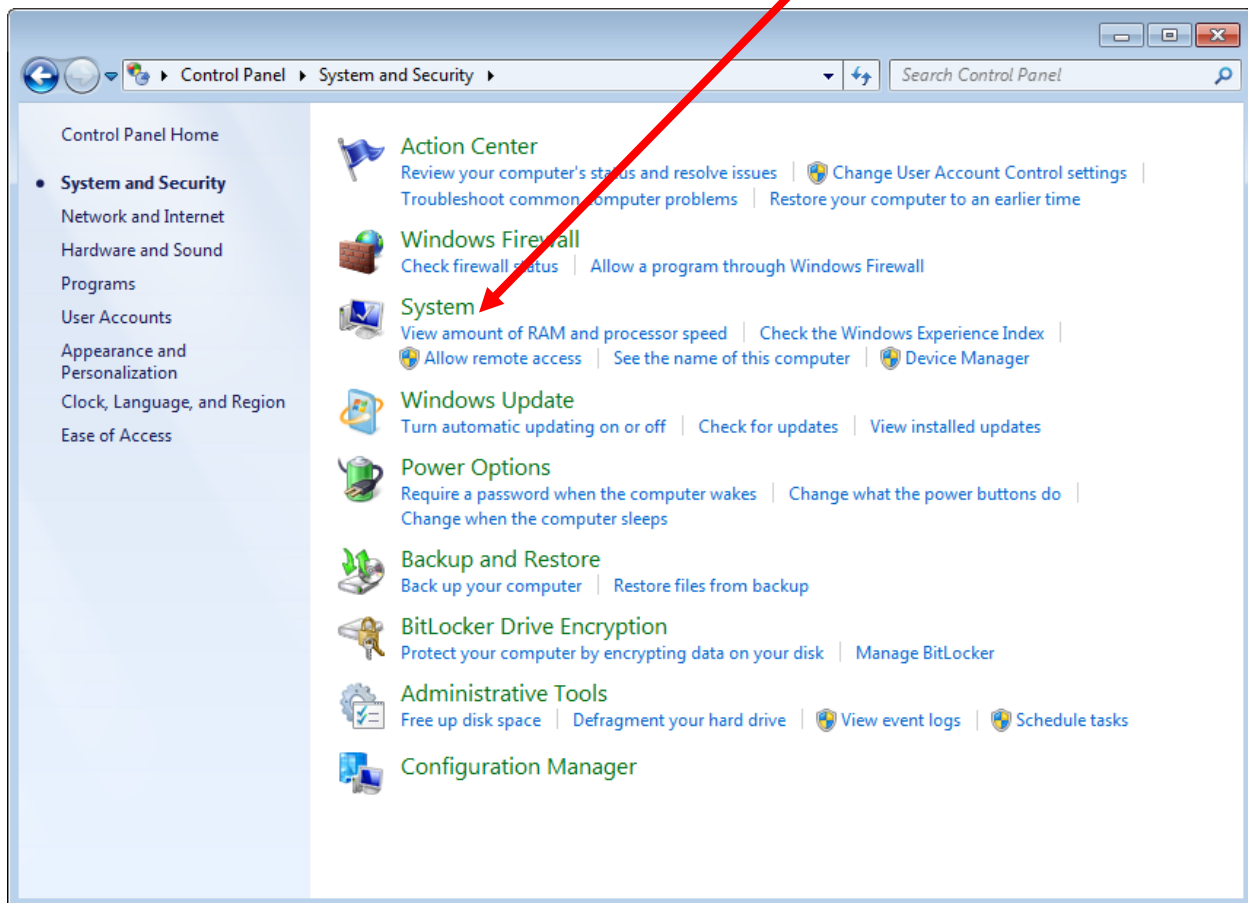




Introduction to Computer Programming in Python *A SunCam online continuing education course*

The System and Security window opens.

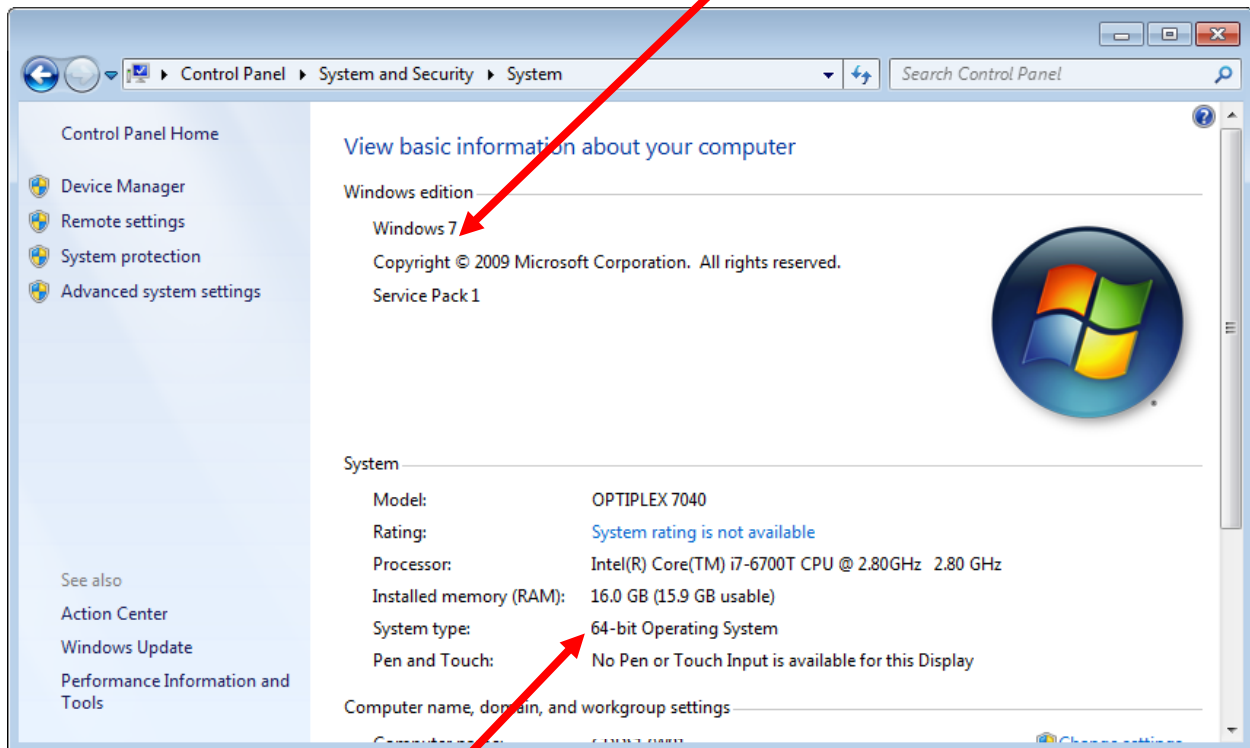
Click on **System**.





Introduction to Computer Programming in Python
A SunCam online continuing education course

And from this window we see that in this example, the user is running Windows 7, with 64-bit operating system.



So we must download a compatible version of Python for this user's operating system.



Introduction to Computer Programming in Python A SunCam online continuing education course

Now, as an example, let's conduct a download of the Python interpreter for a Windows 64-bit operating system.

Hover over **Downloads**.

The screenshot shows the Python.org website. A red arrow points to the 'Downloads' link in the main navigation bar. The 'Downloads' dropdown menu is open, showing options: All releases, Source code, Windows, Mac OS X, Other Platforms, License, and Alternative Implementations. The 'Windows' option is selected, leading to the 'Download for Windows' section. This section features a 'Python 3.7.2' button, a note that Python 3.5+ cannot be used on Windows XP or earlier, and a link to view the full list of downloads. Below the main content, there are four columns: 'Get Started', 'Download', 'Docs', and 'Jobs', each with a brief description.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **View full list of downloads.**

The screenshot shows the Python.org website. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is the Python logo and a search bar. A secondary navigation bar contains links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The 'Downloads' menu is open, showing options like All releases, Source code, Windows, Mac OS X, Other Platforms, License, and Alternative Implementations. A 'Download for Windows' section is visible, featuring a 'Python 3.7.2' button and a note that Python 3.5+ cannot be used on Windows XP or earlier. A red arrow points to the link 'View the full list of downloads.' in this section. At the bottom of the page, there are four main sections: 'Get Started', 'Download', 'Docs', and 'Jobs'. A red arrow also points from the 'View the full list of downloads.' link to the 'Jobs' section.



Introduction to Computer Programming in Python A SunCam online continuing education course

Click on **Windows**.

(If you have another operating system, please click on the relevant link).

The screenshot shows the Python.org website. The main navigation bar includes links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is a search bar and a 'Socialize' button. The secondary navigation bar includes links for About, Downloads, Documentation, Community, Success Stories, News, and Events. The main content area features a large heading 'Download the latest version for Windows' and a prominent yellow button labeled 'Download Python 3.7.2'. Below the button are links for other operating systems: 'Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)'. There are also links for 'Pre-releases' and 'Looking for Python 2.7? See below for specific releases'. A red arrow points from the text above to the 'Download Python 3.7.2' button. Below the main content area, there is a section titled 'Looking for a specific release?' with the text 'Python releases by version number:' and a table of releases.

Release version	Release date	Click for more	
Python 3.7.2	2018-12-24	Download	Release Notes
Python 3.6.8	2018-12-24	Download	Release Notes



Introduction to Computer Programming in Python A SunCam online continuing education course

Click on any Windows 64-bit download option.

(If you have another operating system, please click on the relevant link).

Python >>> Downloads >>> Windows

Python Releases for Windows

- [Latest Python 3 Release - Python 3.7.2](#)
- [Latest Python 2 Release - Python 2.7.15](#)
- [Python 3.7.2 - 2018-12-24](#)
 - [Download Windows x86 web-based installer](#)
 - [Download Windows x86 executable installer](#)
 - [Download Windows x86 embeddable zip file](#)
 - [Download Windows x86-64 web-based installer](#)
 - [Download Windows x86-64 executable installer](#)
 - [Download Windows x86-64 embeddable zip file](#)
 - [Download Windows help file](#)
- [Python 3.6.8 - 2018-12-24](#)
 - [Download Windows x86 web-based installer](#)

Select any one of them to download

Follow the prompts from the webpage and from your computer to complete the download.

The download is now complete.

Go back to the procedure we conducted on page 13 through page 16 to open your latest and greatest version of the desktop interactive shell IDLE (Python GUI).



Introduction to Computer Programming in Python A SunCam online continuing education course

Return to the Python webpage.

Other links of interest on the Python webpage include the Documentation tab where manuals, users' guides, tutorial videos, etc., can be found, as well as the Community tab which enables one to join user communities, mailing lists, conferences, forums, subject matter discussion boards, etc.

The screenshot shows the Python.org homepage. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, Jobs, and Community. Below this is a search bar and a 'Socialize' button. A secondary navigation bar contains links for About, Downloads, Documentation, Community, Success Stories, News, and Events. Two red arrows point to the 'Documentation' and 'Community' links. The main content area features a code editor with Python code for list comprehensions and the enumerate function, alongside an article titled 'Compound Data Types' about lists. At the bottom, there are four columns of quick links: 'Get Started', 'Download', 'Docs', and 'Jobs'.



Introduction to Computer Programming in Python
A SunCam online continuing education course

2.2 A Quick Tour of IDLE (Python GUI)

With the latest and greatest version of the software installed and up and running, let us take a moment to review some of the key features of the interactive shell IDLE (Python GUI).

