1

INTRODUCTION TO PYTHON

LEARNING OBJECTIVES

- Explain Python's background and important features
- Describe free, open-source software (FOSS)
- Summarize Python's user community and available resources
- Install Python's platform-independent interpreter
- Execute Python code in an Interactive Development Environment (IDE)
- Describe the two data sets used throughout the book

INTRODUCTION

This chapter gives a brief background of Python and then goes on to illustrate Python programming using an Interactive Development Environment (IDE). **Python** is an interpreted computer programming language in which you can enter code instructions one at a time or as part of a larger program, which comprises many instructions. Throughout this book, illustrations of entering and executing Python code provide hands-on experience and familiarity with programming in Python. The Python code examples begin in this chapter with writing and running a sample instruction of Python code that prints a simple message to the screen. At the end of the chapter, we introduce the two real-world, large-scale data sets that we will use throughout the book. These data sets embody many different types of data and are well suited for the data analysis and visualization covered in later chapters.

BRIEF INTRODUCTION TO PYTHON AND PROGRAMMING

Guido van Rossum first conceived the Python programming language in December 1989 with the idea that it should be easy to read and that it should let users create their own packages of special-purpose coding modules that others could use (Anonymous, 2018). Python's first release was in 1991, and it combines simple syntax, abundant online resources, and a rich ecosystem of scientifically focused toolkits with a heavy emphasis on community (Perkel, 2015). **Syntax** is a set of rules that dictate how to specify instructions in a programming language. **Packages** are libraries of code modules that other programming code can access and use. As of January 2020, there are more than 212,000 projects with packages available for download from the Python Package Index (PyPI), a repository of packages for the Python programming language (Python Software Foundation, 2020). In addition, Python was the most popular introductory language at American universities in 2014, but the teaching of it is generally limited to those studying science, technology, engineering, and mathematics (Anonymous, 2018). The intended audience for this textbook is students and researchers in business and the social sciences. There is prolific use of Python today in both business and the social sciences to develop applications for data analytics. In addition to statistical analysis, we can use Python for web scraping, text mining, machine learning, and developing applications with graphical user interfaces (all of which we cover in this book). Although we can accomplish each of these individually with other programming languages (such as R) and software packages (such as SAS and Tableau), learning Python enables you do all these things and much more.

Python's Use in Education, Research, and the Corporate World

The development of the Internet has both made large amounts of information available to users, as well as enabled users to create large amounts of information and make it available to the rest of the world. Data are manipulated and processed using computer programming by both business and the social sciences to gain insights that would be too difficult to obtain otherwise. The Python programming language has been the most popular introductory programming language taught at American universities for good reasons. The explosive growth of "big data" in disciplines such as bioinformatics, neuroscience, and astronomy has made programming know-how ever more crucial, in that researchers who can write code in Python can manage their data sets and work more efficiently on research-related tasks, including analyzing and visualizing data (Perkel, 2015). The corporate world also recognizes the importance of analyzing data to gain insights about current and potential customers. In addition, corporations are developing business applications using Python for numerous purposes, including developing strategic information systems, including enterprise resource planning (ERP), customer relationship management (CRM), and ecommerce applications (Smets, 2019).

Several advantages that Python has compared to commercial packages such as SPSS and SAS are that Python is open source and can run on many platforms. Users are free to make copies, distribute, and even change the software. Python is perfect for teaching statistics in a data-rich environment and has simplified debugging for the programmer using its built-in debugging feature (Ozgur, Colliau, Rogers, Hughes, & Myer-Tyson, 2017). Like Python, the R programming language is also an open-source programming language used for data analytics. There is prolific use of both R and Python in the business world and for academic and research purposes. We focus on Python because we see a need for a text that presents Python programming specifically for those in the fields of social sciences and business to develop applications for data analytics. Whereas some may prefer R for statistical analysis and plotting charts, Python is a general-purpose scripting language used to develop applications with graphical user interfaces (GUIs) and may be favored when working with text-based data.

Python Is Free, Open-Source Software (FOSS)

Perhaps the most important reason for the rapid growth in the usage of Python in business and the social sciences is that Python is **free open-source software** (**FOSS**). FOSS is an inclusive term that covers both free software and open-source software (Marsan, Pare, & Beaudry, 2012). The definition of **free software** is that the users have the freedom to run, copy, distribute, study, change, and improve the software (Free Software Foundation, 2019). **Open-source software** requires that the license to use the software shall not restrict any party from selling or giving away the software as a component of a larger software distribution (Open Source Initiative, 2007). As a result, organizations not only are free to use and change Python but also can create and sell commercial applications using Python.

Being FOSS is a true advantage that Python has over other commercially available packages, as it is continually improved. Software development peers iteratively develop, incrementally release, review, and refine FOSS projects in an ongoing agile manner (Scacchi, 2004b, referenced in Goth, 2007). FOSS communities develop software that is extremely valuable, generally reliable, globally distributed, made available for acquisition at little or no cost, and readily used in its associated community (Scacchi, 2004a).

