**Auomate the Boring Stuff with Python Programming**

Lesson 4

**Flow control**

The Boolean data type has only two values: True and False (both beginning with capital letters).

Comparison operators compare two values and evaluate to a Boolean value: ==, !=, <, >, <=, >=

== is a comparison operator, while = is the assignment operator for variables.

Boolean operators (and, or, not) also evaluate to Boolean values.

Lesson 5

An if statement can be used to conditionally execute code, depending on whether the "if" statement's condition is True or False.

An elif (that is, "else if") statement can follow an if statement. Its block executes if its condition is True and all the previous conditions have been False.

An else statement comes at the end. Its block is executed if all of the previous conditions have been False.

The values 0 (integer), 0.0 (float), and ‘‘ (the empty string) are considered to be "falsey" values. When used in conditions they are considered False. You can always see for yourself which values are truthy or falsey by passing them to the bool() function.

Lesson 6

When the execution reaches the end of a "while" statement's block, it jumps back to the start to re-check the condition.

You can press ctrl-c to interrupt an infinite loop. This hotkey stops Python programs.

A "break" statement causes the execution to immediately leave the loop, without re-check the condition.

A "continue" statement causes the execution to immediate jump back to the start of the loop and re-check the condition.

Lesson 7

A "for" loop will loop a specific number of times.

The range() function can be called with one, two, or three arguments.

The break and continue statements can be used in for loops just like they're used in while loops.

**Function**

Lesson 8

You can import modules and get access to new functions.

The modules that come with Python are called the standard library, but you can also install third-party modules using the pip tool.

The sys.exit() function will immediately quit your program.

The Pyperclip third-party module has copy() and paste() functions for reading and writing text to the clipboard().

Lesson 9

Type programs into the file editor (not in the interactive shell window with the >>> prompt)

The execution starts at the top and moves down.

Comments begin with a # character and are ignored by Python; they are notes & reminders for the programmer.

Functions are like mini-programs in your program.

The print() function displays the value passed to it.

The input() function lets users type in a value.

The len() function takes a string value and returns an integer value of the string's length.

The int(), str(), and float() functions can be used to convert values' data type.

Lesson 10

A scope can be thought of as an area of the source code, and as a container of variables.

The global scope is code outside of all functions. Variables assigned here are global variables.

Each function's code is in its own local scope. Variables assigned here are local variables.

Code in the global scope cannot use any local variables.

Code in a function's local scope cannot use variables in any another function's local scope.

(If there is a global statement for a variable at the top of a function, that variable in that function is a global variable.)

If there's an assignment statement for a variable in a function, that is a local variable. The exception is if there's a global statement for that variable; then it's a global variable.

Lesson 11

**Handling Errors with try/Except**

A divide-by-zero error happens when Python divides a number by zero.

Errors cause the program to crash. (This doesn't damage your computer at all. It's just that the computer doesn't know how to carry out this instruction, so it immediately stops the program by "crashing" rather than continue.)

An error that happens inside a try block will cause code in the except block to execute. That code can handle the error or display a message to the user so that the program can keep going.

Lesson 12

List

Lesson 13

A list is a value that contains multiple values: [42, 3.14, ‘hello']

The values in a list are also called items.

You can access items in a list with its integer index.

The indexes start at 0, not 1.

You can also use negative indexes: -1 refers to the last item, -2 refers to the second to last item, and so on.

You can get multiple items from the list using a slice.

The slice has two indexes. The new list's items start at the first index and go up to, but doesn't include, the second index.

The len() function, concatenation, and replication work the same way on lists that they do with strings.

You can convert a value into a list by passing it to the list() function.

Lesson 14

A for loop technically iterates over the values in a list.

The range() function returns a list-like value, which can be passed to the list() function if you need an actual list value.

Variables can swap their values using multiple assignment: a, b = b, a

Augmented assignment operators like += are used as shortcuts.

Lesson 15

Methods are functions that are "called on" values.

The index() list method returns the index of an item in the list.