First, let's look at the **Command Line**. The triple angle bracket prompt is where the commands shall be typed for execution on a line-by-line basis.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following text:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> |
```

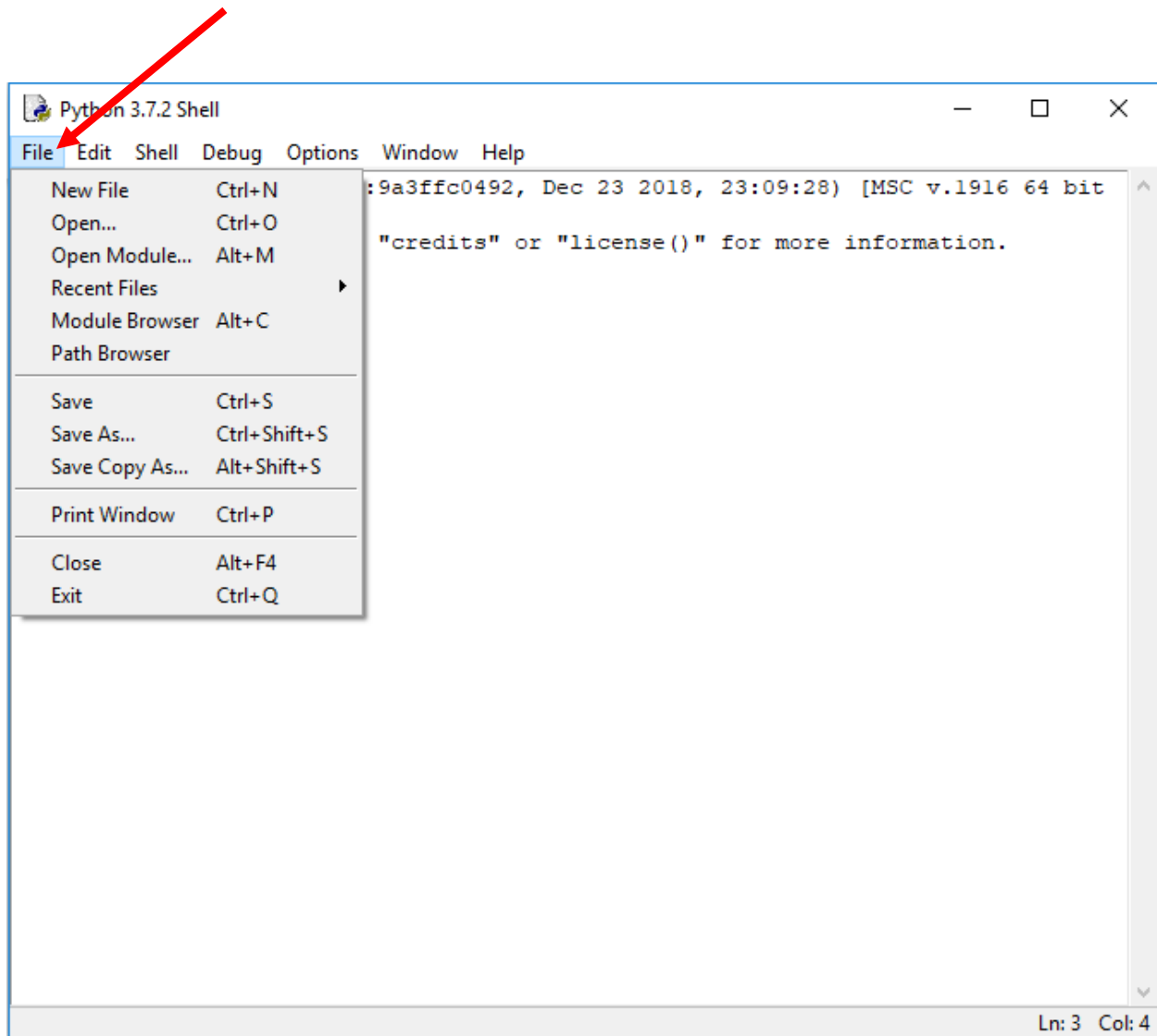
A red arrow points from the text "Command Line" to the triple angle bracket prompt ">>> |". The status bar at the bottom right shows "Ln: 3 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **File**.

These are tools to create manipulate and manage files.

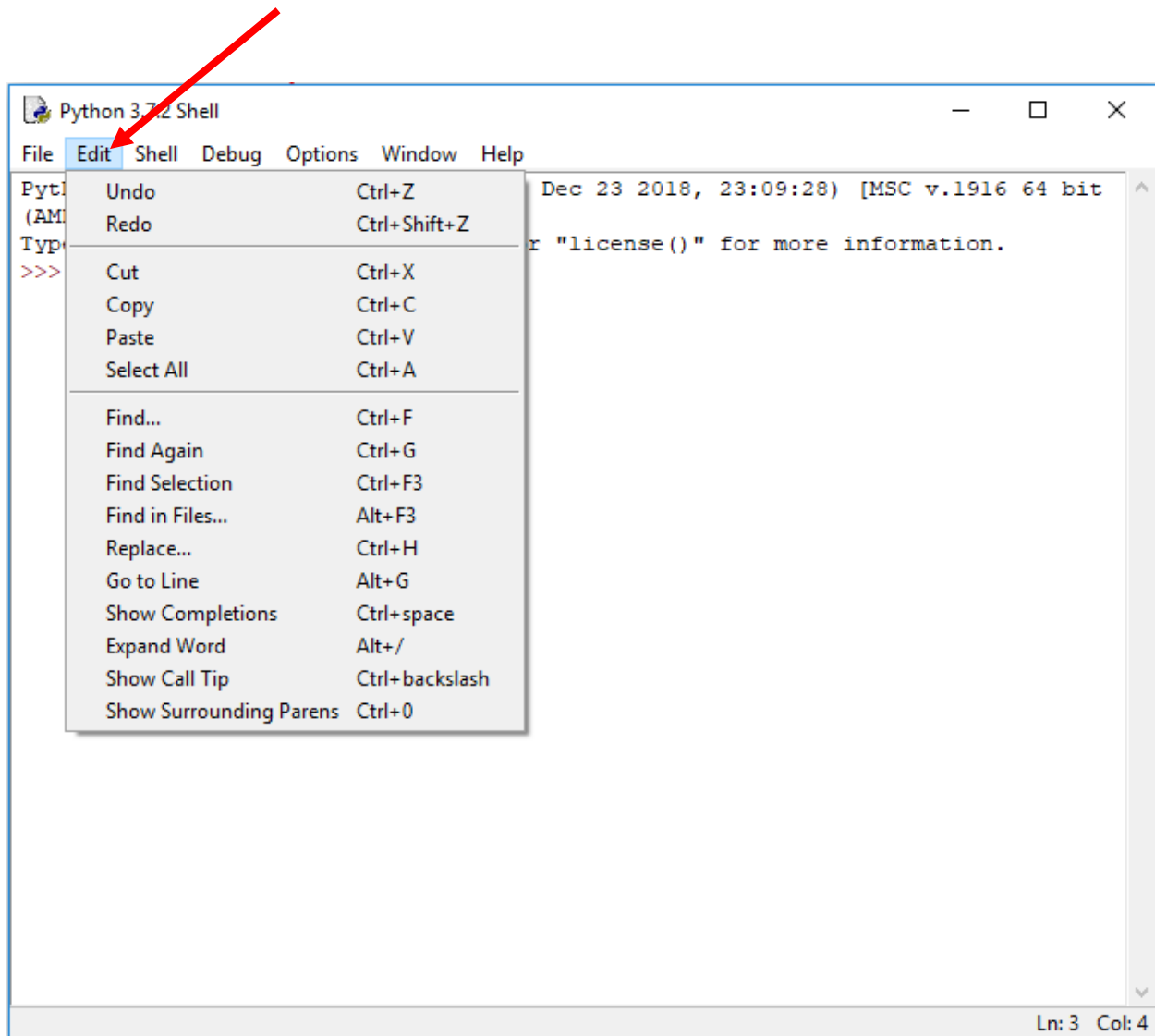




Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Edit**.

These are tools to edit, update, modify etc., the contents of a file.

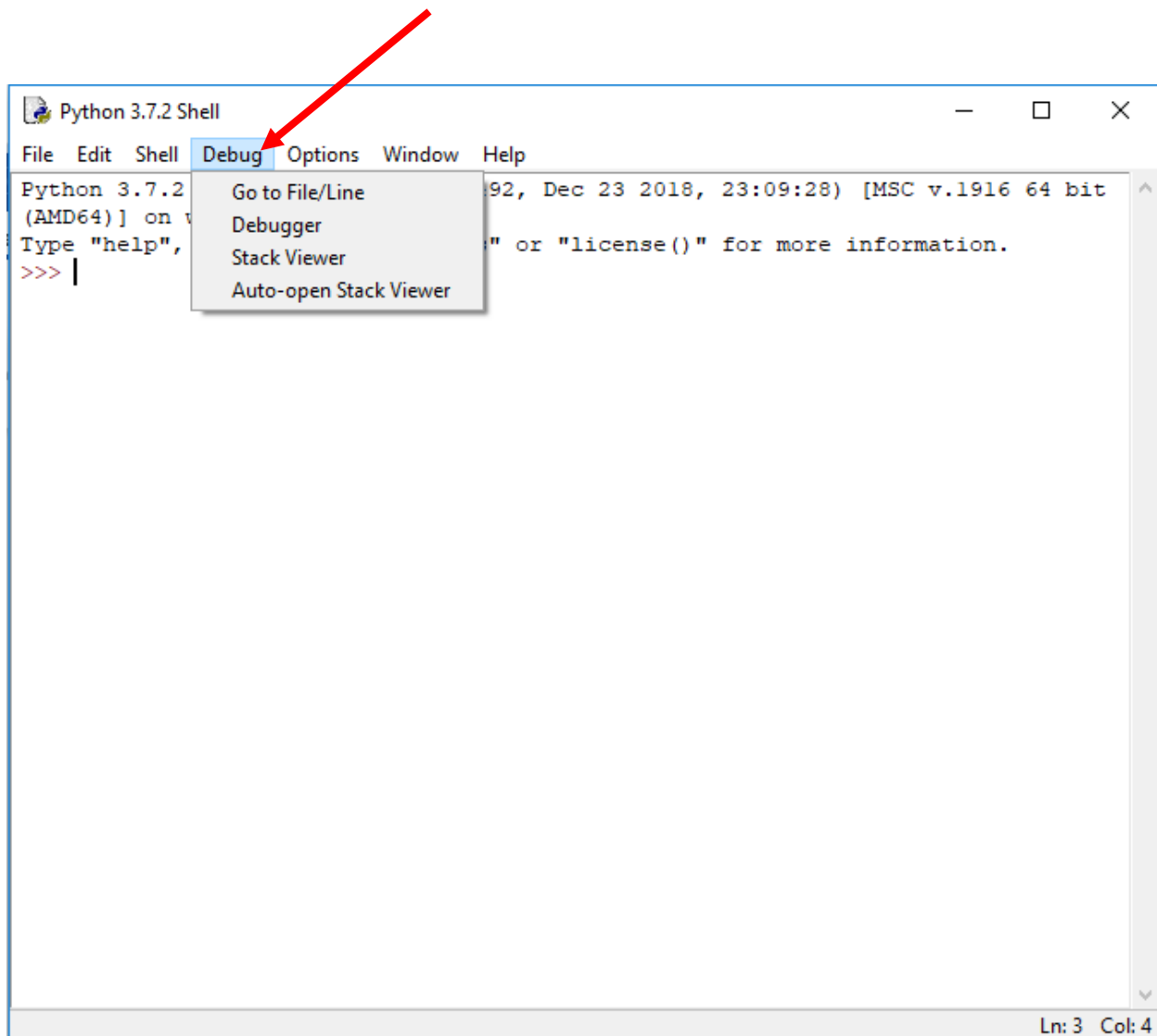




Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Debug**.

Here we find the tools to troubleshoot the code if needed, which is almost always, even for seasoned programmers. These tools assist the programmer to identify, isolate, and fix errors in the code commonly called “bugs.”

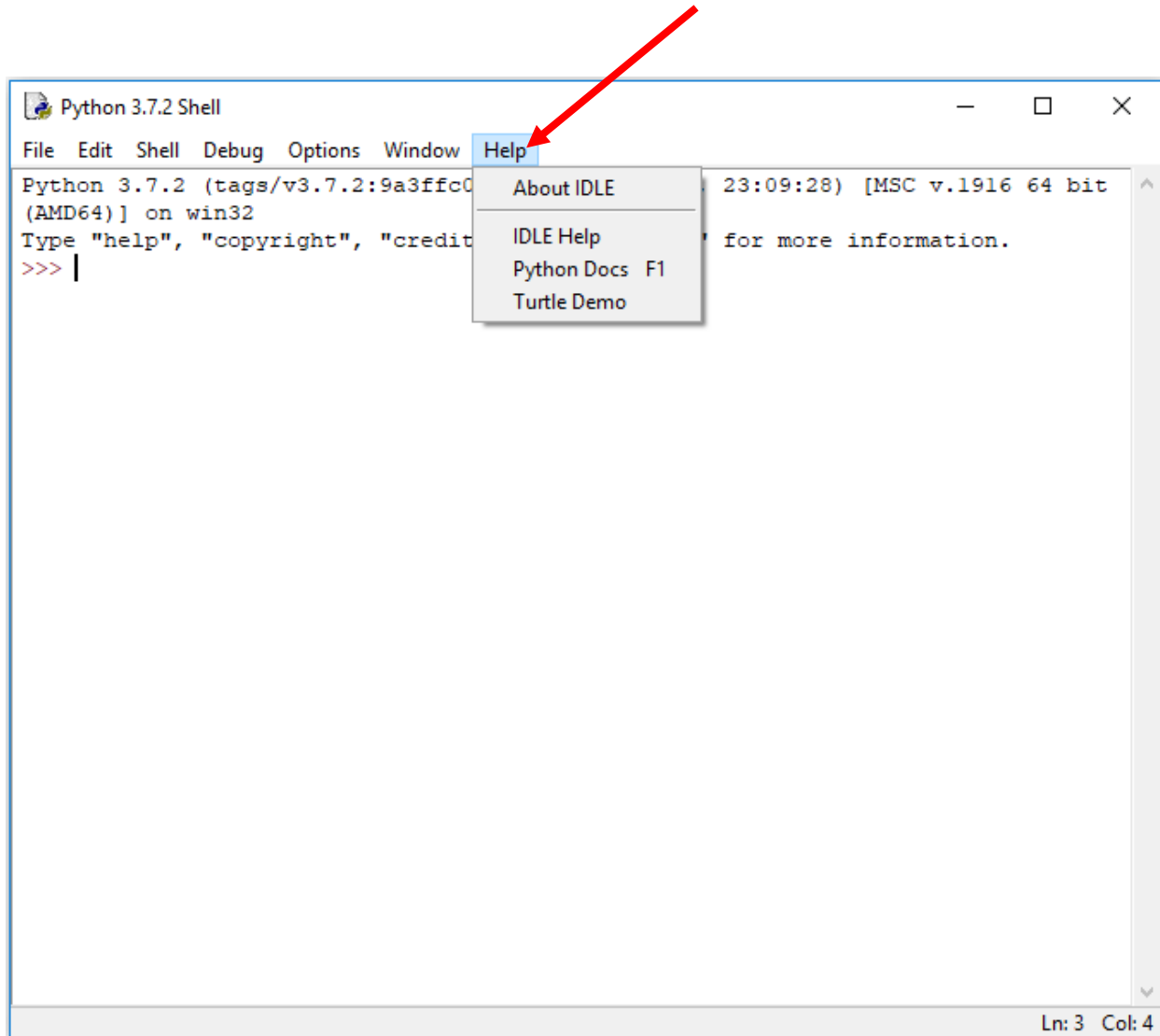




Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Help**.

The Help menu provides links to *Python* documents, and interactive help and search by topic.





Introduction to Computer Programming in Python
A SunCam online continuing education course

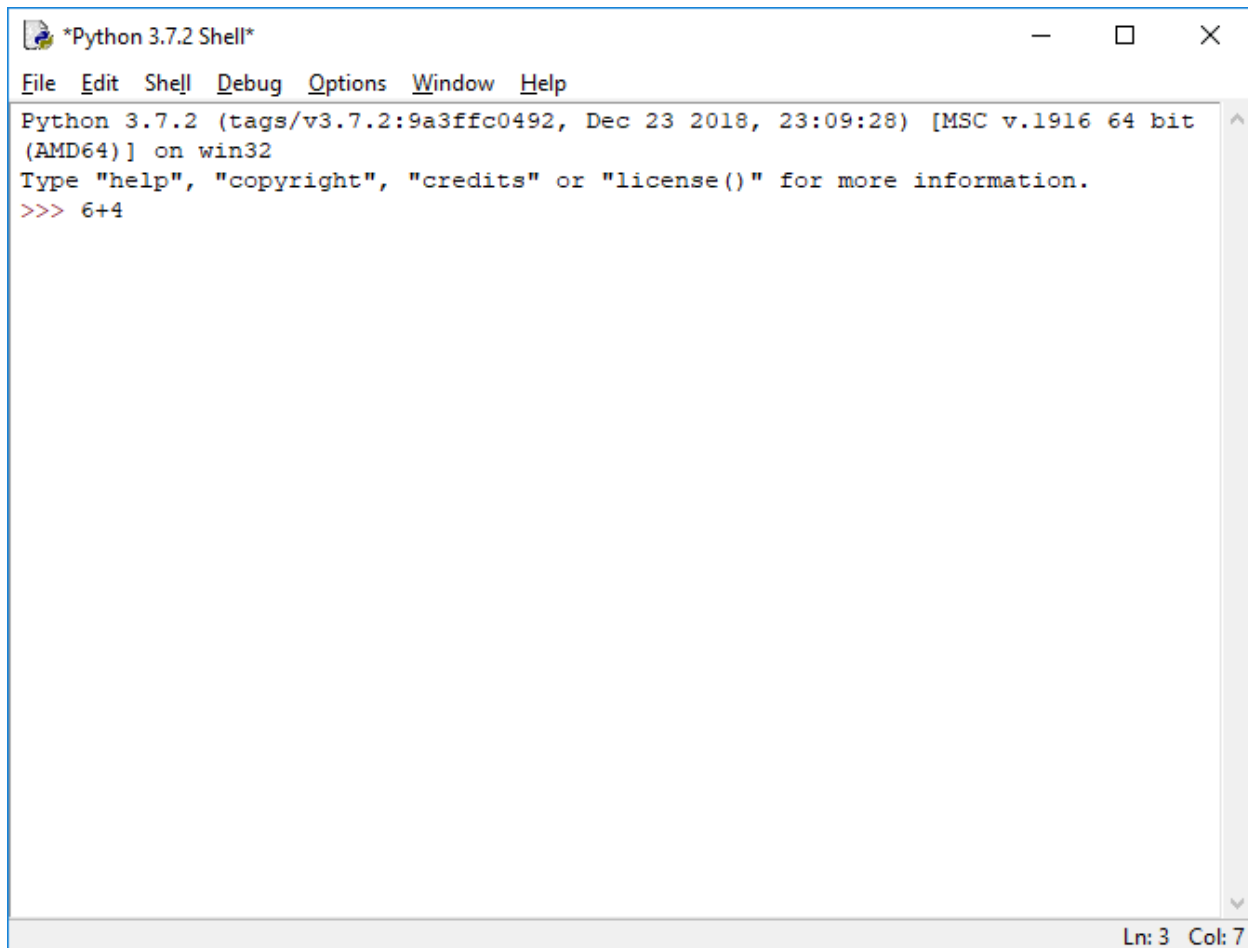
3. EXPRESSIONS, DATA TYPES AND VARIABLES

3.1 Expressions

The fundamental *Python* instruction is called an **expression**. An expression consists of values and an operator(s). An expression always evaluates or reduces to a single a value. If an instruction does not reduce to a single value it is called a **statement**.

Open IDLE (Python GUI).

Enter the following expression at the triple angle bracket prompt, which is the Command Line.

A screenshot of a Python 3.7.2 Shell window. The window title is "*Python 3.7.2 Shell*". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main text area shows the Python version and build information: "Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32". Below this, it says "Type 'help', 'copyright', 'credits' or 'license()' for more information." The prompt ">>>" is followed by the expression "6+4". The status bar at the bottom right shows "Ln: 3 Col: 7".

```
*Python 3.7.2 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 6+4
Ln: 3 Col: 7
```



Introduction to Computer Programming in Python
A SunCam online continuing education course

Hit **Enter** on the keyboard.

The expression evaluates (reduces to a single value).

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> 6+4  
10  
>>> |
```

The status bar at the bottom right indicates "Ln: 5 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

Repeat the procedure to replicate the following arithmetic calculations, one line at a time.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 6+4
10
>>> 15-8
7
>>> 8*3
24
>>> 108/9
12.0
>>> 5**2
25
>>> (2+1)**(5-1)
81
>>>
```

The status bar at the bottom right of the window shows "Ln: 15 Col: 4".

The latter two expression involve “**” which is exponentiation. The “regular” exponentiation keys on the keyboard, **Shift+6**, will not perform exponentiation in *Python*.



Introduction to Computer Programming in Python
A SunCam online continuing education course

3.2 Arithmetic Operators

The *Python* arithmetic operators are summarized in Table 3.1.

Table 3. 1: Arithmetic operators

Operator	Name	Description
+	addition	
-	subtraction	
*	multiplication	
/	division	
**	exponentiation	
%	modulo	the remainder of a division
//	floor division	rounds down to the nearest integer

3.3 Order of Operations

The order in which arithmetic operations are conducted in *Python* is as follows:

1. Parentheses – expressions within parenthesis are always evaluated first
2. Exponentiation
3. Multiplication/ division – multiplication and division have the same precedence and will always be conducted before addition/ subtraction
4. Addition/ subtraction – addition and subtraction have the same precedence and are evaluated from left to right

For long or complex arithmetic expressions where there may be doubts or the programmer would like to control the calculations in some desired manner, simply add parentheses as needed to control the calculations.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Replicate the following calculations in IDLE (Python GUI).

Remember to hit the **Enter** key on your keyboard at the end of each command line.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the following output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> 8+6-9*7-6/8  
-49.75  
>>> 8+(6-9)*7-(6/8)  
-13.75  
>>> (8+6-9)*(7-6)/8  
0.625  
>>> |
```

The status bar at the bottom right indicates "Ln: 9 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

Replicate the following.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 8+6-9*7-6/8
-49.75
>>> 8+(6-9)*7-(6/8)
-13.75
>>> (8+6-9)*(7-6)/8
0.625
>>> 4^2
6
>>> 8^2
10
>>> (8+6-9)*(7-6)/
SyntaxError: invalid syntax
>>> |

```

Ln: 15 Col: 4

Python does not understand the expression, or the user has entered instructions that violate some syntax rule or other rule of *Python*. In this case there is a denominator missing. The program throws an error message. For new programmers who may not have a clue what to do, this may cause some panic. However, error messages are a common occurrence and not a cause for alarm.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Simply retype your expression correctly and re-run it.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 8+6-9*7-6/8
-49.75
>>> 8+(6-9)*7-(6/8)
-13.75
>>> (8+6-9)*(7-6)/8
0.625
>>> 4^2
6
>>> 8^2
10
>>> (8+6-9)*(7-6)/
SyntaxError: invalid syntax
>>> (8+6-9)*(7-6)/4
1.25
>>> |

```

Ln: 17 Col: 4

Error messages of some type are regularly triggered or encountered by even seasoned programmers. Error messages and how to address them will be covered in detail later in this course series.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Close out of the program.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell" and it has standard Windows window controls (minimize, maximize, close). A red arrow points to the close button (an 'X' in a square). The shell displays the following text:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 8+6-9*7-6/8
-49.75
>>> 8+(6-9)*7-(6/8)
-13.75
>>> (8+6-9)*(7-6)/8
0.625
>>> 4^2
6
>>> 8^2
10
>>> (8+6-9)*(7-6)/
SyntaxError: invalid syntax
>>> (8+6-9)*(7-6)/4
1.25
>>> |
```

The status bar at the bottom right shows "Ln: 17 Col: 4".

Later in this course series we shall learn the numerous options available to save our work.



Introduction to Computer Programming in Python
A SunCam online continuing education course

3.4 Data Types

As the phrase suggests, **data type** refers to the type or category of data. The basic data types in *Python* are:

- Integer
- Floating Number (or Floating-Point Number)
- Complex Number
- String (or String Literal)
- Boolean
- Built-in Function

Integer (or **int**), floating number (or **float**), and complex numbers (or **complex**) are collectively referred to as the **numeric types** in *Python*.

Integers

An int is a positive or negative whole number, with no decimals, of any length. An int may be of unlimited length, only constrained by the amount of memory on the user's computer.

Floating Numbers

A float is a positive or negative number that contains at least one decimal. Alternately, the character "e" (or "E") followed by a positive or negative integer may be appended to express the float in scientific notation, where "e" (or "E") represents a power of 10.

Complex Numbers

A complex number is specified as < real part > + < imaginary part > j. E.g. $2 + 7j$

Strings

A string literal (or **str** type) is a sequence of character data. A str may be of any length, constrained only by the amount of memory on the user's computer. String literals are delimited by either single or double quotation marks. Any character with the quotation marks is part of the string. E.g., "hello," or "hello world." Note that in the latter example the space between the two "words" is itself a character of the string. We shall discuss strings in much detail later on in this course series.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Boolean

A Boolean type may have one of two values, namely **True** or **False**. In many programming situations an expression may evaluate to a “true” value or a value of “false.”

Built-In Functions

The *Python* interpreter supports numerous “families” of built-in functions. We shall cover these in-depth later in this course series. Some common examples of *Python* built-in functions include:

Function	Description
<i>round</i> ()	rounds a floating-point value
<i>bin</i> ()	converts an integer to a binary string
<i>type</i> ()	returns the type of an object or creates a new type object
<i>len</i> ()	returns the length of an object
<i>open</i> ()	opens a file and returns a file object
<i>print</i> ()	prints to a text stream or the console
<i>eval</i> ()	evaluates a Python expression

[Note that the function calls are case-sensitive]



Introduction to Computer Programming in Python
A SunCam online continuing education course

Open a new session of IDLE (Python GUI).
 Replicate the following tasks.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> round(2.85791)
3
>>> type(2.85791)
<class 'float'>
>>> len("programming is fun")
18
>>> (2+7j)+(7+3j)
(9+10j)
>>> (6.875e9)*(1.27e5)
873125000000000.0
>>>
  
```

round a float
 return the data type of input
 return the length of a string
 yields a value of 18, note that this includes the spaces in the string
 manipulating complex numbers
 working with floats expressed in scientific notation

Ln: 13 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

3.5 Variables

A variable is area of computer memory allocated to hold data of a certain type. A simplistic analogy is a mailbox in a corporate office staff room. The mailbox for each employee must have a unique identification and will be restricted by its physical size to holding a certain kind or type of mail delivery. Once a variable is set up, its content will be the value of the variable. When the variable is **called** for some calculation, its value will be inserted into the calculation process.

In programming, variables are necessary to facilitate calculations and data management that otherwise will be cumbersome, error-prone or inefficient to do by directly using the values.