User Community and Python Resources

You can find many Python resources at the Python website, **https://www.python.org**. You can download the latest version of Python from **https://www.python.org/downloads/** (Version 3.8.0 as of October 14, 2019). Python is **platform independent,** software that can run on most if not all the latest operating systems/computing platforms. A **platform** is the combination of a physical device and an operating system. You can run the latest version of Python on Windows, Linux/Unix, Mac OS X, and other operating systems. You can find documentation for the latest version of Python (as well as for older versions) at **https://docs.python.org/dev/**. Table 1.1 lists the most recent versions of Python documentation that were available on the Python website as of November 22, 2019. Previous versions of documentation remain available online as well. The Python Package Index is a repository of software for the Python programming language located at **https://pypi.org/**.

TABLE 1.1 ● STATUS OF DOCUMENTATION FOR MOST RECENT PYTHON VERSIONS AS OF NOVEMBER 22, 2019

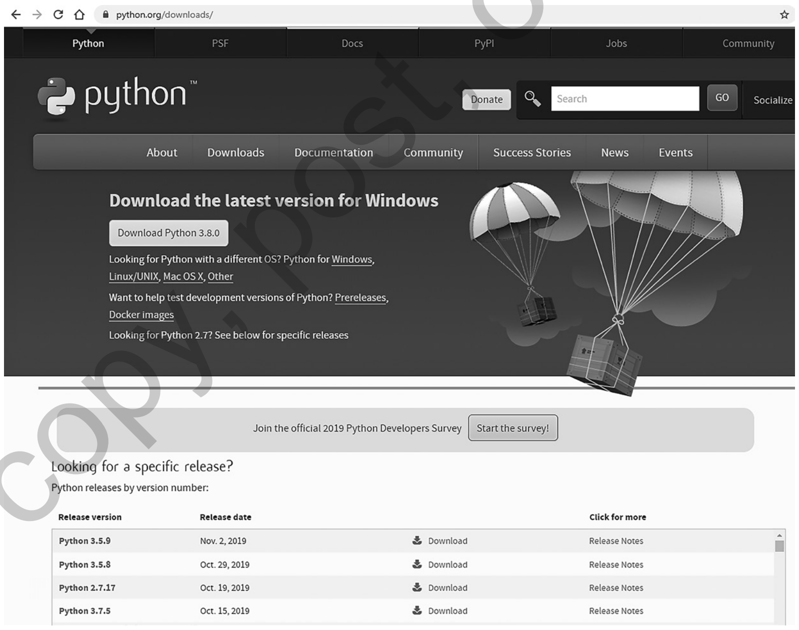
Python Version	Status
Python 3.9	In development
Python 3.8	Stable
Python 3.7	Stable
Python 3.6	Security fixes
Python 2.7	Stable

Lessons learned: In this section, we learned that Python is free and open-source software (FOSS) and that there are now more than 212,000 projects with packages written in Python that are available to use and modify in the Python Package Index. The goal of this book is to teach Python programming to those in the fields of social sciences and business to develop applications using Python packages for data analytics.

SETTING UP A PYTHON DEVELOPMENT ENVIRONMENT

One way to set up a Python development environment on a computing device is to connect to the Python download webpage (**https://www.python.org/downloads/**), as shown in Figure 1.1. Once on that webpage, if you are running Windows, simply click on the Download Python button (for the latest version). If you are not running Windows, select the link that corresponds to the operating system you are using (found immediately below that download button) and follow the instructions found on the corresponding webpage. Note that we will be using the Windows operating system for illustrating Python throughout the book, so if you are using Mac OS or another operating system, you will have some variations in the appearance and detailed workings of Python.

FIGURE 1.1 ● PYTHON DOWNLOAD WEBPAGE



Download the latest version for Windows

Download Python 3.8.0

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Prereleases, Docker images

Looking for Python 2.7? See below for specific releases

Join the official 2019 Python Developers Survey Start the survey!

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.5.9	Nov. 2, 2019	Download	Release Notes
Python 3.5.8	Oct. 29, 2019	Download	Release Notes
Python 2.7.17	Oct. 19, 2019	Download	Release Notes
Python 3.7.5	Oct. 15, 2019	Download	Release Notes

The lower half of the Python download webpage shown in Figure 1.1 lists the release dates for different release versions.