The append() list method adds a value to the end of a list.

The insert() list method adds a value anywhere inside a list.

The remove() list method removes an item, specified by the value, from a list.

The sort() list method sorts the items in a list.

The sort() method's reverse=True keyword argument can make the sort() method sort in reverse order.

These list methods operate on the list "in place", rather than returning a new list value.

Lesson 16

Strings can do a lot of the same things lists can do, but strings are immutable.

Immutable values like strings and tuples cannot be modified "in place".

Mutable values like lists can be modified in place.

Variables don't contain lists, they contain references to lists.

When passing a list argument to a function, you are actually passing a list reference.

Changes made to a list in a function will affect the list outside the function.

The \ line continuation character can be used to stretch Python instruction across multiple lines.

**Dictionaries**

Lesson 17

Dictionaries contain key-value pairs. Keys are like a list's indexes.

Dictionaries are mutable. Variables hold references to dictionary values, not the dictionary value itself.

Dictionaries are unordered. There is no "first" key-value pair in a dictionary.

The keys(), values(), and items() methods will return list-like values of a dictionary's keys, vaues, and both keys and values, respectively.

The get() method can return a default value if a key doesn't exist.

The setdefault() method can set a value if a key doesn't exist.

The pprint module's pprint() "pretty print" function can display a dictionary value cleanly. The pformat() function returns a string value of this output.

More about strings

Lesson 19

Strings are enclosed by a pair of single quotes or double quotes (as long as the same kind are used).

Escape characters let you put quotes and other characters that are hard to type inside strings.

Raw strings (which have the r prefix before the first quote) will literally print any backslashes in the string and ignore escape characters.

Multiline strings begin and end with three quotes, and can span multiple lines.

Indexes, slices, and the "in" and "not in" operators all work with strings.

Lesson 20

upper() and lower() return an uppercase or lowercase string.

isupper(), islower(), isalpha(), isalnum(), isdecimal(), isspace(), istitle() returns True or False if the string is that uppercase, lowercase, alphabetical letters, and so on.

startswith() and endswith() also return bools.

‘,'.join([‘cat', ‘dog']) returns a string that combines the strings in the given list.

‘Hello world'.split() returns a list of strings split from the string it's called on.

rjust() ,ljust(), center() returns a string padded with spaces.

strip(), rstrip(), lstrip() returns a string with whitespace stripped off the sides.

replace() will replace all occurrences of the first string argument with the second string argument.

Pyperclip has copy() and paste() functions for getting and putting strings on the clipboard.

Lesson 21

Running program from the command line

Lesson 22

The shebang line tells your computer that you want to run the script using Python 3.

On Windows, you can bring up the Run dialog by pressing Win+R.

A batch file can save you a lot typing by running multiple commands.

The batch files you'll make will look like this:

@py C:\Users\Al\MyPytonScripts\hello.py %\*

@pause

You'll need to add the MyPythonScripts folder to the PATH environment variable first.

Command-line arguments can be read in the sys.argv list. (Import the sys module first.)

10 – Regular Expression

Lesson - 23

Regular expressions are mini-language for specifying text patterns. Writing code to do pattern matching without regular expressions is a huge pain.

Regex strings often use backslashes (like \d), so they are often written using raw strings: r'\d'

\d is the regex for a numeric digit character.

Import the re module first.

Call the re.compile() function to create a regex object.

Call the regex object's search() method to create a match object.

Call the match object's group() method to get the matched string.

Lesson 24

Groups are created in regex strings with parentheses.

The first set of parentheses is group 1, the second is 2, and so on.

Calling group() or group(0) returns the full matching string, group(1) returns group 1's matching string, and so on.

Use \( and \) to match literal parentheses in the regex string.

The | pipe can match one of many possible groups.

Lesson 25

The ? says the group matches zero or one times.

The \* says the group matches zero or more times.

The + says the group matches one or more times.

The curly braces can match a specific number of times.

The curly braces with two numbers matches a minimum and maximum number of times.