In *Python*, a variable is created by giving it a name and **assigning** a value to it. The assignment is done by the equals (=) sign. The type of the variable will be the type of the value assigned to it. The variable can now be inserted into some calculation. Once created and assigned a value, the value of the variable can be updated or changed by assigning a new value which will replace or overwrite the existing value. If the new value is of a different data type, the variable will now assume that new data type.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Open a new session of IDLE (Python GUI).
 Replicate the following variable manipulations.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> x = 20
>>> y = 6
>>> 2*x - 4*y
16
>>> rate = 32.50
>>> hours = 28
>>> wages = rate*hours
>>> print(wages)
910.0
>>> wages
910.0
>>> hours = 36
>>> wages = rate*hours
>>> wages
1170.0
>>> rate = 34.65
>>> hours
36
>>> wages
1170.0
>>> rate*wages
40540.5
>>> rate*hours
1247.3999999999999
>>> wages = rate*wages
>>> wages
40540.5
>>> wages = rate*hours
>>> wages
1247.3999999999999
>>>
  
```

← create variable, by making up a name and assigning a value to the name
 ← insert variables into calculation $2x - 4y$
 ← print the value of the variable wages
 ← or call the variable directly to display its value
 ← update the hours
 ← recalculate the wages, re-call the wages
 ← update the rate
 ← call the hours to see current value
 ← call the wages to see current value
 ← call the product of rate and wages, note that this computation is not assigned to any variable
 ← recalculate the wages
 ← call the wages to display current/latest value

Ln: 33 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

3.6 Variable Names

Depending on the context, a programmer may choose simple algebra type variable names (x, y, z, p1, etc.) or more descriptive variables names e.g. int_Num_Students (as in an integer type variable to hold the number of students). Regardless of the preference, in *Python*, the following variable naming rules shall be followed:

1. A variable name must start with a letter or an underscore (_).
2. A variable name can only contain alphanumeric characters (lowercase a through z, uppercase A through Z, the numerals 0 through 9), and the underscore (_).
3. Variable names are case-sensitive. Thus *Address*, *ADDRESS*, and *address* are three different variables and cannot be used interchangeably.
4. Reserved words (or keywords) that designate special functionality in Python cannot be used as variable names. E.g., “for”, “if”, “while”, “True”, “return”. (Review *Python* documentation for a comprehensive list of keywords).

Consider the following examples:

Variable Name	Acceptable?	Comment
myCar1	Yes	
_1MyCar	Yes	
1_MyCar	No	Cannot start with a number
My Car1	No	Space in between is not alphanumeric or underscore
My%Car1	No	% is not alphanumeric or underscore
my_Car1	Yes	
class	No	<i>Python</i> keyword
my1_car	Yes	Be careful not to put a space at the beginning or end of the name



Introduction to Computer Programming in Python
A SunCam online continuing education course

3.7 Variable Name Conventions

As programs get large and complex it may be advantageous to establish a name convention to manage the variables. Following a name convention can make it easier to identify the type of the variable. It will also make the code easier to figure out when it is reviewed by a third party or revisited after an extended period of time. For programs and applications developed by multiple individuals or teams of programmers, adopting a common name convention for variables will provide a common platform for communication, seamless integration and to prevent confusion.

Different programming languages are amenable to different variable name conventions. One common convention is to append a three letter prefix that indicates the data type of that variable, in lower case letters, and the variable name “proper” starting with an uppercase letter. This is called **camel case**.

Consider the following variable name convention:

Data Type	Prefix	Example
Integer	int	intNumber
String	str	strAddress
Boolean	bln	blnApprove
Float	flt	fltAmount

Some programmers prefer **pascal case**, where each “word” starts with uppercase, e.g. StrZipCode. Another style is **snake case** which is all lower case with each “word” separated by an underscore, e.g. flt_market_price.

Variable name conventions are good practice but they are not required, and not applying one does not violate any rules of *Python*. However if a variable name convention is adopted it must be compatible with all *Python* variable naming rules.



Introduction to Computer Programming in Python
A SunCam online continuing education course

4. STRINGS

4.1 Definition

A string (or string literal) is an ordered sequence of character data. The characters are delimited by either single or double quotation marks. E.g. 'hello world.' The **length** of a string is the number of characters the string has. Each character in the string has a unique position, address, or **index** in the string. Individual characters in the string can therefore be accessed directly using a numeric index or **key value**. This process is known as **indexing**.

4.2 String Indexing

An individual character in a string can be manipulated by specifying the string name (the name of the string variable) followed by its index number in straight brackets ([]). The first character in a string has index 0, the next character has index 1, and so on. The index of the last character is therefore the length of the string minus one. Thus, in *Python*, strings use standard **zero-based** indexing.

Consider the string,

```
strMessage = 'hello world'
```

The first character, *strMessage*[0], has a value of 'h', or *strMessage*[0] = 'h'. Thus

```
strMessage[1] = 'e'
```

```
strMessage[2] = 'l', and so on.
```

Note that sixth character, *strMessage*[5] is a blank space. This can be specified as

```
strMessage[5] = ' ' ("empty" quotation marks).
```

Alternately, string indices may be specified using negative numbers such that the last character (on the right-side end) has index -1, the next character to the left has index -2, and so on.

Using **negative indexing** for *strMessage* = 'hello world',



Introduction to Computer Programming in Python
A SunCam online continuing education course

strMessage[-1] = 'd'

strMessage[-2] = 'l'

strMessage[-3] = 'r', and so on.

4.2 String Slicing

Slicing involves extracting a substring from a string. In general, for a string variable,

strVariable[m : n]

specifies the substring starting at index m up to but not including index n.

If the former index is omitted,

strVariable[: n]

specifies the substring starting at index 0 up to but not including index n.

If the latter index is omitted,

strVariable[m :]

specifies the substring starting at index m up to **and including** the last index.

Adding a third index in the slicing call specifies a “jump”, or “step” or “skip” to the next index to extract.

The code,

strMessage = 'hello world'

strMessage[2 : 8 : 2]



Introduction to Computer Programming in Python A SunCam online continuing education course

specifies a substring starting at index 2, jumping 2 steps to index 4, jumping 2 steps to index 6, and so on up to but not including index 8. Thus

```
strMessage[ 2 : 8 : 2 ] = 'low'
```

Slicing can also be accomplished using negative indexing.

4.3 String Operators

The + Operator

The + Operator **concatenates** two strings together and returns a string of the two **operands** “welded” together.

The * Operator

The * Operator creates multiple copies of a string.
For a string *strString*, and an integer *n*,

```
n * strString  
or  
strString * n
```

will return a string with *n* concatenated copies of the string operand, where *n* is typically a positive integer. However *n* may be zero or a negative integer, in which case an empty or “blank” string (“”) will be returned.

The in Operator

The *in* Operator returns a True or a False if a first string operand is contained within a second string operand. For example,

```
'hello' in 'hello world'
```

will return the Boolean value *True*.

The not in Operator

The *not in* Operator returns a True or a False if a first string operand is not contained within a second string operand.



Introduction to Computer Programming in Python
A SunCam online continuing education course

4.4 Built-in String Functions

The *Python* interpreter has many built-in functions for manipulating strings. A built-in function is a callable procedure that is invoked to perform a specific task(s). In this section we shall review a very limited selection of built-in string functions, for demonstrative purposes. Please consult the *Python* literature for a comprehensive review of the extensive built-in functions available in *Python*.

Function	Description
<i>ord</i> ()	converts a character to an integer
<i>chr</i> ()	converts an integer to a character
<i>str</i> ()	returns a string representation of an object
<i>len</i> ()	returns the length of a string

4.5 Built-in String Methods

A method is a special type of callable procedure that is associated with an object. Remember that in *Python* every data item is an object. Like a function, a method is called to perform a specific task, however it is invoked on a specific object and has knowledge of its target object during execution.



Introduction to Computer Programming in Python
A SunCam online continuing education course

In this section we shall review a very limited selection of string methods. Please consult the *Python* literature for a comprehensive review of the extensive methods available in *Python*.

Method	Description
<i>s.capitalize ()</i>	capitalizes an input string <i>s</i>
<i>s.count (<sub>[, <start>[, <end>]])</i>	counts the number of occurrences of a substring in a target string <i>s</i>
<i>s.isalnum ()</i>	determines whether a string <i>s</i> consists of alphanumeric characters, returns a Boolean
<i>s.lstrip ([<chars>])</i>	trims leading characters from a string <i>s</i>
<i>s.strip ([<chars>])</i>	strips characters from the left and right ends of a string <i>s</i>
<i>s.replace (<old>, <new>[, <count>])</i>	replaces each occurrence of a substring within a string <i>s</i>
<i>s.partition (<sep>)</i>	divides a string <i>s</i> based on a separator
<i>s.join ()</i>	concatenates strings



Introduction to Computer Programming in Python
A SunCam online continuing education course

Open a new session of IDLE (Python GUI).

Replicate the following variable manipulations. Take note of any error message(s).

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> strReport = 'Python Programming for Engineers'
>>> strReport[0]
Traceback (most recent call last):
  File "<pysHELL#1>", line 1, in <module>
    strReport[0]
NameError: name 'strReport' is not defined
>>> strReport[0]
'P'
>>> strReport[1]
'y'
>>> strReport[-9]
'E'
>>> strReport[4:12]
'on Progr'
>>> strReport[:13]
'Python Progra'
>>> strReport[15:]
'ing for Engineers'
>>> strReport[7:18:2]
'Pormig'
>>> strReport[-3:-18:3]
''
>>> strReport[-18:-3:3]
'mgoEi'
>>> strReport[-3:-18:-3]
'eiEog'
>>> |

```

create string variable
 call character at Index[0]
 Oops! misspelled variable name, try again, be careful
 call character at Index[0]
 call character using negative index
 call characters at Index[4] though Index[11]
 call characters at Index[0] though Index[12]
 call characters at Index[15] though last index
 characters at Index[7] though Index[17] every 2 characters
 using negative indices, from Index[-13] through Index[-18] in steps of 3 to the right
 from Index[-18] though Index[-3] in steps of 3 to the right
 from Index[-3] though Index[-18] in steps of 3 to the left

Ln: 29 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

4.6 Modifying Strings

In Python, strings are **immutable**, that is they cannot be modified. That is the individual characters of the string cannot be changed. In order to “modify” a string, a copy of the original string must be made, with the desired changes incorporated.

Continuing your session of IDLE (Python GUI), replicate the following.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>> strReport.replace('Engineers','Geeks')
'Python Programming for Geeks'
>>> strReport.partition('for')
('Python Programming ', 'for', ' Engineers')
>>> strReport[31] = ' and Scientists'
Traceback (most recent call last):
  File "<pyshell#33>", line 1, in <module>
    strReport[31] = ' and Scientists'
TypeError: 'str' object does not support item assignment
>>> strReport = strReport + ' and Scientists'
>>> print (strReport)
Python Programming for Engineers and Scientists
>>>
>>>
>>>
>>>
>>>
>>>
>>>
Ln: 86 Col: 4

```

replace the former substring with the latter

partition string based on a substring

overwrite or update an index

Cant do it! Remember, immutable!

we can make a copy and update that

which is exactly what the *replace()* function does

Close your session of IDLE (Python GUI).



Introduction to Computer Programming in Python
A SunCam online continuing education course

4.7 Formatted String Literal

A formatted string literal, often referred to as an **f-string**, is a string that contains expressions inside curly brackets ({ }).

Open a new session of IDLE (Python GUI).

Conduct the following manipulation of a **standard string**.

```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>>
>>> 'Pythom Programming for Engineer ' +'Is Fun !'
'Pythom Programming for Engineer Is Fun !'
>>>
>>>
>>>
>>> |
```

Ln: 47 Col: 4

We shall now replicate the above manipulation using an f-string as follows:

```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>> x = 'Is Fun !'
>>> f'Python Programming for Engineers {x}'
'Python Programming for Engineers Is Fun !'
>>>
>>>
>>>
```

Ln: 42 Col: 4

f-string, calling x in the curly brackets

This process of calling the variable within the curly brackets is called **variable interpolation**.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Another example.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell" and it has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area shows a Python interactive session with the following code and output:

```
>>>  
>>> a = 20  
>>> b = 15  
>>> z = a*b  
>>> f'the prodcut of {a} and {b} is {z}'  
'the prodcut of 20 and 15 is 300'  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>
```

The status bar at the bottom right shows "Ln: 74 Col: 4".

Close IDLE (Python GUI).



Introduction to Computer Programming in Python
A SunCam online continuing education course

5. LISTS AND TUPLES

5.1 Lists

A list is an ordered **collection** of items. A collection is analogous to an array. A collection with only one (1) item is called a **singleton**.

The items (or elements, or objects) of a list are changeable or **mutable**. Lists are also **dynamic**, that is, items may be added (or appended or prepended) or removed to expand or shrink the size of the list respectively. The size of a list is limited only by your computer's memory. The items of a list may be arbitrary, that is, the items may be of the same data type or they may be of different data types.

A list is defined by enclosing a comma-separated sequence of elements in straight brackets ([]). The elements in a list may be accessed or manipulated by referring to the element's index number. As with strings, list indexing is zero-based.

Many of the Python string operators as well as built-in functions for strings can be used with lists in analogous ways.



Introduction to Computer Programming in Python
A SunCam online continuing education course

5.2 Manipulating a List

Open a new session of IDLE (Python GUI).
Replicate the following list manipulations.

A screenshot of a Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell" and it has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area contains a series of Python commands and their outputs. A red arrow points from the text "checks if A and B are identical, returns a boolean" to the line "A == B".

```
>>>
>>>
>>>
>>>
>>> A = [1, 6, 7, 9, 3]
>>>
>>> B = [4, 3, 8, 2, 5]
>>>
>>> A == B ← checks if A and B are identical, returns a boolean
False
>>>
>>> A is B
False
>>>
>>>
>>> S = ['FL', 'TX', 'NY', 'MN', 'CO']
>>>
>>> T = ['AZ', 'OR', 'FL', 'TN', 'OH']
>>>
>>> D = [33.25, 'FL', 8, 15, 'NV']
>>>
>>> E = [-7, 'FL', 8.975, 'FL', True]
>>>
```

Ln: 116 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> A[3] ← call a list element by index
9
>>>
>>> S[-2]
'MN'
>>>
>>> T[1:3] ← call a slice of a list
['OR', 'FL']
>>>
>>> D[1:4:2] ← call a slice with steps/ strides
['FL', 15]
>>>
>>> F = A[1:3] + D[2:] ← concatenate slices
>>> f
Traceback (most recent call last):
  File "<pyshell#79>", line 1, in <module> ← Oops! Typo!
    f
NameError: name 'f' is not defined
>>> F
[6, 7, 8, 15, 'NV']
>>>
>>> B[::-1] ← reverse order of elements of list
[5, 2, 8, 3, 4]
>>>
>>>
>>>
>>>
>>> |
Ln: 123 Col: 4

```

It is worth noting for strings, the `[:]` syntax returns a reference to the string, whereas for a list it returns a copy of the list.



Introduction to Computer Programming in Python
A SunCam online continuing education course

In your current session of IDLE (Python GUI), continue by replicating the following list manipulations.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> D
[33.25, 'FL', 8, 15, 'NV']
>>>
>>> D[3] = 'WY' ← insert new element at specified index to augment the list
>>>
>>> D
[33.25, 'FL', 8, 'WY', 'NV'] ← new element inserted into list
>>>
>>> B[-1] = 35 ← insert new element by negative index
>>>
>>> B
[4, 3, 8, 2, 35]
>>>
>>> del B[2] ← delete element at specified index
>>>
>>> B
[4, 3, 2, 35] ← elements shift and list shrinks
>>>
>>> B[1:3] = [3.5, 9, 4.2] ← insert slice at specified indices
>>>
>>> B
[4, 3.5, 9, 4.2, 35] ← list is augmented / expanded
>>>
>>> B[2] = [1.95, 2.24, 'GA'] ← insert nested list
>>>
>>> B
[4, 3.5, [1.95, 2.24, 'GA'], 4.2, 35]
>>>
>>> D[3:3] = [5.65, 7.35] ← insert slice at specified index
>>>
>>> D
[33.25, 'FL', 8, 5.65, 7.35, 'WY', 'NV'] ← list expands
>>>
>>> D[:2] = [] ← delete slice
>>>
>>> D
[8, 5.65, 7.35, 'WY', 'NV'] ← list shrinks
>>>
Ln: 197 Col: 4

```




Introduction to Computer Programming in Python
A SunCam online continuing education course

5.3 List Methods

In this section we shall review a very limited selection of other list methods. Please consult the *Python* literature for a comprehensive review of the extensive methods available in *Python*.

Method	Description
<i>a.append</i> (<obj>)	adds an object at the end of the list a
<i>a.clear</i> ()	removes all the elements from the list a
<i>a.copy</i> ()	returns a copy of the list a
<i>a.count</i> (<obj>)	returns the number of elements in the list a with the specified value
<i>a.extend</i> (<iterable>)	takes the elements of a list, and appends them to the end of the current list a
<i>a.pop</i> ()	returns the last item in the list a, and then removes the item from the list
<i>a.pop</i> (index)	returns the value at the index, and then removes the item at the index from the list a
<i>a.insert</i> (<index>, <obj>)	adds/ inserts an object at the specified index in list a
<i>a.index</i> (<obj>)	returns the index of the first occurrence of the specified value
<i>a.remove</i> (<obj>)	removes the item from list a that has the specified value
<i>a.reverse</i> ()	reverses the order of the list items
<i>a.sort</i> ()	sorts the list



Introduction to Computer Programming in Python
A SunCam online continuing education course

In your current session of IDLE (Python GUI), continue by replicating the following list methods.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>>
>>> S
['Tallahassee', 15, 'FL', 'TX', 'NY', 'MN', 'CO']
>>>
>>> S.append('Abilene')
>>>
>>> S
['Tallahassee', 15, 'FL', 'TX', 'NY', 'MN', 'CO', 'Abilene']
>>>
>>> S.insert(4, 'Grand Junction') ← insert element at specified index
>>>
>>> S
['Tallahassee', 15, 'FL', 'TX', 'Grand Junction', 'NY', 'MN', 'CO', 'Abilene']
>>>
>>> S.remove('MN') ← delete matching element
>>>
>>> S
['Tallahassee', 15, 'FL', 'TX', 'Grand Junction', 'NY', 'CO', 'Abilene']
>>>
>>> S.pop(1) ← returns element at specified index, and
15 ← deletes element from list
>>>
>>> S
['Tallahassee', 'FL', 'TX', 'Grand Junction', 'NY', 'CO', 'Abilene']
>>>
>>>
>>>
>>> |
Ln: 281 Col: 4

```

You may close out of IDLE (Python GUI).



Introduction to Computer Programming in Python
A SunCam online continuing education course

5.4 Tuples

A tuple is another type of ordered collection of items. All aspects and properties of a tuple are the same as those of a list except for the following:

- a) A tuple is defined by enclosing the elements in parentheses (“()”).
- b) A tuple is immutable (unchangeable).

Although the parentheses are used to define a tuple, the elements in a tuple are accessed by referring to the element’s index number enclosed in straight brackets ([]), in a similar manner as with strings and lists.

There are advantages to using tuples in a program, such as:

- a) Using a tuple prevents inadvertent modification of a collection when the values are intended to remain constant throughout the life the program.
- b) In large programs, program execution is noticeably faster when manipulating tuples versus other types of collections.

5.5 Manipulating Tuples

Open a new session of IDLE (Python GUI).