Python Versions

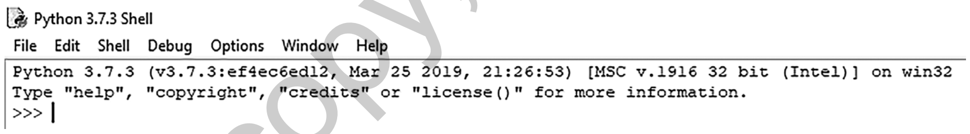
Python versions are numbered A.B.C., where A is the major version, B is a minor version number (for incremental changes), and C is a micro-level number (for bug fixes; Python Software Foundation, 2019, "General Python FAQ"). Release Version 3.x has significant changes from release Version 2.x, and code written in each version is not compatible with the other version. Figure 1.1 shows that the release date for release Version 2.7.17 is October 19, 2019. There are still new updates for Python Version 2.x for those who have developed Python code in the past and do not want to make the changes necessary for that code to be compatible with Version 3.x. Python Version 3.x is the recommended version as Version 2.x, although still widely used, will no longer be maintained after January 1, 2020 (Python Software Foundation, 2019, "General Python FAQ"). This book uses Python Version 3.7, and we have tested all Python code in the book using that specific version. If you use either an older or a newer version of Python, there may be some issues with code execution.

Lessons learned: In this section, we learned how to install Python on our computer and that there are different versions of Python. We also learned that using different operating systems and different versions of Python can affect how we write and execute Python code.

EXECUTING PYTHON CODE IN THE IDLE SHELL WINDOW

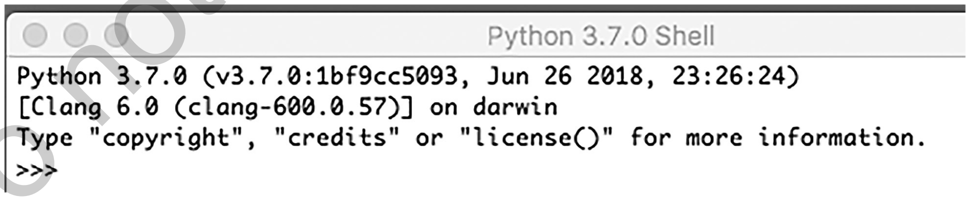
After downloading and installing the Python development environment, you can verify that it is properly functioning. The standard distribution of Python comes with an Interactive Development Environment (IDE) named IDLE (for Integrated Development and Learning Environment). An **Interactive Development Environment (IDE)** contains facilities for writing and editing code as well as testing and debugging code. IDLE runs on Windows, Mac OS X, and UNIX. Documentation for IDLE can be found at **https://docs.python.org/3/library/idle.html**. The IDLE Python shell window is an interactive interpreter that can execute lines of Python code one at a time. Figure 1.2 is the IDLE Python shell console on Windows, and Figure 1.3 is the IDLE Python shell console on a Mac. Although mostly similar, there are platform-specific variations. For consistency, we will be using Windows-based Python illustrations throughout the remainder of this textbook.

FIGURE 1.2 ● THE IDLE PYTHON SHELL WINDOW ON WINDOWS



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> |
```

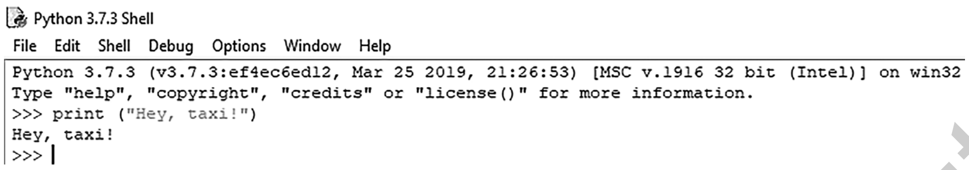
FIGURE 1.3 ● THE IDLE PYTHON SHELL WINDOW ON A MAC



```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 26 2018, 23:26:24)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>>
```

The simplest way to execute Python statements is to type in a Python statement at the shell window command prompt; after the enter key is pressed, the Python interpreter executes the statement. Figure 1.4 illustrates how this works by typing the command `print("Hey, Taxi!")` and pressing enter. Interacting directly with the Python interpreter in this way can be very useful for learning about different Python commands and features. Figure 1.4 also shows four other commands: `help`, `copyright`, `credits`, and `license`.

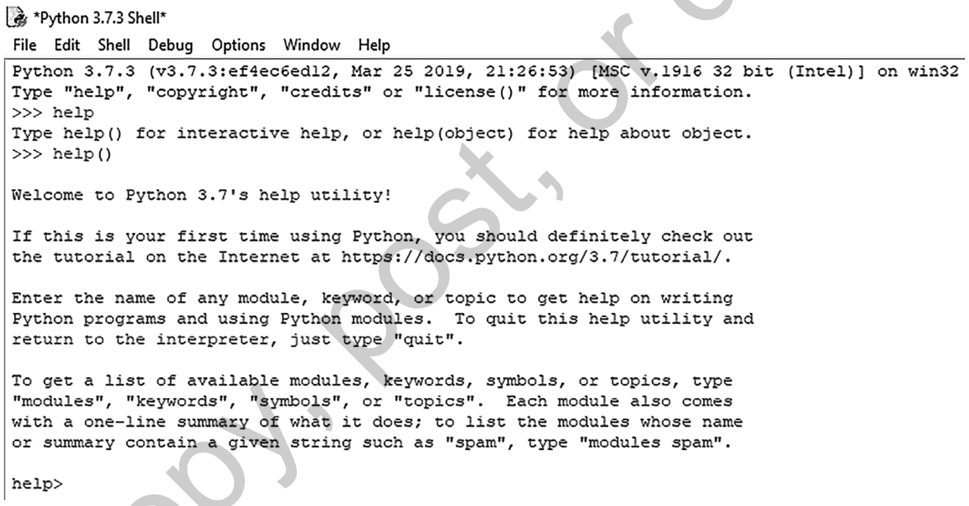
FIGURE 1.4 ⬢ EXECUTING PYTHON STATEMENT AT IDLE SHELL WINDOW COMMAND PROMPT



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print ("Hey, taxi!")
Hey, taxi!
>>> |
```