Leaving out the first or second number in the curly braces says there is no minimum or maximum.

"Greedy matching" matches the longest string possible, "nongreedy matching" (or "lazy matching") matches the shortest string possible.

Putting a question mark after the curly braces makes it do a nongreedy/lazy match.

Lesson 26

The regex method findall() is passed a string, and returns all matches in it, not just the first match.

If the regex has 0 or 1 group, findall() returns a list of strings.

If the regex has 2 or more groups, findall() returns a list of tuples of strings.

\d is a shorthand character class that matches digits. \w matches "word characters" (letters, numbers, and the underscore). \s matches whitespace characters (space, tab, newline).

The uppercase shorthand character classes \D, \W, and \S match charaters that are not digits, word characters, and whitespace.

You can make your own character classes with square brackets: [aeiou]

A ^ caret makes it a negative character class, matching anything not in the brackets: [^aeiou]

Lesson 27

^ means the string must start with pattern, $ means the string must end with the pattern. Both means the entire string must match the entire pattern.

The . dot is a wildcard; it matches any character except newlines.

Pass re.DOTALL as the second argument to re.compile() to make the . dot match newlines as well.

Pass re.I as the second argument to re.compile() to make the matching case-insensitive.

Lesson 28

The sub() regex method will substitute matches with some other text.

Using \1, \2 and so will substitute group 1, 2, etc in the regex pattern.

Passing re.VERBOSE lets you add whitespace and comments to the regex string passed to re.compile().

If you want to pass multiple arguments (re.DOTALL , re.IGNORECASE, re.VERBOSE), combine them with the | bitwise operator.

Lesson 29

11 : Files

Lesson 30

Files have a name and a path.

The root folder is the lowest folder.

In a file path, the folders and filename are separated by backslashes on Windows and forward slashes on Linux and Mac.

Use the os.path.join() function to combine folders with the correct slash.

The current working directory is the oflder that any relative paths are relative to.

os.getcwd() will return the current working directory.

os.chdir() will change the current working directory.

Absolute paths begin with the root folder, relative paths do not.

The . folder represents "this folder", the .. folder represents "the parent folder".

os.path.abspath() returns an absolute path form of the path passed to it.

os.path.relpath() returns the relative path between two paths passed to it.

os.makedirs() can make folders.

os.path.getsize() returns a file's size.

os.listdir() returns a list of strings of filenames.

os.path.exists() returns True if the filename passed to it exists.

os.path.isfile() and os.path.isdir() return True if they were passed a filename or file path.

Lesson 31

The open() function will return a file object which has reading and writing –related methods.

Pass ‘r' (or nothing) to open() to open the file in read mode. Pass ‘w' for write mode. Pass ‘a' for append mode.

Opening a nonexistent filename in write or append mode will create that file.

Call read() or write() to read the contents of a file or write a string to a file.

Call readlines() to return a list of strings of the file's content.

Call close() when you are done with the file.

The shelve module can store Python values in a binary file.

The shelve.open() function returns a dictionary-like shelf value.

Lesson 33

os.unlink() will delete a file.

os.rmdir() will delete a folder (but the folder must be empty).

shutil.rmtree() will delete a folder and all its contents.

Deleting can be dangerous, so do a "dry run" first.

send2trash.send2trash() will send a file or folder to the recycling bin.

12 – Debug

Lesson 35

You can raise your own exceptions: raise Exception(‘This is the error message.')

You can also use assertions: assert condition, ‘Error message'

Assertions are for detecting programmer errors that are not meant to be recovered from. User errors should raise exceptions.

Lesson 36

The logging module lets you display logging messages.

Log messages create a "breadcrumb trail" of what your program is doing.

After calling logging.basicConfig() to set up logging, call logging.debug(‘This is the message') to create a log message.

When done, you can disable the log messages with logging.disable(logging.CRITICAL)

Don't use print() for log messages: It's hard to remove the mall when you're done debugging.

The five log levels are: DEBUG, INFO, WARNING, ERROR, and CRITICAL.