Introduction to Computer Programming in Python
A SunCam online continuing education course

Replicate the following dictionary manipulations.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> T = ('Paris', 'Berlin', 'Oslo', 'Copenhagen', 'London')
>>> T[3]
'Copenhagen'
>>> S = 'Nantes', 'Stuttgart', 'Stavanger', 'Aalborg', 'Cardiff'
>>> S[1]
'Stuttgart'
>>> V = T[2], S[-4], 'Munich'
>>> print(V)
('Oslo', 'Stuttgart', 'Munich')
>>> V[0:2]
('Oslo', 'Stuttgart')
>>> S,T
(('Nantes', 'Stuttgart', 'Stavanger', 'Aalborg', 'Cardiff'), ('Paris', 'Berlin',
'Oslo', 'Copenhagen', 'London'))
>>> S,T = T,S
>>> S
('Paris', 'Berlin', 'Oslo', 'Copenhagen', 'London')
>>> T
('Nantes', 'Stuttgart', 'Stavanger', 'Aalborg', 'Cardiff')
>>> S[2] = 'Geneva'
Traceback (most recent call last):
  File "<pyshell#11>", line 1, in <module>
    S[2] = 'Geneva'
TypeError: 'tuple' object does not support item assignment
>>>

```

define tuple

define tuple without parenthesis also works

call an element

call elements to define new tuple

tuple of tuples

swap tuples' elements, one of the popular uses of tuples, no need to create a temporary variable like in other programs

change an element

can't do it! tuples are immutable

Ln: 26 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

6. DICTIONARIES AND SETS

6.1 Dictionary

A dictionary, also known as an **associative array**, is an unordered collection of items that are changeable and indexed. Each item has a unique **key name** (or key) which has an associated value.

A dictionary is defined by enclosing a comma-separated list of key-value pairs within curly brackets ({ }), and separating each key from its associated value with a colon (:).

The general definition of a dictionary is of the form

```
D = {
    <key 1> : <value > ,
    <key 2> : <value > ,
    .
    .
    .
    <key n> : <value >
}
```

For example,

```
xCar = {
    'brand' : 'Mercedez Bens' ,
    'model' : 'X-Class' ,
    'style' : 'SUV'
    'year' : 2018 ,
    'color' : 'silver' ,
    'interior' : 'deluxe'
    'fuel economy' : '25 mpg' ,
    'safety rating' : 'high' ,
    'technology' : 'hybrid'
    'capacity' : 7 ,
```



Introduction to Computer Programming in Python
A SunCam online continuing education course

```

    'conveniences' : 'regular'> ,
    'insurance' : 'moderate' ,
    'maintenance' : 'low'

}

```

Another example.

```

xCustomer = {
    'accountnumber' : '14789640' ,
    'firstname' : 'Robert' ,
    'lastname' : 'Apostolakis' ,
    'birthdate' : 'July 24'
    'ssn' : 1098 ,
    'address' : '1564 Park Avenue' ,
    'city' : 'Ryansville'
    'state' : 'FL'
    'zipcode' : '72310'
    'phone' : '392-678-9114'
}

```

6.2 Manipulating Dictionaries

An item is accessed by calling its key name within straight brackets ([]). The value of an item can be changed by referring to its key name. An item can be added to augment a dictionary by creating a new key and assigning a value to it. Items can be removed to shrink the dictionary by using one of several dictionary methods. An existing item can be modified or updated by assigning a new value to relevant key.

A dictionary key must be of an immutable data type. Therefore, a tuple, for example, can be a key as tuples are immutable. A key name must be unique. Duplicating a key name is treated as updating a pre-existing key and will result in the latter assigned value overwriting the former. The values assigned to keys may be of any data type. Dictionary values do not have to be unique. Thus, there are no restrictions on dictionary values.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Open a new session of IDLE (Python GUI).
 Replicate the following dictionary definitions.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> North = {
    'Duval': 'Jacksonville',
    'Nassau': 'Fernandina Beach',
    'Baker': 'Macclenny',
    'Clay': 'Green Cove Springs'
}
>>>
>>> Central = {
    11: 'Orlando',
    12: 'Sanford',
    13: 'Tavares',
    14: 'Kissimmee'
}
>>>
>>> mCars = dict([
    ('Mercedes Benz', 'S-Class'),
    ('Honda', 'Accord'),
    ('Jeep', 'Patriot'),
    ('Kia', 'Sorrento'),
    ('BMW', 'X6')
])
>>>
>>> Pro_Team = dict(
    Baseball='Jays',
    Soccer='Stormers',
    Basketball='Brumbies',
    Hockey='Crusaders',
    Football='Pirates'
)
>>> |
  
```

← another way to define a dictionary

← and another way

Ln: 32 Col: 4



Introduction to Computer Programming in Python
A SunCam online continuing education course

A dictionary can also be built by incrementally adding items.

In your current session of IDLE (Python GUI).

Replicate the following dictionary definitions.

A screenshot of the Python 3.7.2 Shell window. The window title is "Python 3.7.2 Shell" and it has a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The shell contains the following code:

```
>>>  
>>> nCompany = {}  
>>>  
>>> nCompany['CEO'] = 'Bob'  
>>>  
>>> nCompany['COO'] = 'Kim'  
>>>  
>>> nCompany['MD'] = 'Tran'  
>>>  
>>> nCompany['GM'] = 'Erica'  
>>>  
>>> nCompany  
{'CEO': 'Bob', 'COO': 'Kim', 'MD': 'Tran', 'GM': 'Erica'}  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>  
>>>
```

A red arrow points from the text "call the dictionary" to the output of the last command. The status bar at the bottom right shows "Ln: 71 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

6.3 Dictionary Methods

In this section we shall review a very limited selection of other dictionary methods. Please consult the *Python* literature for a comprehensive review of the extensive dictionary methods available in *Python*.

Method	Description
<i>d.clear</i> ()	empties dictionary d of all key-value pairs
<i>d.get</i> (<key>[, <default>])	returns the value for a key if it exists in the dictionary d
<i>d.keys</i> ()	returns a list of all keys in dictionary d
<i>d.pop</i> (<key>[, <default>])	removes a key from a dictionary d, if it is present, and returns its value
<i>d.update</i> (<obj>)	merges the entries from <obj> into dictionary d
<i>d.values</i> ()	returns a list of all values in dictionary d



Introduction to Computer Programming in Python
A SunCam online continuing education course

Continuing your current session of IDLE (Python GUI).
 Replicate the following dictionary manipulations.

The screenshot shows a Python 3.7.2 Shell window with the following code and output:

```

>>>
>>> North['Duval']
'Jacksonville'
>>>
>>> Central[13]
'Tavares'
>>>
>>> mCars[-13]
Traceback (most recent call last):
  File "<pyshell#49>", line 1, in <module>
    mCars[-13]
KeyError: -13
>>>
>>> Pro_Team.keys()
dict_keys(['Baseball', 'Soccer', 'Basketball', 'Hockey', 'Football'])
>>>
>>> nCompany.values()
dict_values(['Bob', 'Kim', 'Tran', 'Erica'])
>>>
>>> |
  
```

Annotations in the image:

- A red arrow points to the code `North['Duval']` with the text "call item by key".
- A red arrow points to the error message `KeyError: -13` with the text "key does not exist, cannot treat as a list index".

The status bar at the bottom right of the window shows "Ln: 89 Col: 4".



Introduction to Computer Programming in Python
A SunCam online continuing education course

6.4 Sets

A set is a collection of unordered and unindexed items. Set elements are unique, thus duplicate elements are not allowed. A set may be modified, however the elements of the set must be of an immutable data type.

A set can be defined with by enclosing a comma-separate list curly brackets ({ }). The general definition of a set is of the form

$$x = \{ \langle obj1 \rangle, \langle obj2 \rangle, \dots, \langle obj n \rangle \}$$

For example,

$$x = \{ 'Tallahassee' , 'Jacksonville' , 'Pensacola' , 'Orlando' , 'Tampa' , 'Miami' \}$$

The set **constructor** may also be used to define a set, as follows

$$x = \text{set} (\langle iterable \rangle)$$

The argument is an **iterable**. An iterable being a type that generates a list of objects, such as a string, a list, or a tuple.

Examples:

Using a tuple argument,

$$x1 = \text{set} (('Atlanta' , 'Albany' , 'Athens' , 'Columbus' , 'Thomasville' , 'Macon'))$$

Using a list argument,

$$x2 = \text{set} (['Sacramento' , 'Los Angeles' , 'San Diego' , 'San Francisco' , 'Fresno' , 'Pasadena'])$$



Introduction to Computer Programming in Python
A SunCam online continuing education course

6.5 Manipulating Sets

Sets are unordered and unindexed therefore the elements cannot be accessed with indices as can be conducted with other collections. Once a set is created, the items cannot be changed, however items can be added to the set and can be removed from the set. Sets cannot be sliced. Many operations conducted on other collections are not applicable to sets. *Python* provides a host of operations on sets that mimic the operations conducted on mathematical sets. There are also a host of set methods that are available.

6.6 Set Operations and Methods

Set operations may be conducted by using a set operator or a set method. In this section we shall review a very limited selection of other set operations and methods. Please consult the *Python* literature for a comprehensive review of the extensive set operations and methods available in *Python*.

Method	Description
<i>x.add (<elem>)</i>	adds an element to a set x
<i>x.clear ()</i>	removes all elements from set x
<i>x1.difference(x2[, x3 ...])</i>	returns the set of all elements that are in set x1 but not in sets x2, x3,...
<i>x1.difference_update(x2[, x3 ...])</i>	updates set x1 such that it removes the items not common to the all the sets x1, x2, x3, ...
<i>x.discard (<elem>)</i>	removes an element from set x if element is in x
<i>x1.intersection (x2)</i>	returns a set that contains the items that are in both set x1, and set x2
<i>x1.intersection_update(x2[, x3 ...])</i>	updates set x1, retaining only common elements of all the sets x1, x2, x3, ...
<i>x.pop ()</i>	removes and returns an arbitrarily chosen element from set x, throws error message if x is empty



Introduction to Computer Programming in Python
A SunCam online continuing education course

Method	Description
<i>x.remove (<elem>)</i>	removes an element from set x if element is in x, else throws an error message
<i>x1.union (x2[, x3 ...])</i>	returns a set containing the union of sets x1, x2, x3, ...
<i>x1.update (x2[, x3 ...])</i>	updates set x1 with the union of this set and other set(s)

The following operations yield identical results as the corresponding set method.

Method	Operation
<i>x1.difference(x2[, x3 ...])</i>	$x1 - x2 [- x3 \dots]$
<i>x1.difference_update(x2[, x3 ...])</i>	$x1 -= x2 [x3 \dots]$
<i>x1.intersection (x2[, x3 ...])</i>	$x1 \& x2 [\& x3 \dots]$
<i>x1.intersection_update(x2[, x3 ...])</i>	$x1 \&= x2 [\& x3 \dots]$
<i>x1.union (x2[, x3 ...])</i>	$x1 x2 [x3 \dots]$
<i>x1.update (x2[, x3 ...])</i>	$x1 = x2 [x3 \dots]$



Introduction to Computer Programming in Python
A SunCam online continuing education course