Typing *help* at the command line and pressing enter prompts the user to either enter *help()* for interactive help or else *help(object)* for help about a specific object, as shown in Figure 1.5. Figure 1.5 also shows the response for the interactive help, which directs first-time users to a web-based tutorial for the Python version in use, as well as detailed directions on how to use the interactive help utility.

FIGURE 1.5 ⬢ PYTHON HELP COMMAND AT IDLE SHELL WINDOW COMMAND PROMPT



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> help
Type help() for interactive help, or help(object) for help about object.
>>> help()

Welcome to Python 3.7's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.7/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help>
```

Lessons learned: In this section, we learned how to write and execute Python code in the IDLE shell window.

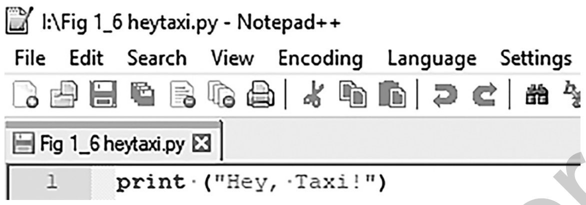
PYTHON INSIGHT

Python is **case sensitive,** which means it interprets uppercase letters (capitals) as different from lowercase letters. As a result, the code statement `Print ("Hey, Taxi!")` will result in an error stating that the name "Print" is not defined. This case sensitivity applies to Python reserved words as well as variables and function names, which we will discuss and illustrate in the next chapter. This can sometimes be annoying for beginning Python programmers (but you will quickly adapt), but it greatly reduces the processing of the Python interpreter that would otherwise need to treat all the possible combinations of upper- and lowercase letters the same.

EXECUTING PYTHON CODE IN FILES

Entering in a few lines of Python code interactively is a great way to learn, but programmers typically store their Python code in text files with the file name extension ".py." We discuss text files in Chapter 5. The convention is to use this file name extension as it identifies the purpose of the file. Figure 1.6 illustrates a text file that we created in a text editor (Notepad++, a free source code editor that can be downloaded from the website **https://notepad-plus-plus.org/**). We use Notepad++ throughout this book to illustrate Python code with line numbers, so that we can refer to specific lines of code by line number. However, we have written and executed all the code examples in the IDLE IDE.

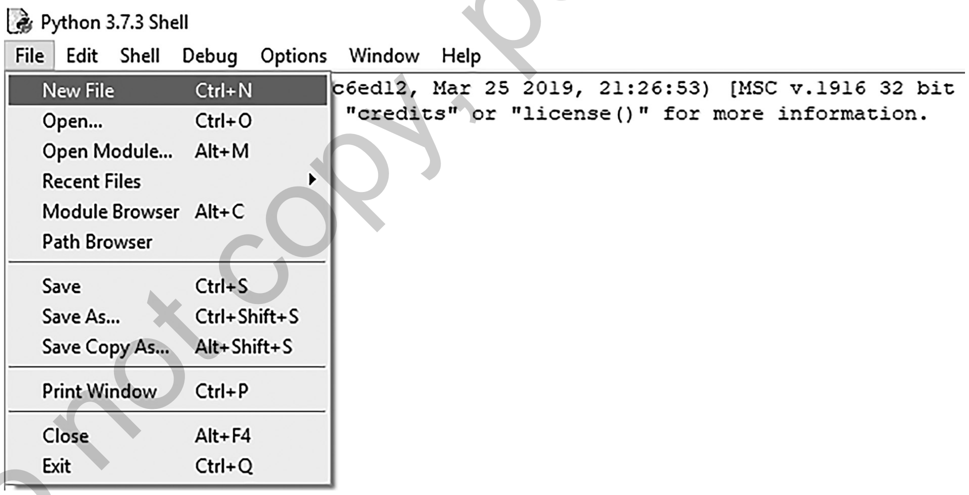
FIGURE 1.6 ⬢ PYTHON CODE IN A TEXT FILE


I:\Fig 1_6 heytaxi.py - Notepad++
File Edit Search View Encoding Language Settings
Fig 1_6 heytaxi.py