Open a new session of IDLE (Python GUI).
 Replicate the following set definitions.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
>>> Contracts_1 = {'Renovation', 'Refurbish', 'Plumbing', 'Electrical', 'Plumbing'}
>>>
>>> Contracts_1
{'Electrical', 'Renovation', 'Plumbing', 'Refurbish'}
>>>
>>> Contracts_2 = set(['Refurbish', 'Paving', 'Electrical'])
>>>
>>> Contracts_2
{'Paving', 'Electrical', 'Refurbish'}
>>>
>>> Contracts_2.add('Demolition')
>>>
>>> Contracts_2
{'Paving', 'Electrical', 'Demolition', 'Refurbish'}
>>>
>>> Contracts_1.intersection(Contracts_2)
{'Electrical', 'Refurbish'}
>>>
>>> Contracts_1.union(Contracts_2)
{'Plumbing', 'Paving', 'Electrical', 'Refurbish', 'Demolition', 'Renovation'}
>>>
>>> Contracts_1.pop()
'Electrical'
>>>
>>> Contracts_1
{'Renovation', 'Plumbing', 'Refurbish'}
>>>
>>> Contracts = Contracts_1.difference(Contracts_2)
>>>
>>> Contracts
{'Renovation', 'Plumbing'}
>>>
>>>
Ln: 128 Col: 4
  
```

unique values only, duplicates not accepted, also items are randomized

definition by the set constructor

randomly selected and removed



Introduction to Computer Programming in Python
A SunCam online continuing education course

7. RUNNING THE PYTHON FILE

7.1 Code Execution

Thus far, all codes have been typed into the IDLE (Python GUI) command line and executed line-by-line. Unfortunately, given the nature of engineering problems, a line-by-line execution of codes will be inefficient and in fact impractical for an engineering practitioner. What is needed is for multiple lines of code - a **script**, to be typed out and then executed under one operation, or for the script file itself to be called and executed.

This chapter presents methods for executing Python scripts, and calling and executing Python files. In this regard, the execution of Python code on a line-by-line basis, as has been presented thus far, shall henceforth be limited to the purposes of testing or experimenting individual commands, or for debugging and troubleshooting, during the development of scripts. This is indeed how line-by-line code execution is used by professional programmers.



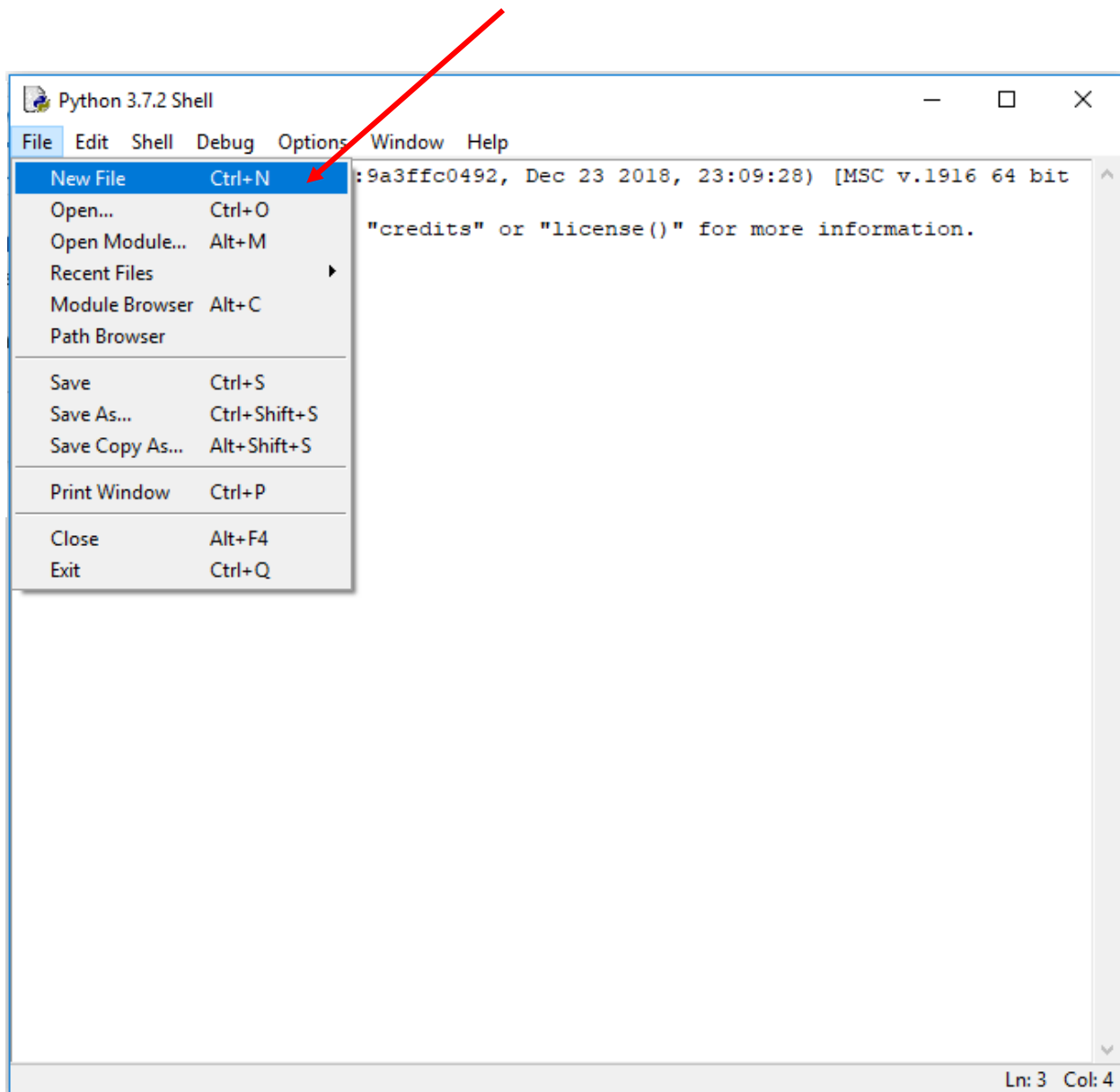
Introduction to Computer Programming in Python
A SunCam online continuing education course

7.2 The File Editor

Open a new session of IDLE (Python GUI).

Click on **File**.

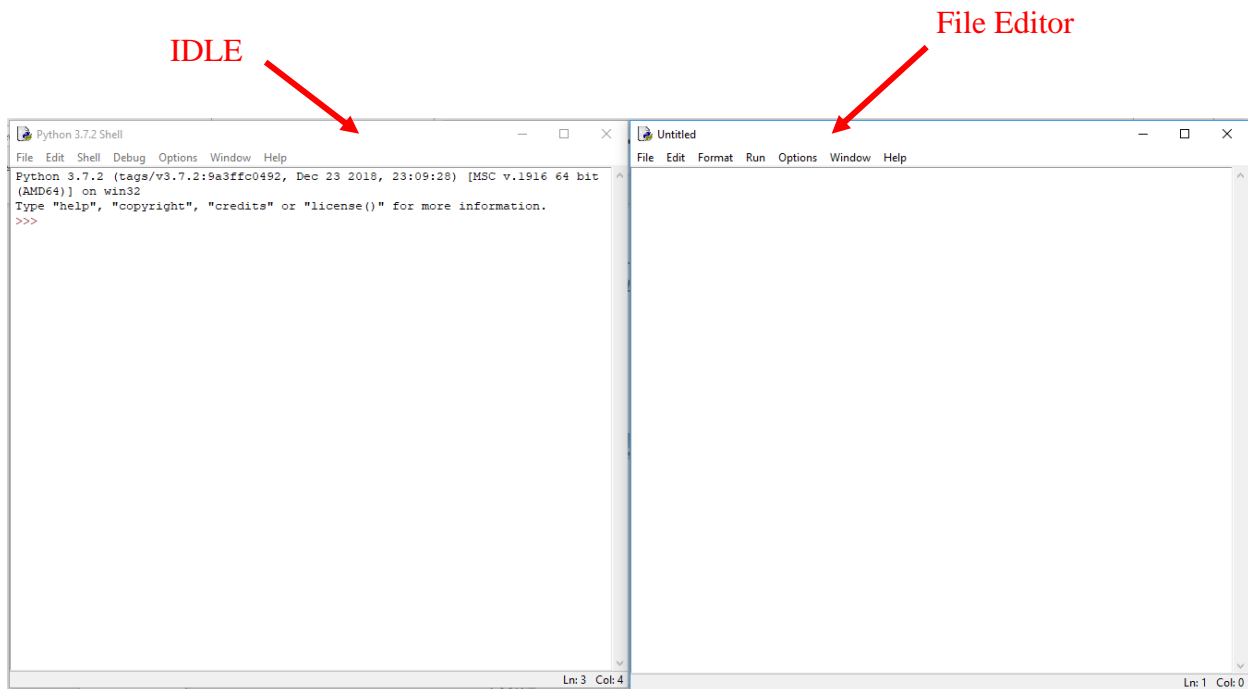
Click on **New File**.





Introduction to Computer Programming in Python
A SunCam online continuing education course

The File Editor opens.



The File Editor is used to build scripts for entire programs whereas in IDLE codes can only be executed line-by-line. The IDLE window and the File Editor may be difficult to distinguish. Note that the IDLE Command Line (the triple angle bracket prompt) appears in IDLE only.

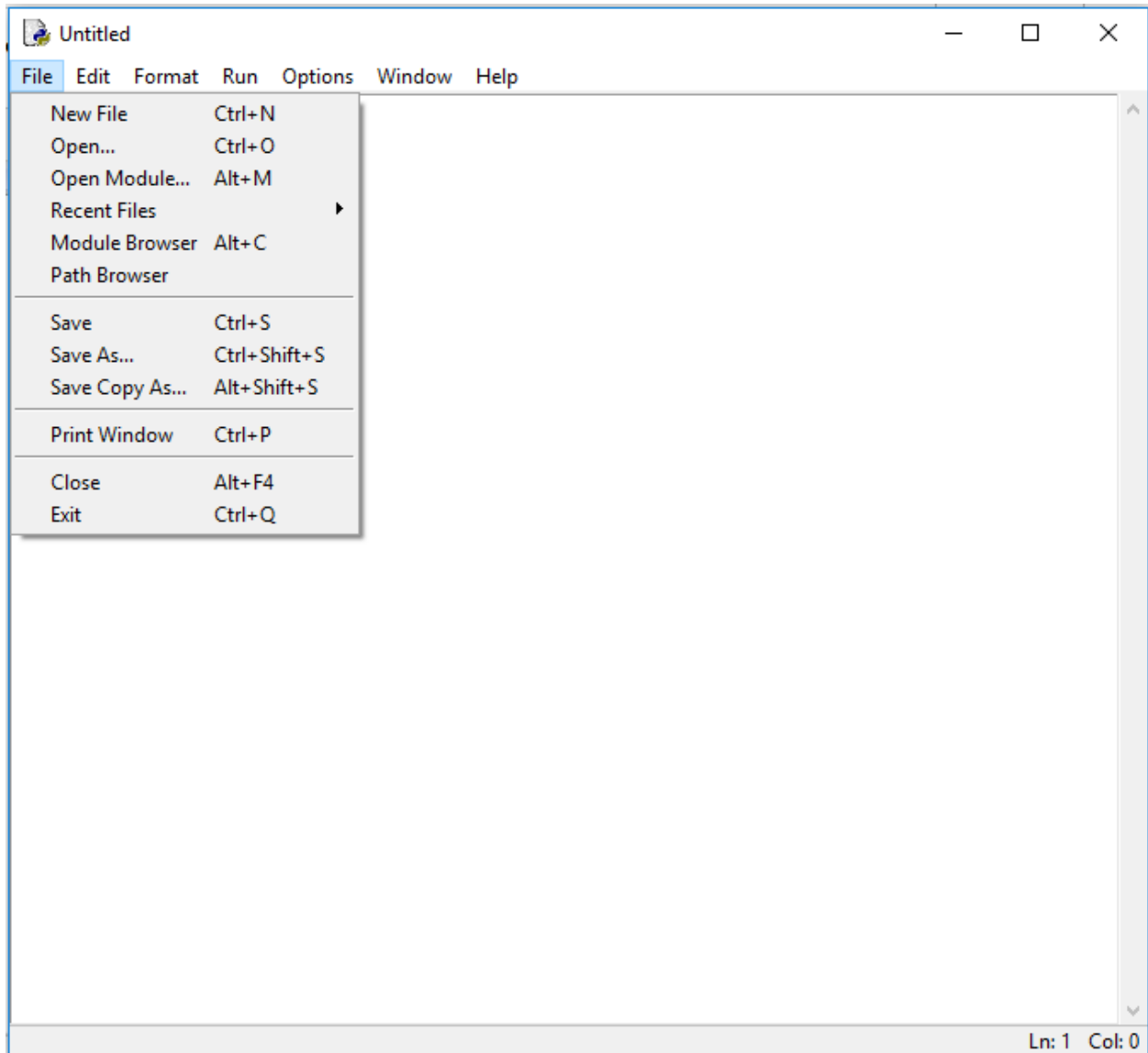
Take a moment to review some of the File Editor drop down menus.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **File**.

These are tools to create manipulate and manage files.

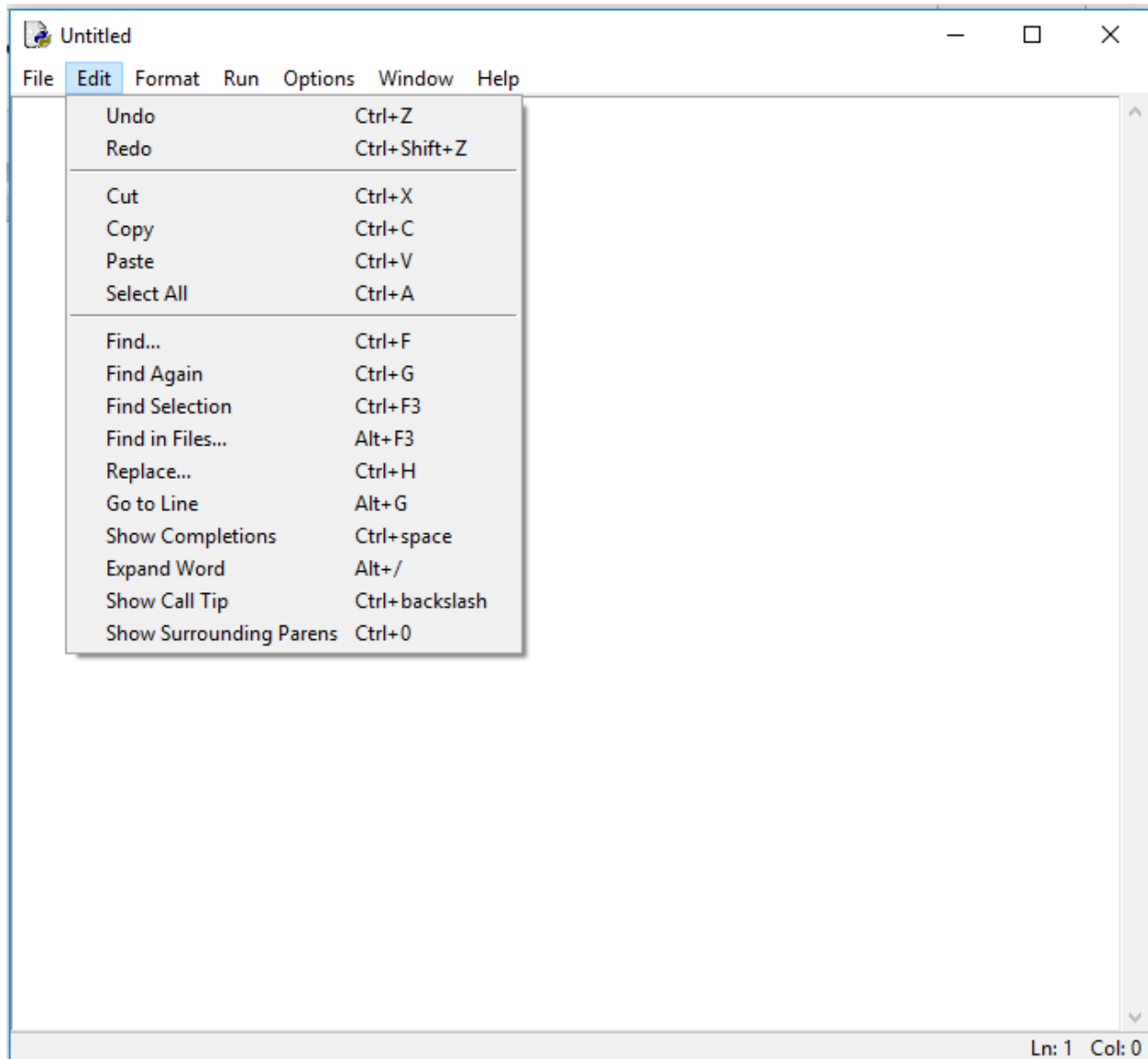




Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Edit**.

These are tools to edit, update, modify, etc., the contents of a file.

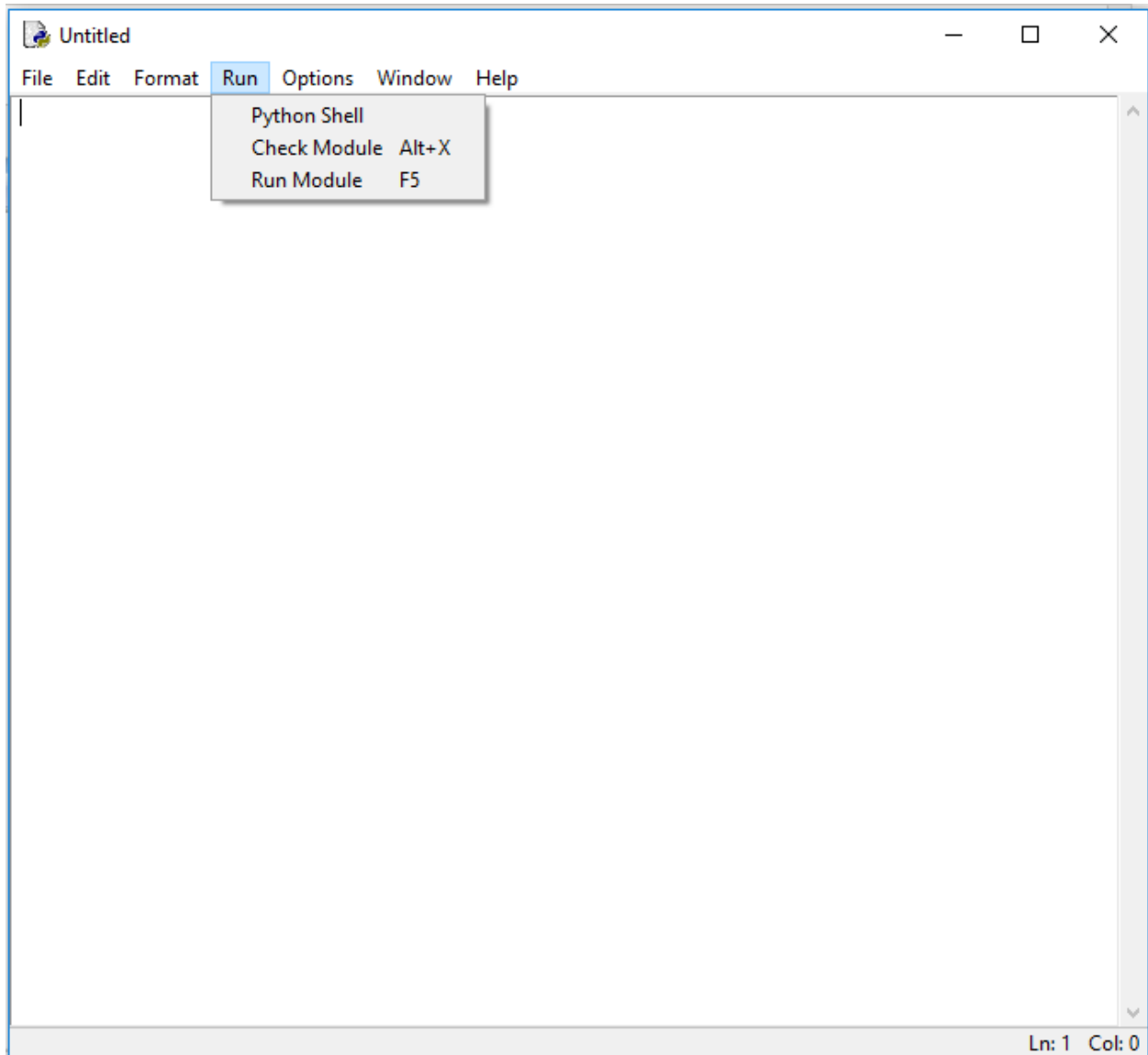




Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Run**.

These are tools to the script or execute the file.



Leave both IDLE and the File Editor open, we shall come back to them.



Introduction to Computer Programming in Python
A SunCam online continuing education course

7.3 Practical Example

Let us consider a parking lot analysis problem encountered by a traffic engineer. The goal is to build an application that calculates the number of parking spaces required for a new retail store.

The parked vehicles can be subdivided into groups based on, for example the parking duration, namely “short-term parking”, “intermediate”, “long-term”, “extended/ all day” etc.

The math is as follows.

Let i denote a parking duration group

Let t_i denote the average parking duration for group i .

Let n_i denote the number of vehicles that come to the parking facility that fall under group i .

For a given parking group (i), the parking demand (D_i) in space-hours will be

$$D_i = n_i t_i$$

One space-hour is parking demand created by the use of one parking space for a period of one hour. Then it follows that the total parking demand (D) in space-hours for all groups will be the summation of the space-hours of the individual parking groups, as follows.

$$D = n_1 t_1 + n_2 t_2 + n_3 t_3 + \dots$$

Let f denote the efficiency factor, which accounts for the space-hours lost due to drivers searching for open parking spaces, turnover, as well as time lost during maneuvering into and out of parking spaces, typical values of f are 0.9 for curbside parking, 0.8 for parking garages, and 0.9 for parking lots.

Let M denote the actual number of physical parking spaces on the site, this is the number of required parking spaces for the development.

Let T denote the hours of operation of the establishment, or the length of time that a vehicle can be legally parked at the parking facility for the sole purpose of participating in the activities of the establishment and therefore exerting a parking demand.



Introduction to Computer Programming in Python
A SunCam online continuing education course

It therefore follows that,

$$D = f * M * T$$

Thus, the number of parking spaces required,

$$M = \frac{D}{f * T}$$

The data for this retail store is as follows:

1. Hours of operation: 8:00am to 10:00pm.
2. Per company procedures, at any point in time there must be 40 employees on site.
3. Based on previous data and projections for this location, the store estimates 2500 vehicle trips to the store per day.
4. In addition the store has an online grocery pickup program where customers shop online and come to the store to pick up their shopping. These online pickup shoppers may park anywhere on the premises and use a mobile app to direct a store employee to bring out their merchandise. The store projects 600 vehicle trips to the store per day due to the online pickup program,
5. The store projects that a regular shopper will park for an average duration of 25 minutes, whereas an online pickup shopper will park on the premises for an average duration of 10 minutes.



Introduction to Computer Programming in Python
A SunCam online continuing education course

In the open File Editor type the following.

```

*Untitled*
File Edit Format Run Options Window Help
# parking calculator
# this app calculates the number of parking spaces
# required to meet the demand
# D = f*M*T
# where D is the parking demand in space-hours
# where M is the number of parking spaces required
# where T is the hours of operation of the business
# where f is the parking efficiency factor

print('What are the number of hours of operation of this business ? ')
T = input() # will stop program for user to respond with data entry
            # user response will stored at variable on the left hand side

print('Enter the number of employees :')
n1 = input()

print('Enter the number of regular parking vehicles expected :')
n2 = input()

print('What is the average regular parking duration in minutes ? ')
t2 = input()

print('Enter the number of vehicles expected due to the online pickup program :')
n3 = input()

print('What is the average online pickup parking duration in minutes ? ')
t3 = input()
|
Ln: 1 Col: 0

```



Introduction to Computer Programming in Python
A SunCam online continuing education course

The `input()` function will stop the program execution and only resumes after the user has entered some data in response and hit the **Enter** button on the keyboard. The data entry is stored to the variable that the `input()` is assigned to on that line. The `input()` function shall be discussed in detail later in this course series.

Note the lines that begin with “#”. These are **comment** lines. These are personal notes and reminders to yourself. When the Python interpreter encounters the “#”, any code to right of it on that line is ignored, the cursor skips to the next line and continues with the program execution. **Commenting** is a basic feature of all programming languages. The different programming languages have different techniques for entering comments. There are no rules governing how much commenting is required or necessary. A common practice is to provide as much comments as necessary for a non–specialist to be able to read the comments alone and understand or at least follow what the program is doing. Another “rule of thumb” is to provide enough comments such that if you were to call out sick tomorrow, a competent coworker can review the comments and continue with the project without the need to start all over.

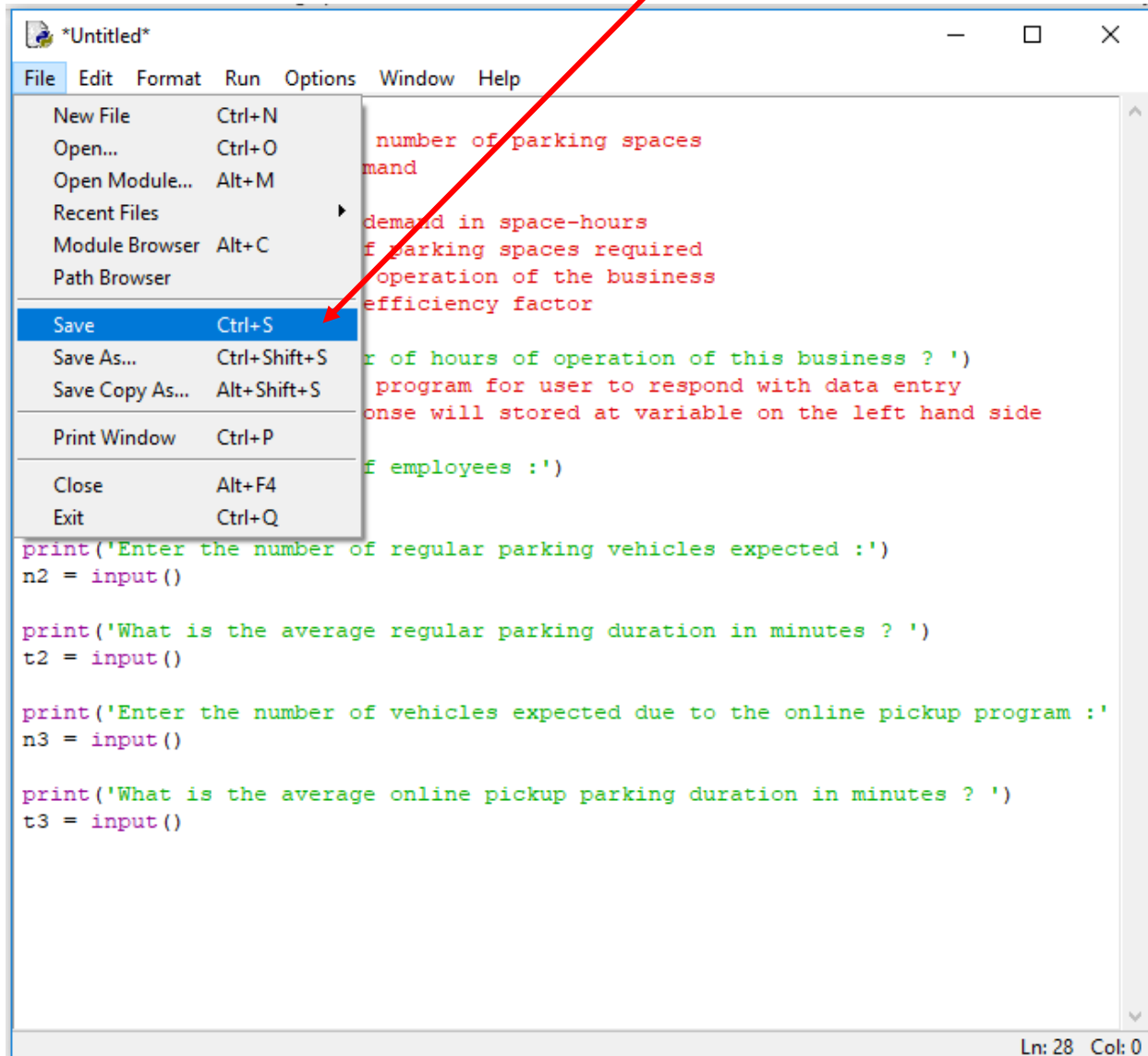
Before going any further let us save the work.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **File**.

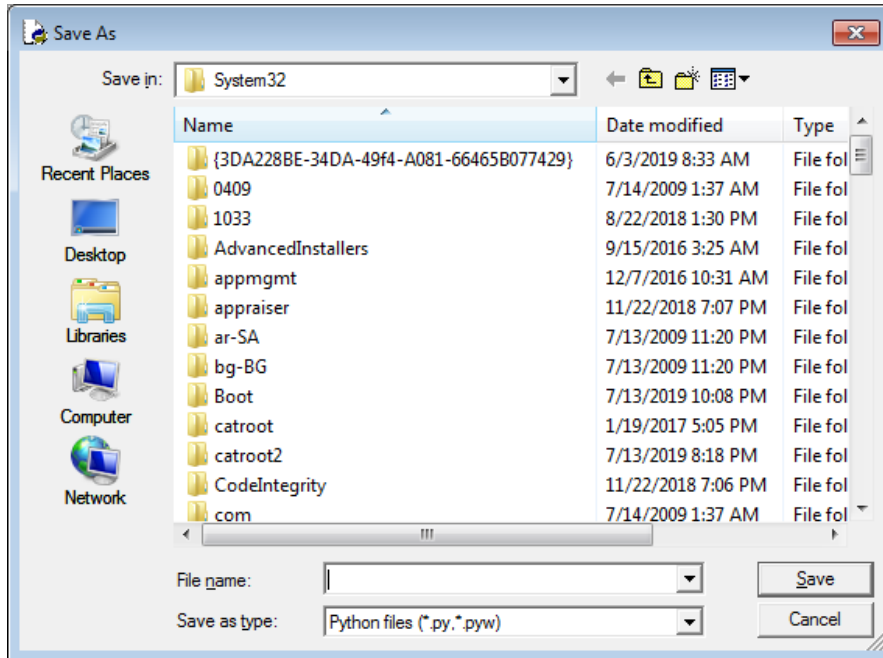
Click on **Save** (or **Save As**).





Introduction to Computer Programming in Python
A SunCam online continuing education course

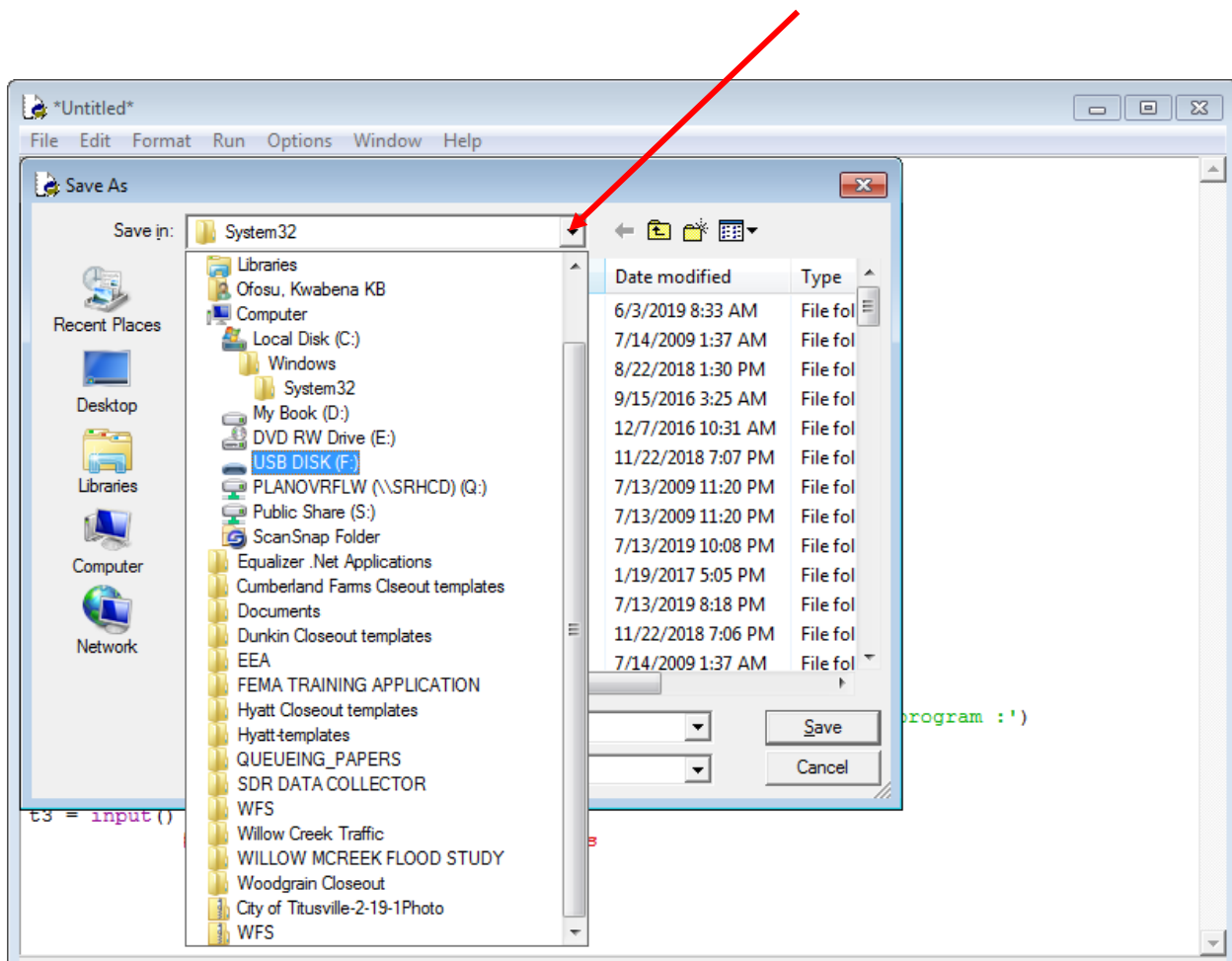
The Folder window opens.





Introduction to Computer Programming in Python
A SunCam online continuing education course

Navigate to a Folder of your choice.



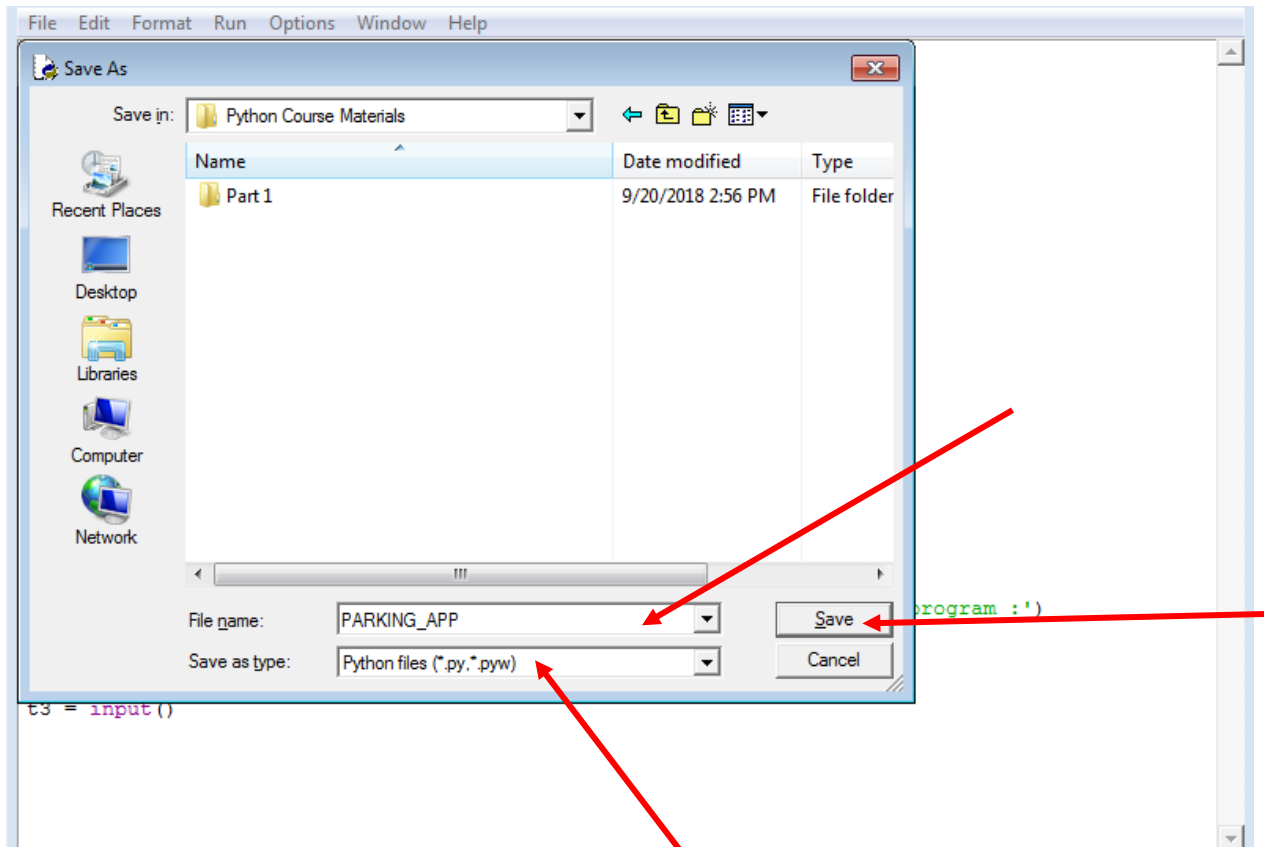


Introduction to Computer Programming in Python
A SunCam online continuing education course

Name your file.

Note that the file type is defaulted to a “Python files (*.py *.pyw)”, which shall be kept.

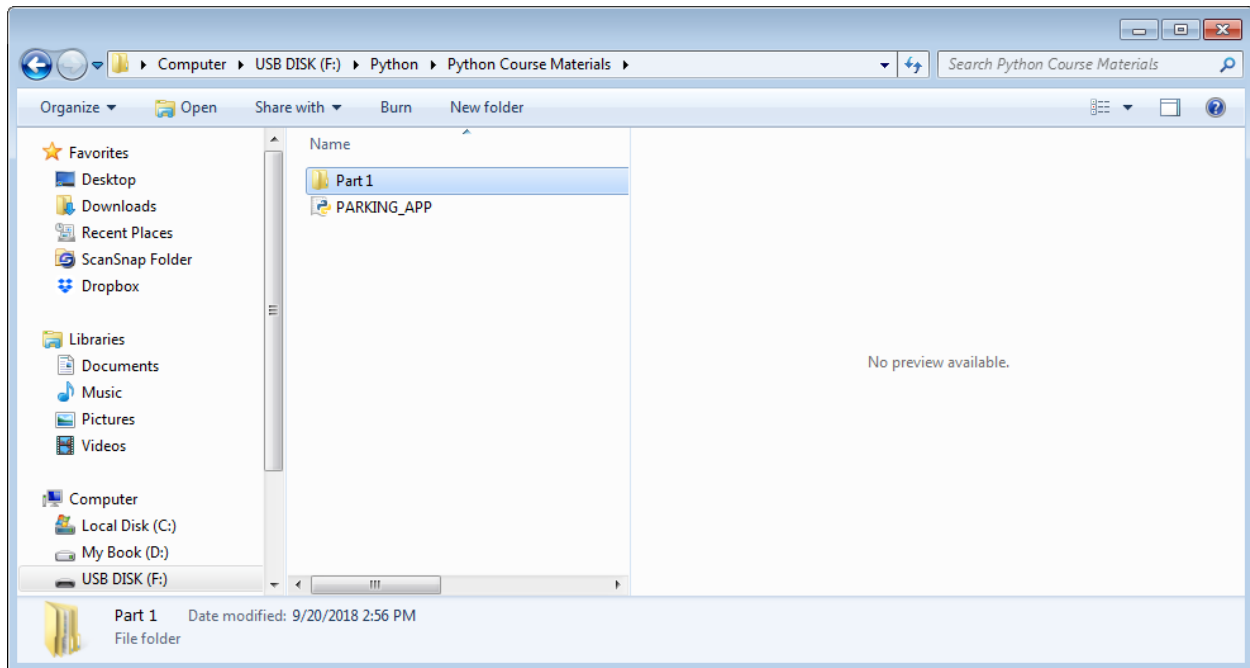
Click on **Save** to complete the process.





Introduction to Computer Programming in Python
A SunCam online continuing education course

Use your File Explorer to navigate to the folder and confirm the file is indeed saved correctly to that folder, otherwise repeat the process accordingly.



We shall resume typing code.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Add code to conduct the parking calculations as follows.

```

PARKING_APP.py - E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PARK...
File Edit Format Run Options Window Help
n2 = input ()

print('What is the average regular parking duration in minutes ? ')
t2 = input ()

print('Enter the number of vehicles expected due to the online pickup program :')
n3 = input ()

print('What is the average online pickup parking duration in minutes ? ')
t3 = input ()

# use a parking efficiency factor of 0.9 which is typical for a parking lot
f = 0.9

# the parking demand
D = n1*T + n2*t2/60 + n3*t3/60 #dividing by 60 converts minutes to hours

# and now compute the number of parking spaces needed
M = D/(f*T)

print('-----')

# and now print the result in a "report"
print('NUMBER OF PARKING SPACES REQUIRED.')
print(round(M))

print('=====')
print(' ')
print('This is the minimum number of parking spaces required!')

print('=====')
print('All Commands Executed !')
Ln: 6 Col: 50

```

Annotations in the image:

- Red arrow pointing to `print(round(M))`: makes sense to round the number of spaces
- Red arrow pointing to `print('-----')`: just to make the report look good

Save your file again.

We are now ready to run the script but before we do so we can check the code for any syntax and other errors.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Run**.

Click on **Check Module**.

```

PARKING_APP.py - E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PARK...
File Edit Format Run Options Window Help
n2 = input()
print('What is parking duration in minutes ? ')
t2 = input()

print('Enter the number of vehicles expected due to the online pickup program :')
n3 = input()

print('What is the average online pickup parking duration in minutes ? ')
t3 = input()

# use a parking efficiency factor of 0.9 which is typical for a parking lot
f = 0.9

# the parking demand
D = n1*T + n2*t2/60 + n3*t3/60 #dividing by 60 converts minutes to hours

# and now compute the number of parking spaces needed
M = D/(f*T)

print('-----')

# and now print the result in a "report"
print('NUMBER OF PARKING SPACES REQUIRED.')
print(round(M))

print('=====')
print(' ')
print('This is the minimum number of parking spaces required!')

print('=====')
print('All Commands Executed !')
|
Ln: 50 Col: 0

```



Introduction to Computer Programming in Python
A SunCam online continuing education course

Any applicable error messages will appear in IDLE.

Check IDLE for errors that need to be addressed before proceeding to run the script.

In this case the coast is clear so we may proceed.

Click on **Run**.

Click on **Run Module**.

```

PARKING_APP.py - E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PARK...
File Edit Format Run Options Window Help
n2 = input ()
print ('What is parking duration in minutes ? ')
t2 = input ()

print ('Enter the number of vehicles expected due to the online pickup program :')
n3 = input ()

print ('What is the average online pickup parking duration in minutes ? ')
t3 = input ()

# use a parking efficiency factor of 0.9 which is typical for a parking lot
f = 0.9

# the parking demand
D = n1*T + n2*t2/60 + n3*t3/60 #dividing by 60 converts minutes to hours

# and now compute the number of parking spaces needed
M = D/(f*T)

print ('-----')

# and now print the result in a "report"
print ('NUMBER OF PARKING SPACES REQUIRED.')
print (round (M))

print ('=====')
print (' ')
print ('This is the minimum number of parking spaces required!')

print ('=====')
print ('All Commands Executed !')
|
Ln: 50 Col: 0

```




Introduction to Computer Programming in Python
A SunCam online continuing education course

Code execution proceeds through IDLE

A screenshot of a Python 3.7.2 Shell window. The window title is "*Python 3.7.2 Shell*" and it has standard Windows window controls (minimize, maximize, close). The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main text area shows the following output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
RESTART: E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PA  
RKING_APP.py  
What are the number of hours of operation of this business ?
```

The status bar at the bottom right indicates "Ln: 6 Col: 2".



Introduction to Computer Programming in Python
A SunCam online continuing education course

Follow the input prompts accordingly to complete execution of the script.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
  RESTART: E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PA
  RKING_APP.py
  What are the number of hours of operation of this business ?
  12
  Enter the number of employees :
  40
  Enter the number of regular parking vehicles expected :
  2500
  What is the average regular parking duration in minutes ?
  25
  Enter the number of vehicles expected due to the online pickup program :
  600
  What is the average online pickup parking duration in minutes ?
  10
  Traceback (most recent call last):
    File "E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PA
    RKING_APP.py", line 33, in <module>
      D = n1*T + n2*t2/60 + n3*t3/60 #dividing by 60 converts minutes to hours
  TypeError: can't multiply sequence by non-int of type 'str'
  >>> |
  
```

Ln: 21 Col: 4

Unfortunately, the program crashes. The error message suggests that the data obtained via the *input()* prompts was stored as string (“str”) data type which cannot compute in a mathematical expression. There is a need to “force” the data to a numerical type before inserting into the calculation formula.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Update the `input()` lines as follows to convert the data entries to the float data. We shall discuss data type conversions in detail later in this course series.

```

*PARKING_APP.py - E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PAR...
File Edit Format Run Options Window Help
# D = f*M*T
# where D is the parking demand in space-hours
# where M is the number of parking spaces required
# where T is the hours of operation of the business
# where f is the parking efficiency factor

print('What are the number of hours of operation of this business ? ')
T = float(input()) # will stop program for user to respond with data entry
                  # user response will stored at variable on the left hand si

print('Enter the number of employees :')
n1 = float(input())

print('Enter the number of regular parking vehicles expected :')
n2 = float(input())

print('What is the average regular parking duration in minutes ? ')
t2 = float(input())

print('Enter the number of vehicles expected due to the online pickup program :')
n3 = float(input())

print('What is the average online pickup parking duration in minutes ? ')
t3 = float(input())

# use a parking efficiency factor of 0.9 which is typical for a parking lot
f = 0.9

# the parking demand
D = n1*T + n2*t2/60 + n3*t3/60 #dividing by 60 converts minutes to hours

# and now compute the number of parking spaces needed
M = D/(f*T)

```

Ln: 12 Col: 21

Save the file.

Rerun the file.



Introduction to Computer Programming in Python
A SunCam online continuing education course

Follow the prompts accordingly to complete execution of the program.