```
print·("Hey,·Taxi!")
```

IDLE has a menu system, which makes interactions more user-friendly. Being able to specify which action to perform by selecting an option from a menu reduces the possibilities of typing errors and is much less technically demanding. The File menu in the Python IDLE Shell window shown in Figure 1.7 has facilities for creating new files, opening files and modules, saving files, and printing.

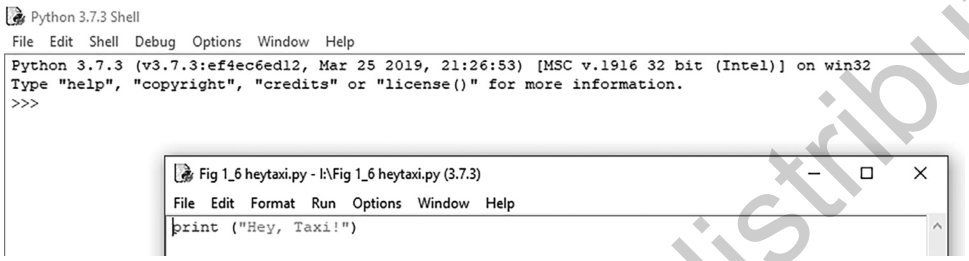
FIGURE 1.7 ⬢ IDLE FILE MENU



```
c6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit
 "credits" or "license()" for more information.
```

Figure 1.8 illustrates that the Python file from Figure 1.6 is open. The IDLE Python Shell is in the background, and the IDLE text editor is in the foreground with the title bar showing the name and location of the opened file. You can open multiple files in the IDLE editor and use colors to differentiate parts of Python code. In Figure 1.8, the word "`print`," which is a Python built-in function, appears in a darker color than the text "`Hey, Taxi!`" which is a string. In the program, the former appears highlighted in purple; the latter, in green. These colors conform to what is known as the “IDLE classic” color scheme, but the colors used can be customized by selecting from the menu “Options” and then “Configure IDLE,” which displays the Settings dialog box. To customize these colors, in the Settings dialog box, choose the Highlights tab to select different types of objects and change their highlight color. Because of limited color used in the print version of this textbook, colors are not displayed as they actually appear.

FIGURE 1.8 ● IDLE EDITOR MENU WITH PYTHON FILE

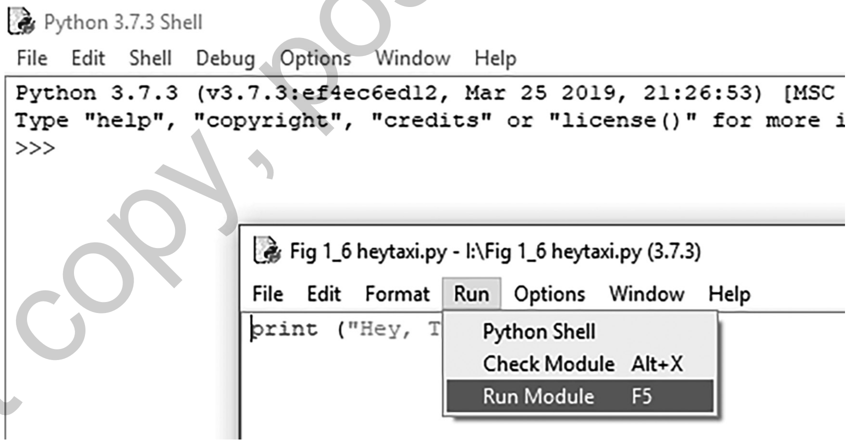

Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Fig 1_6 heytaxi.py - I:\Fig 1_6 heytaxi.py (3.7.3)
File Edit Format Run Options Window Help

```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
```

```
print ("Hey, Taxi!")
```

In addition to being able to edit multiple Python files in different windows, the IDLE menu has an option to run a module (which corresponds to the F5 shortcut key), as depicted in Figure 1.9. A **module** is a text file that contains Python code. Figure 1.10 illustrates the result of clicking on “Run Module” when the “Fig 1_6 heytaxi.py” file window is active. The output of the execution of the Python code in that file is in the IDLE Python Shell (much as we saw the result appear after the execution of the code instruction in the file from the shell command prompt previously in Figure 1.4).

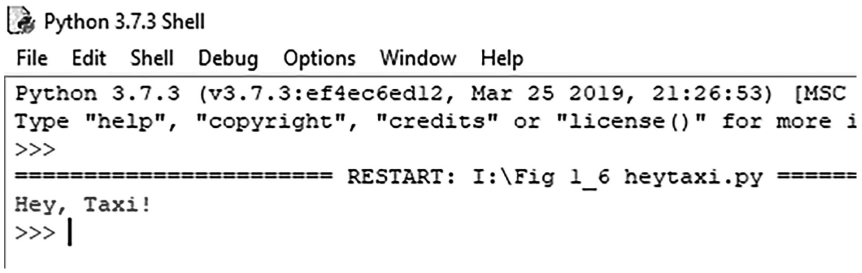
FIGURE 1.9 ● IDLE RUN MENU


Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Fig 1_6 heytaxi.py - I:\Fig 1_6 heytaxi.py (3.7.3)
File Edit Format Run Options Window Help
Python Shell
Check Module Alt+X
Run Module F5