```

Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
25
Enter the number of vehicles expected due to the online pickup program :
600
What is the average online pickup parking duration in minutes ?
10
Traceback (most recent call last):
  File "E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PARKING_APP.py", line 41, in <module>
    D = n1*T + n2*t2/60 + n3*t3/60    #dividing by 60 converts minutes to hours
TypeError: can only concatenate str (not "float") to str
>>>
RESTART: E:\Python\Python Course Materials\Tutorial Files\1.7_File_Execution\PA
RKING_APP.py
What are the number of hours of operation of this business ?
12
Enter the number of employees :
40
Enter the number of regular parking vehicles expected :
2500
What is the average regular parking duration in minutes ?
25
Enter the number of vehicles expected due to the online pickup program :
600
What is the average online pickup parking duration in minutes ?
10
-----
NUMBER OF PARKING SPACES REQUIRED.
150
=====

This is the minimum number of parking spaces required!
=====
All Commands Executed !
>>>
Ln: 79 Col: 4

```

Success!

Take note of where you saved your parking app file.
 Close out of IDLE and the File Editor.



Introduction to Computer Programming in Python A SunCam online continuing education course

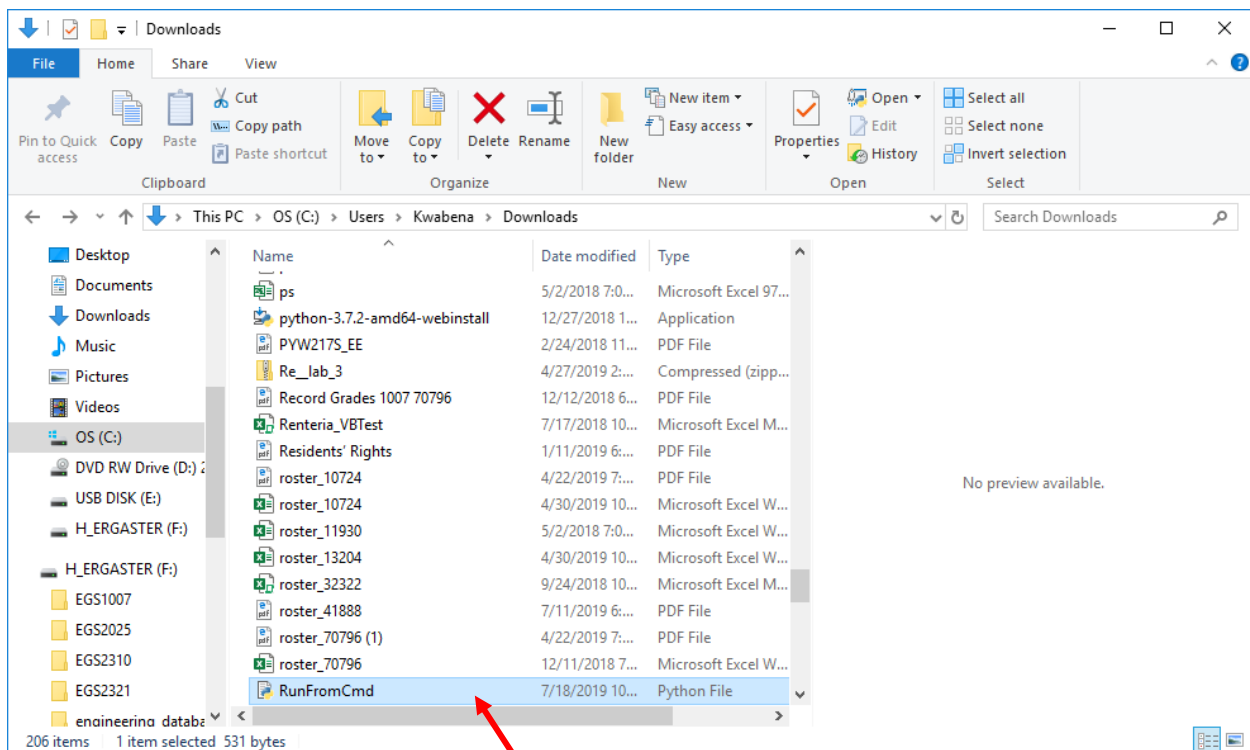
7.4 Windows Command Prompt

Another method of running a Python file is to call the file from the **Windows Command Prompt**. This may a bit weird but bear with us.

Locate the file **RunFromCmd.py** that was supplied with the course materials when you signed up for this course on Suncam.com.

Copy the file and paste it in your Downloads folder.

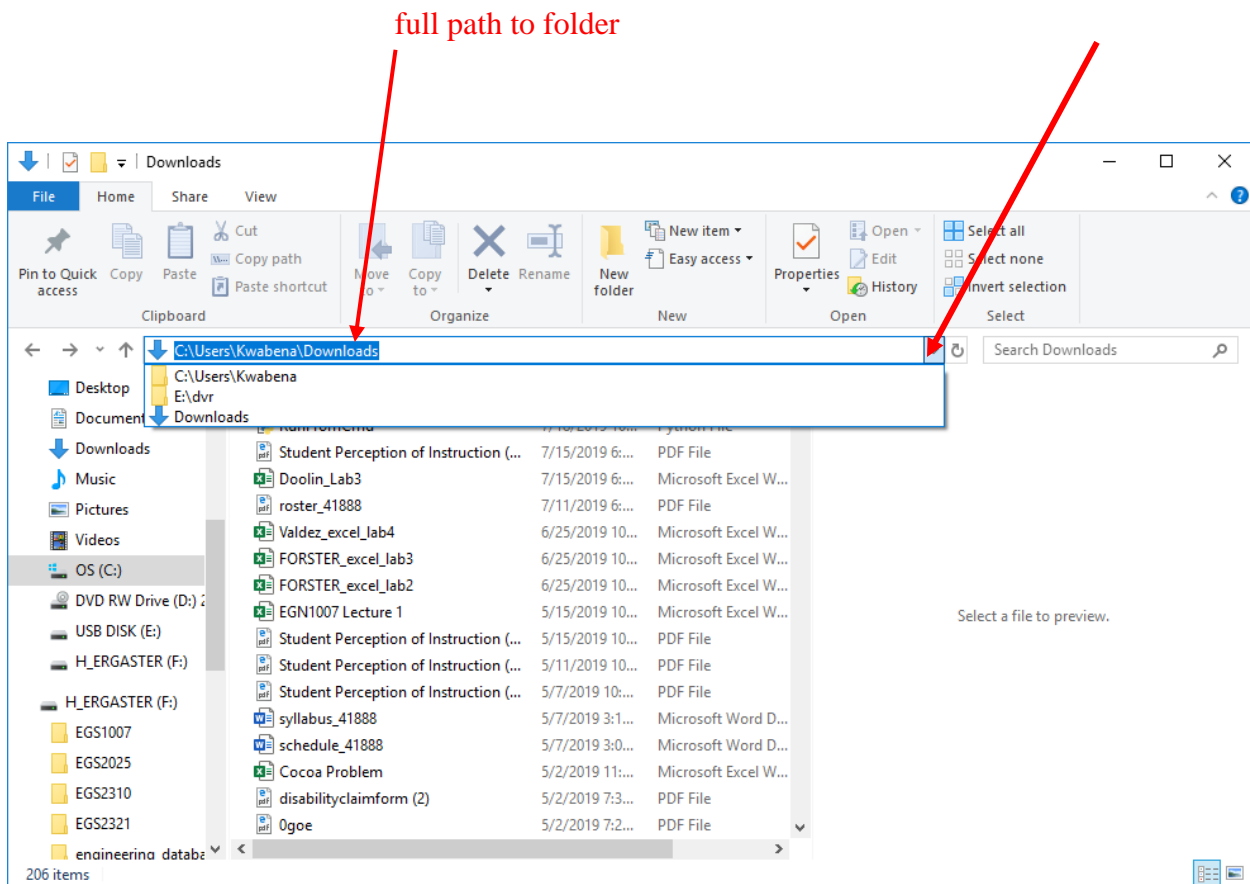
(This exercise will work from any desktop folder, but just for the sake of consistency of results let's all agree to use the Downloads folder.)





Introduction to Computer Programming in Python A SunCam online continuing education course

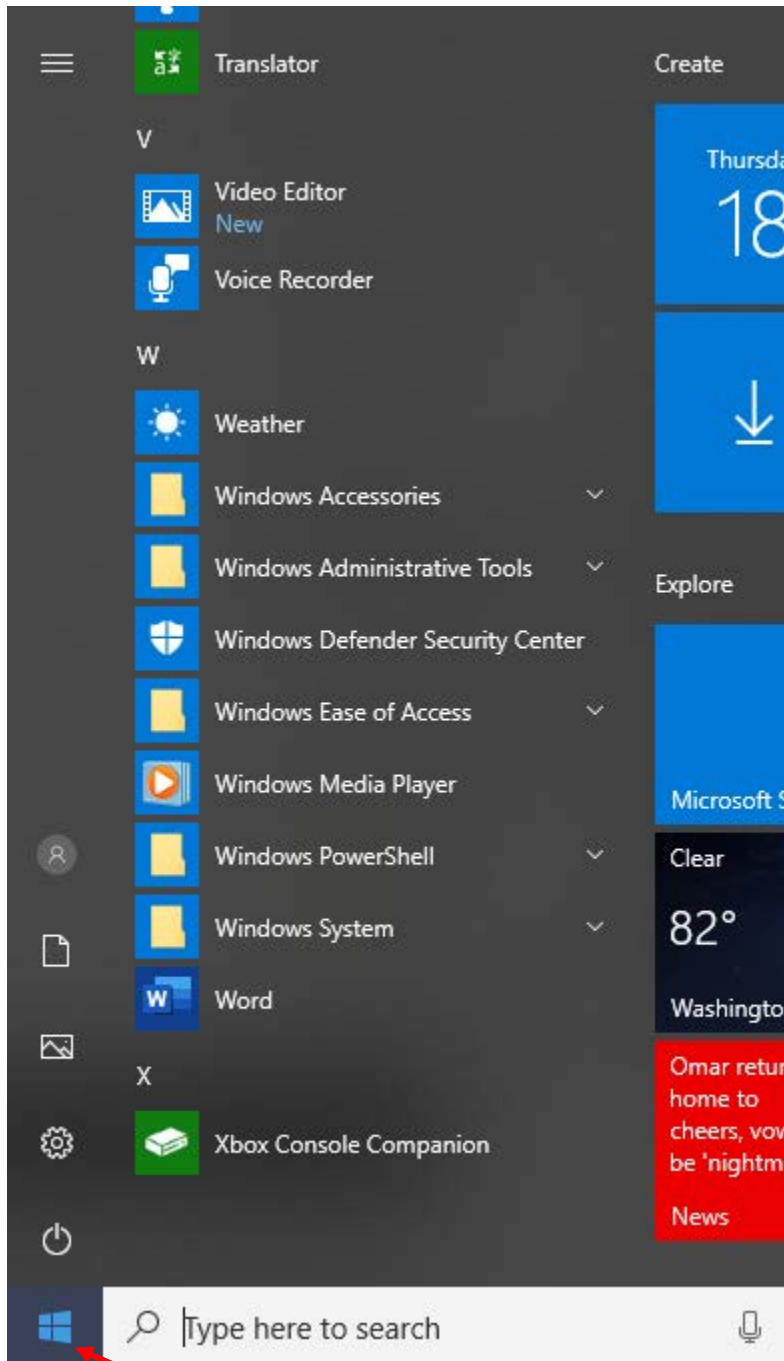
Click on the drop-down arrow of the navigation bar and note the full path to the folder.





Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Windows Start**.

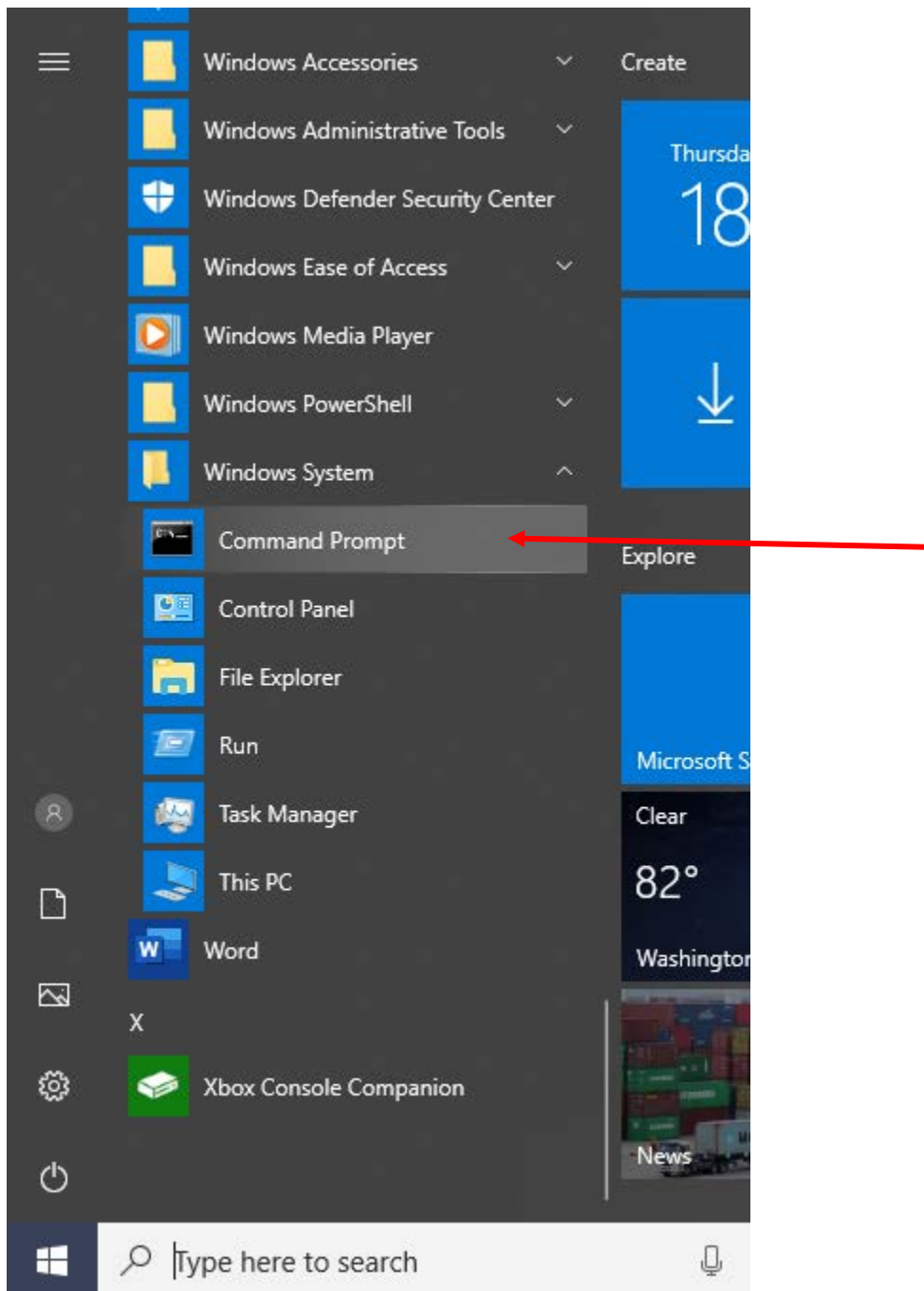




Introduction to Computer Programming in Python
A SunCam online continuing education course

Scroll down to **Windows System**.

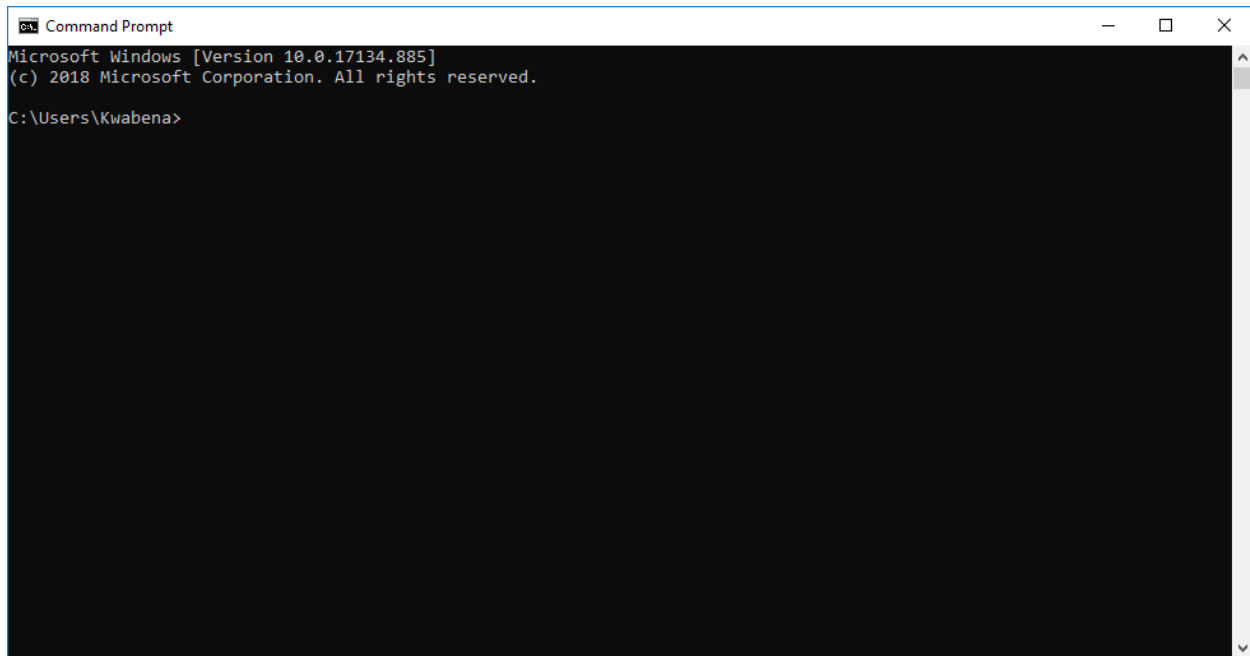
Click on the drop-down and select **Command Prompt**.





Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on the **Command Prompt** to open it.



The **Command Prompt** displays the directory (folder) it is currently pointed to. We shall change the current directory to the Downloads folder where the file **RunFromCmd.py** currently resides.

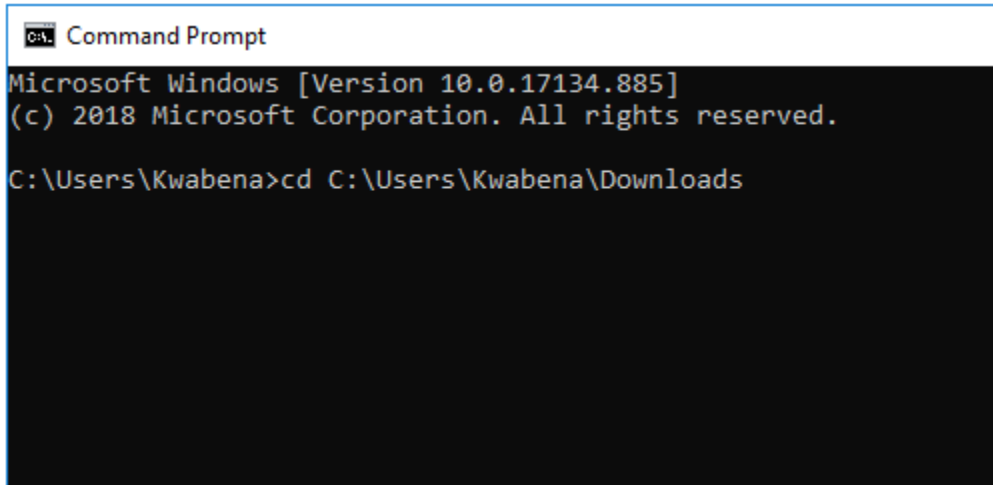
At the **Command Prompt** cursor (the flashing cursor) type the full path to your Downloads folder that you reviewed and noted three steps ago, (On my computer the full path is C:\Users\Kwabena\Downloads. Please type yours), as follows.

cd C:\Users\Kwabena\Downloads



Introduction to Computer Programming in Python
A SunCam online continuing education course

The *cd* is the **Command Prompt** command to change directory.

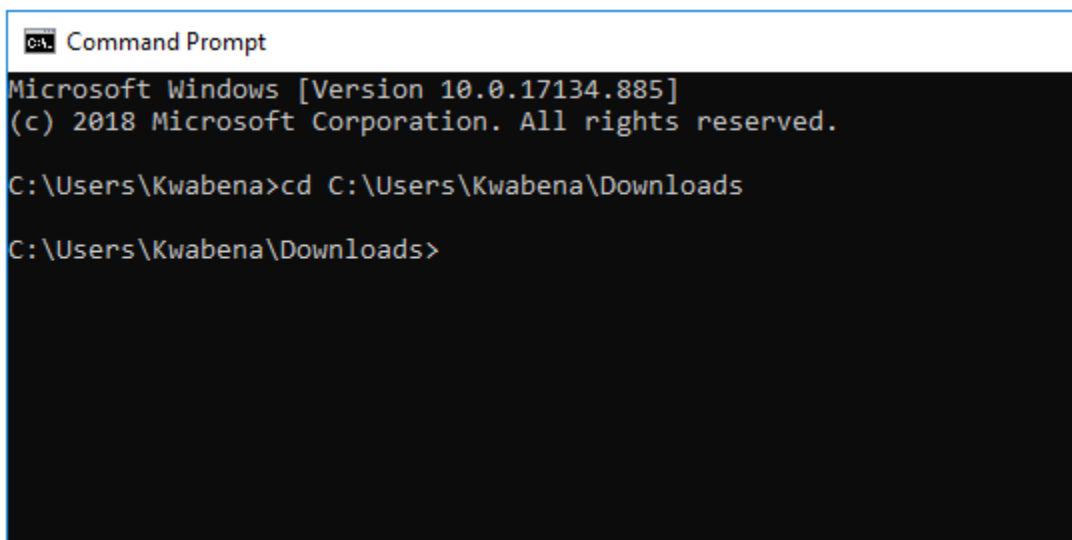
A screenshot of a Windows Command Prompt window. The title bar reads "Command Prompt". The text inside the window shows the Windows version and copyright information, followed by the command to change the directory to the Downloads folder.

```
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Kwabena>cd C:\Users\Kwabena\Downloads
```

Hit the **Enter** key on your keyboard.

The current directory is changed.

A screenshot of a Windows Command Prompt window, similar to the one above. It shows the same initial text and command, but now includes a second line showing the current directory after the command is executed.

```
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Kwabena>cd C:\Users\Kwabena\Downloads

C:\Users\Kwabena\Downloads>
```



Introduction to Computer Programming in Python
A SunCam online continuing education course

At the cursor type the name of the Python file including the Python **.py** suffix, to run it.

```
Command Prompt
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Kwabena>cd C:\Users\Kwabena\Downloads
C:\Users\Kwabena\Downloads>RunFromCmd.py
```

Hit the **Enter** key on your keyboard.
The file executes.

```
Command Prompt - RunFromCmd.py
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Kwabena>cd C:\Users\Kwabena\Downloads
C:\Users\Kwabena\Downloads>RunFromCmd.py
Enter your name :
```



Introduction to Computer Programming in Python
A SunCam online continuing education course

Follow the prompts to complete the program execution.

```

Command Prompt
Microsoft Windows [Version 10.0.17134.885]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Kwabena>cd C:\Users\Kwabena\Downloads

C:\Users\Kwabena\Downloads>RunFromCmd.py
Enter your name :

Josephine
Hi Josephine. Welcome to Python Programming for Engineers.

You have successfully completed this task.
Thanks and see you soon.

-----
=====

C:\Users\Kwabena\Downloads>_

```

Close **Command Prompt**.

The exercise is complete!

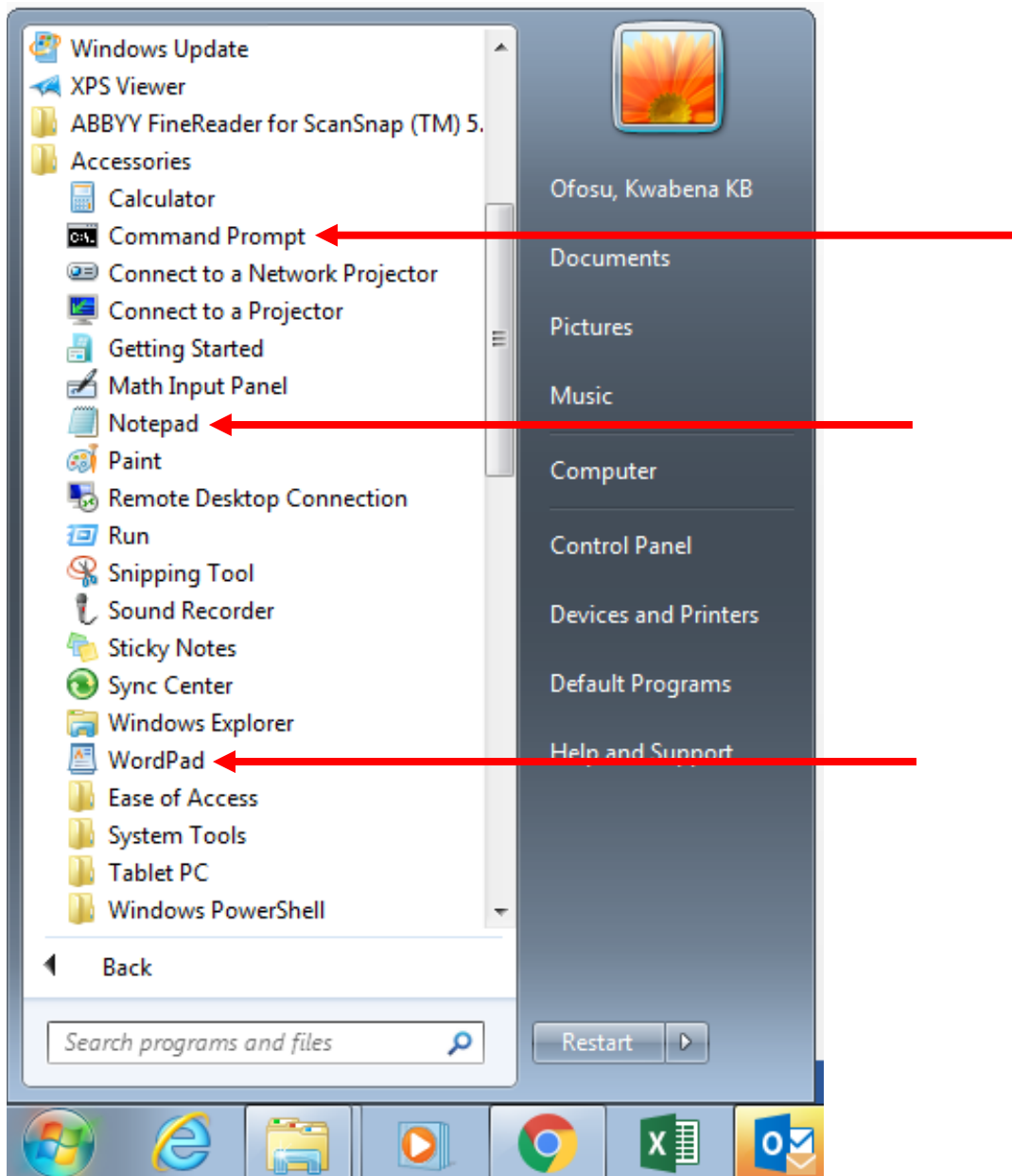
So in fact this entire course series and any *Python* work in general, can be completed through the **Command Prompt** once a Python interpreter has been installed on your computer. IDLE (Python GUI) is one of several products put out there to help *Python* programmers. There are many other such products. A google search will pull up many options to consider.

A Python executable file can be created using any text editor program, for example, *Notepad*, *WordPad* etc., and saving the file with the Python **.py** suffix, and then running the file in the **Command Prompt** (or your preferred Python interpreter program for that matter).



Introduction to Computer Programming in Python
A SunCam online continuing education course

Remember that *Notepad* and *WordPad* for example come free with your Microsoft Windows operating system.



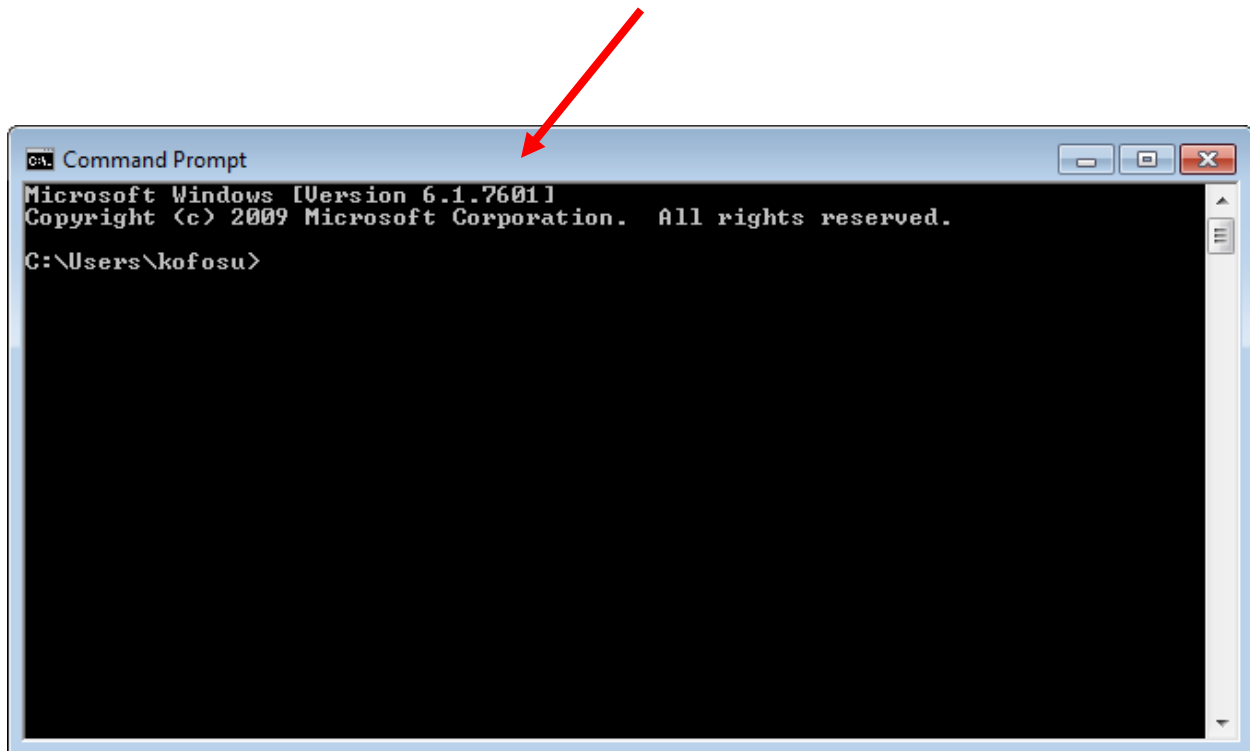


Introduction to Computer Programming in Python
A SunCam online continuing education course

Reopen the **Command Prompt**.

If desired, the **Command Prompt** color scheme – background color, font etc. can be changed to the user's preferences as follows.

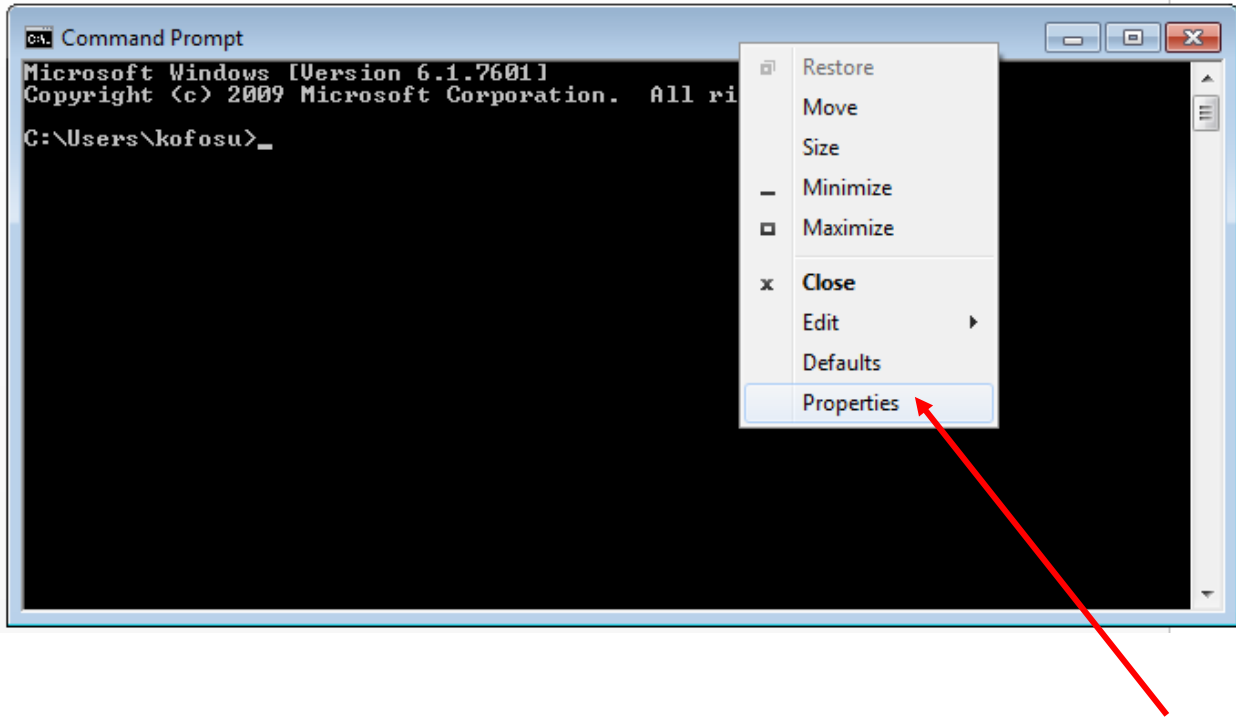
Right click on the **Command Prompt** title bar.





Introduction to Computer Programming in Python
A SunCam online continuing education course

Click on **Properties**.





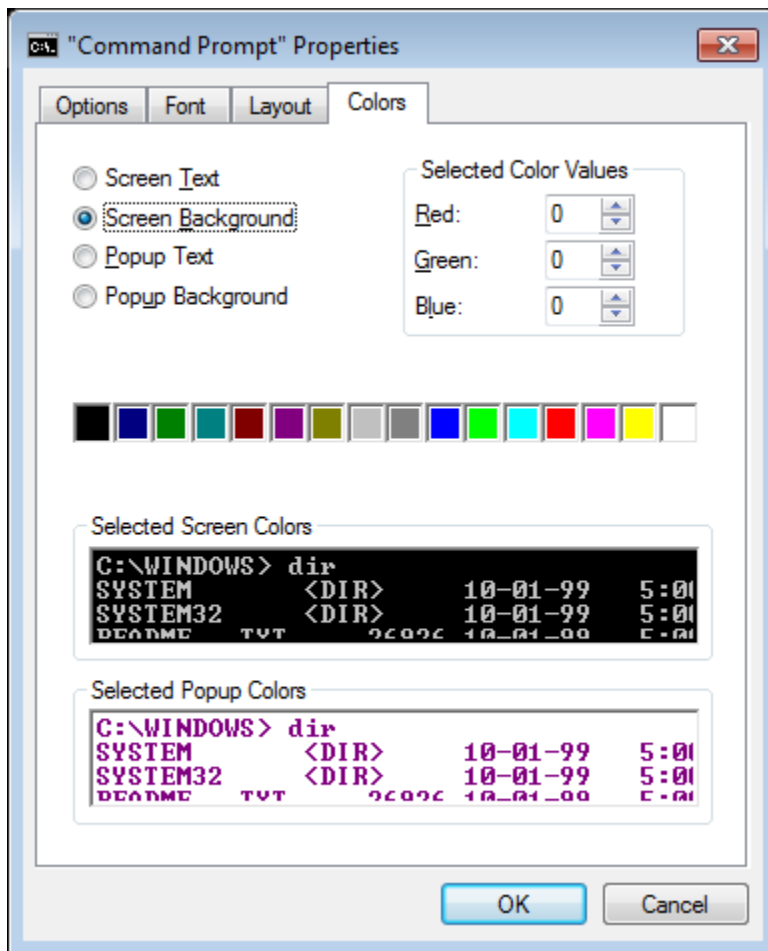
Introduction to Computer Programming in Python
A SunCam online continuing education course

The **Command Prompt** Properties window opens.

Click on the tabs to review and change the properties to your preference.

You may have to try a few colors back and forth until you find what you like best.

If you prefer the original black and white color scheme, you do not have to change anything.



Click on **OK** to dismiss the **Command Prompt** Properties window.
 Close the **Command Prompt**.

Successful Completion !



Introduction to Computer Programming in Python
A SunCam online continuing education course

8. CONCLUSION

This course has presented a broad overview of fundamental concepts and principles of computer programming and presented them in situations encountered by practicing engineers and scientists. All codes were developed using the *Python* programming language.

This course, the first of a series on *Python* programming, covered an introduction to computers and computer programming languages. This was followed by an introduction to the IDLE (Python GUI) application for developing and testing *Python* scripts. The concepts of expressions, data types and variables and how they are applied in *Python* were presented. This was followed by presentations of the fundamental concepts of strings, lists and tuples, and dictionaries and sets. Real-life examples relevant to an engineering practitioner were used to demonstrate the application of the concepts and skills presented in this course.

This course has enabled participants to identify situations where programming is useful and advantageous to the professional. Practitioners are strongly encouraged to look out for situations in their domains of expertise where programming solutions are applicable and beneficial to their work and their organization.

Computer programming requires a careful and meticulous approach and can only be mastered and retained by practice and repetition.

Good Luck and Happy Programming.



Introduction to Computer Programming in Python
A SunCam online continuing education course

REFERENCES

Python Software Foundation. (2019). *Python Software Foundation*. Retrieved May 2019, from Python Software Foundation: <https://www.python.org/psf/>

Real Python. (2019). *Python Tutorials - Real Python*. Retrieved March 2019, from Real Python: <https://realpython.com/>

w3schools. (2019). *Python Tutorial*. Retrieved April 2019, from w3schools.com: <https://www.w3schools.com/python/default.asp>

Images were all drawn/ prepared by Kwabena Ofosu