```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC
Type "help", "copyright", "credits" or "license()" for more i
>>>
```

```
print ("Hey, T
```

FIGURE 1.10 ● RESULT OF CODE EXECUTION IN IDLE PYTHON SHELL



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC
Type "help", "copyright", "credits" or "license()" for more i
>>>
======================= RESTART: I:\Fig 1_6 heytaxi.py ======
Hey, Taxi!
>>>
```

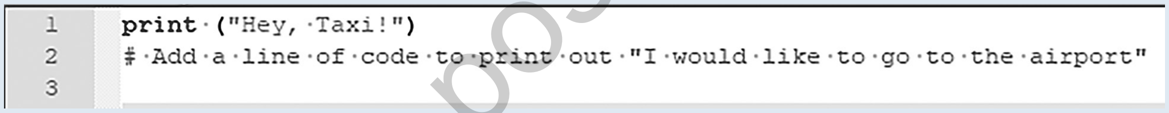
The example just given, although simplistic in nature, illustrates the ability to execute Python code from plain text files. Using files enables the storing and execution of many lines of code as a program. Other features of the IDLE editor include providing syntax and autocompletion for Python statements as well as debugging features to set breakpoints and step through code. We demonstrate debugging in Appendix B.

STOP, CODE, AND UNDERSTAND!

SCU 1.1 Introducing Stop, Code, and Understand! Exercises

We incorporate coding exercises throughout the book to make sure you understand the topics that we are covering. Each one uses comments (which begin with the # sign) to indicate precisely where you need to add or change Python code. To complete each exercise, download the corresponding Python file from the companion website, open the file in the Python IDLE editor, make the requested change, and execute the code to verify that you understand the concept. We explain the solutions to the Stop, Code, and Understand! exercises in Appendix D at the end of the book, and the solution files for each exercise are available on the companion website.

For the first exercise, download the file "SCU 1_1.py" from the companion website and save it either on your computer or on a removable storage device. Next, open the Python IDLE shell and then open the file "SCU 1_1.py" in the Python IDLE editor by clicking on File/Open from the menu and selecting the file you just saved. Run the code by clicking on Run/Run Module from the menu. Next, add a line of Python code where indicated in the following program to print out the message "I would like to go to the airport." Run the revised code by clicking on Run/Run Module from the menu to verify that your added code works correctly.



```
print ("Hey, Taxi!")
# Add a line of code to print out "I would like to go to the airport"

```

Lessons learned: In this section, we learned how to write and execute Python code that we save in plain text files. Using files enables the storing and execution of many lines of code as a program that we can save and run later.

PACKAGE MANAGERS

A **package manager** is a program to install libraries of code. These libraries, or packages, contain previously developed code. Once installed, the code found in the package is available to other Python code, saving a great deal of time and effort. Using a package not only prevents "reinventing the wheel" but also usually benefits from the prior development and testing by an entire community of developers. Python comes with a package manager named **pip** already installed (in Versions 3.4 and later). The Python Software Foundation, a nonprofit company, maintains documentation for pip, which is online at the website **https://pip.pypa.io/en/stable/**. We will be using pip to install several packages throughout this textbook.

Another way to set up a Python development environment is to install a Python distribution, such as the Anaconda distribution, found at the following URL: **https://www.anaconda.com/download/**. **Python distributions** are alternative bundles and are modified packages that include additional functionality. Alternative bundles may not include the latest versions of Python or other libraries and are not maintained by the core Python team (Python Software Foundation, 2019, "Alternative bundles"). We use the pip package manager to install individual packages in this textbook. Learning to install individual packages is an important skill for people programming in Python, which enables the use of packages developed for use within organizations and for packages that are not in any Python distributions.

Lessons learned: In this section, we learned how to use package managers for convenient organization and management of libraries of code. Learning to install individual packages is an important skill to take advantage of the Python packages for both business and social sciences purposes that are available to the Python programming community.

DATA SETS USED THROUGHOUT THE BOOK

We use two data sets throughout this book to illustrate numerous issues faced when working with data. The data sets are the City of Chicago's Taxi Trips data set and data from the General Social Survey. We begin with some simple examples in the next chapter to become acquainted with the nature of the data in these data sets, and in later chapters, we work directly with files containing the data sets as well as retrieve the data directly from the World Wide Web. These data sets provide the basis for our coverage of topics later in the book, including statistical analysis, data visualization, and machine learning.

Taxi Trips Data Set

The Chicago Taxi Trips data set has over 100 million records with 26 fields (variables) per record. Table 1.2 presents a subset of fields and their meaning as described in the Taxi Trips documentation (Levy, 2017). We will later see that the formatting of the data in the taxi trip data set is going to present some challenges when working with the data in Python. On a positive note, these challenges working with real data provide a means of learning practical insights into programming with Python. Table 1.3 has sample data selected from the taxi trips data set that correspond to the fields in Table 1.2.

TABLE 1.2 ● SUBSET OF FIELDS FROM TAXI DATA

Field Name	Description
Trip_ID	A unique identifier for the trip
Trip_Seconds	Time of the trip in seconds
Trip_Miles	Distance of the trip in miles
Fare	The fare for the trip
Tips	The tip for the trip
Payment_Type	Type of payment for the trip
Company	The taxi company

TABLE 1.3 ⬢ SAMPLE TAXI TRIPS DATA

Trip_ID	Trip_ Seconds	Trip_ Miles	Fare	Tips	Payment_ Type	Company
da7a62fce064af7ed48ededf96b51 2248c3691b6	180	0	$4.75	$0.00	Cash	Blue Ribbon Taxi Association Inc.
da7a62231f7df3010e54672f0f3ddf 429fff991b	360	1.1	$6.25	$0.00	Cash	Taxi Affiliation Services
da7a60ec64d8e9a5a2177c7791f5 7970860def88	300	0.5	$5.25	$2.00	Credit card	Taxi Affiliation Services
da7a6432fe64595b2cb23a338c00 8a6bc066d846	240	0.8	$5.25	$0.00	Cash	Taxi Affiliation Services
da7a6256ba4814596c8655eee781 b80eac434cd8	780	2.5	$10.50	$0.00	Cash	Northwest Management LLC
da7a61d7910733ae077ab82756ec 04ff5868989e	600	0.1	$11.25	$2.45	Credit card	Taxi Affiliation Services
da7a5f0865e9f18596ea926fcc0a6 010a955774c	360	2.6	$8.05	$0.00	Cash	Dispatch Taxi Affiliation
da7a62ebde1abb32141f966e2c13 c151b4e74301	540	0.8	$6.65	$0.00	Cash	Top Cab Affiliation

General Social Survey (GSS) Data Set

The General Social Survey has over 5,000 variables collected over a period of more than 40 years. You can explore the data online using a data explorer or download the complete data sets (**http://www.gss.norc.org/Get-The-Data**). Table 1.4 presents a subset of fields from the GSS and their meaning as described in the GSS Codebook (Smith, Davern, Freese, & Hout, 1972–2016), which is available at **http://gss.norc.org/get-documentation**.

TABLE 1.4 ⬢ SUBSET OF FIELDS FROM THE GSS DATA SET

Field Name	Description
YEAR	GSS year for this respondent
ID	Respondent ID number
AGE	Age of respondent
EDUC	Highest year of school completed
REGION	Region of interview
HAPPY	General happiness
REALINC	Family income in constant $

TABLE 1.5 SAMPLE GSS DATA

YEAR	ID	AGE	EDUC	REGION	HAPPY	REALINC
1973	3	36	11	3	2	32761
2008	5	37	12	2	2	17403.75
1974	9	51	12	6	1	0
1990	10	68	12	2	2	28830
1991	10	64	14	2	2	90265
1972	13	54	9	7	2	13537
1982	16	51	18	1	1	90722
2014	16	56	16	1	2	31927.5

The sample data shown in Table 1.5 is in ascending order of the ID value for each record. Unlike the Trip_ID in the Taxi Trips data set, the ID value is not unique in the GSS data, as we can see by the duplication of both ID 10 and ID 16 in the data in Table 1.5. In the GSS data, it is the combination of the YEAR and ID fields that is unique (we call using several fields to uniquely identify a record in a data set a composite identifier or composite key). For example, the respondent with ID 10 in YEAR 1990 is not the same as the respondent with ID 10 in YEAR 1991. Another important difference is that the data in the GSS all appear to be numeric; however, the values are not all quantitative. For example, the values for HAPPY are coded responses to a survey where 1 = *very happy*, 2 = *pretty happy*, and 3 = *not too happy*. Another important point is that the values for REALINC are not actually continuous (even though they might appear to be) but are discrete. These values correspond to the midpoints of income ranges specified in a survey, and the values prior to 1986 have been recoded in six-digit numbers and converted to 1986 dollars (Ligon, 1994; Smith et al., 1972–2016).

Lessons learned: In this section, we learned about the Chicago Taxi Trips and General Social Survey data sets, which will use throughout the text.

Chapter Summary

In this chapter, we learned that Python is free and open-source software (FOSS) and that more than 212,000 projects with packages written in Python are available to use and modify in the Python Package Index. The specific goal of this book is to teach Python programming to those in the fields of social sciences and business to develop applications using Python packages for data analytics. We next learned how to install Python on our computer and that there are different versions of Python. We also learned that using different operating systems and different versions of Python can affect how we write and execute Python code. We then learned how to write and execute Python code in the IDLE shell window and how to write and execute Python code that we save in plain text files. Using files enables the storing and execution of many lines of code as a program that we can save and run later. We also learned how to use package managers for convenient organization and management of libraries of code. Learning to install individual packages is an important skill to take advantage of the many Python packages for both business and social sciences purposes that are available to the Python programming community. Last, we learned about the Chicago Taxi Trips and General Social Survey data sets, which we will use throughout the text.

In the next chapter, we will cover the basic elements of Python code, using the IDLE IDE to illustrate the outcomes of executing each code example. All coding examples in the textbook (including all Stop, Code, and Understand! exercises and their solutions) are available on the companion website to download and execute. We use the figure numbers in the file names of the code examples for easy reference. These code examples are also an example of FOSS, as you can freely modify them and use them in other applications.

Glossary

Case sensitive Interpreting uppercase letters (capitals) as different from lowercase letters.

Free open-source software (FOSS) Inclusive term that covers both free software and open-source software.

Free software Users have the freedom to run, copy, distribute, study, change, and improve the software.

Interactive Development Environment (IDE) Contains facilities for writing and editing code as well as testing and debugging code.

Module A text file that contains Python code.

Open-source software Requires that the license to use the software shall not restrict any party from selling or giving away the software as a component of a larger software distribution.

Package Library of code modules used/accessed by programming code.

Package manager A program to install libraries of code.

pip A package manager that comes already installed in Python.

Platform The combination of a device and an operating system.

Platform independent Software that can run on most, if not all, of the latest operating systems/computing platforms.

Python An interpreted computer programming language.

Python distribution Modified package that includes additional functionality. Also referred to as an alternative bundle.

Syntax A set of rules that dictates how to specify instructions of code in a programming language.

End-of-Chapter Exercises

1.1 Download and install Python on your computer by going to the website **https://www.python.org/downloads/** and following the instructions that correspond to the platform that you are using.

1.2 Enter the Python command `print("Hello, World!")` at the IDLE shell command prompt on your computer after having installed Python to verify that Python was installed properly.

1.3 Using a text editor, create a text file named helloworld.py, which has the single line of code: `print("Hello, World!")`

1.4 Use the IDLE IDE to open the file helloworld.py that you created and execute the code in it.

References

Anonymous. (2018, July 21). And now for something completely different. *The Economist, 428,* 67–68.

Free Software Foundation. (2019). The free software definition. Retrieved from **https://www.gnu.org/philosophy/free-sw.html**

Goth, G. (2007, January/February). Sprinting toward open source development. *IEEE Software*, *24*(*1*), 88–91. doi:10.1109/MS.2007.28

Levy, J. (2017, August 12). Taxi Trips [Data file]. Retrieved from **https://dev.socrata.com/foundry/data.cityofchicago.org/wrvz-psew**

Ligon, E. (1994, June). The development and use of a consistent income measure for the General Social Survey. *GSS Methodological Report No. 64*. Retrieved from **http://gss.norc.org/Documents/reports/methodological-reports/MR064.pdf**

Marsan, J., Pare, G., & Beaudry, A. (2012). Adoption of open source software in organizations: A socio-cognitive perspective. *Journal of Strategic Information Systems*, *21*(*4*), 257–273. **https://doi.org/10.1016/j.jsis.2012.05.004**

Open Source Initiative. (2007, March 22). The open source definition. Retrieved from **https://opensource.org/docs/osd**

Ozgur, C., Colliau, T., Rogers, G., Hughes, Z., & Myer-Tyson, E. (2017). MatLab vs. Python vs. R. *Journal of Data Science, 15*(3), 355–372.

Perkel, J. M. (2015, February 5). Pick up Python: A powerful programming language with huge community support. *Nature, 518*(7537), 125–126. doi:10.1038/518125a

Python Software Foundation. (2019, June 17). Python 3.7.3 documentation. Retrieved from **https://docs.python.org/3/**

Python Software Foundation. (2020). The Python Package Index. Retrieved from **https://pypi.org/**

Scacchi, W. (2004a, January/February). Free and open source development practices in the game community. *IEEE Software*, *21*(*1*), 59–66. doi:10.1109/MS.2004.1259221

Scacchi, W. (2004b). When is free/open source software development faster, better, and cheaper than software engineering? Retrieved from **https://www.ics.uci.edu/~wscacchi/Papers/New/Scacchi-Book Chapter.pdf**

Smets, J.-P. (2019). ERP5: Mission-critical ERP/CRM with Python and Zope. Retrieved from **https://www.python.org/about/success/nexedi/**

Smith, T. W., Davern, M., Freese, J., & Hout, M. (1972–2016). General Social Surveys, 1972–2016 [Machine-readable data file]. Chicago, IL: NORC.

Visit study.sagepub.com/researchmethods/statistics/kaefer-intro-to-python for data sets and code to accompany this text!