You can also log to a file instead of the screen with the filename keyword argument in the logging.basicConfig() function.

Lesson 37

The debugger is a tool that lets you execute Python code one instruction at a time and shows you the values in variables.

Open the Debug Control window with Debug > Debugger before running the program.

The Over button will step over the current line of code and pause on the next one.

The Step button will step into a function call.

The Out button will step out of the current function you are in.

The Go button will continue the program until the next breakpoint or until the end of the program if there are no breakpoints.

The Quit button will immediately terminate the program.

Breakpoints are lines where the debugger will pause and let you choose when to continue running the program.

Breakpoints can be set by right-clicking the file editor window and selecting "Set Breakpoint"

13 Web Scraping

Lesson 39

The Requests module is a third-party module for downloading web pages and files.

requests.get() returns a Response object.

The raise\_for\_status() Response method will raise an exception if the download failed.

You can save a downloaded file to your hard drive with calls to the iter\_content() method.

Lesson 40

Web pages are plaintext files formatted as HTML.

HTML can be parsed with the BeautifulSoup module.

BeautifulSoup is imported with the name bs4.

Pass the string with the HTML to the bs4.BeautfiulSoup() function to get a Soup object.

The Soup object has a select() method that can be passed a string of the CSS selector for an HTML tag.

You can get a CSS selector string from the browser's developer tools by right-clicking the element and selecting Copy CSS Path.

The select() method will return a list of matching Element objects.

Lesson 41

To import selenium, you need to run: "from selenium import webdriver" (and not "import selenium").

To open the browser, run: browser = webdriver.Firefox()

To send the browser to a URL, run: browser.get(‘https://inventwithpython.com')

The browser.find\_elements\_by\_css\_selector() method will return a list of WebElement objects: elems = browser.find\_elements\_by\_css\_selector(‘img')

WebElement objects have a "text" variable that contains the element's HTML in a string: elems[0].text

The click() method will click on an element in the browser.

The send\_keys() method will type into a specific element in the browser.

The submit() method will simulate clicking on the Submit button for a form.

The browser can also be controlled with these methods: back(), forward(), refresh(), quit().

14 Excel, word, and PDF Documents

Lesson 42

The OpenPyXL third-party module handles Excel spreadsheets (.xlsx files).

openpyxl.load\_workbook(filename) returns a Workbook object.

get\_sheet\_names() and get\_sheet\_by\_name() help get Worksheet objects.

The square brackets in sheet[‘A1'] get Cell objects.

Cell objects have a "value" member variable with the content of that cell.

The cell() method also returns a Cell object from a sheet.

Lesson 43

You can view and modify a sheet's name with its "title" member variable.

Changing a cell's value is done using the square brackets, just like changing a value in a list or dictionary.

Changes you make to the workbook object can be saved with the save() method.

Lesson 44

The PyPDF2 module can read and write PDFs.

Opening a PDF is done by calling open() and passing the file object to PPdfFileReader().

A Page object can be obtained from the PDF object with the getPage() method.

The text from a Page object is obtained with the extractText() method, which can be imperfect.

New PDFs can be made from PdfFileWriter().

New pages can be appended to a writer object with the addPage() method.

Call the write() method to save its changes.

Lesson 45

The Python-Docx third-party module can read and write .docx Word files.

Open a Word file with docx.Document()

Access one of the Paragraph objects from the "paragraphs" member variable, which is a list of Paragraph objects.

Paragraph objects have a "text" member variable containing the text as a string value.

Paragraphs are composed of "runs". The "runs" member variable of a Paragraph object contains a list of Run objects.

Run objects also have a "text" member variable.

Run objects have a "bold", "italic", and "underline" member variables which can be set to True or False.

Paragraph and run objects have a "style" member variable that can be set to one of Word's built-in styles.

Word files can be created by calling add\_paragraph() and add\_run() to append text content.

16 : GUI Automation

Lesson 48

Controlling the mouse and keyboard is called GUI automation.

The PyAutoGUI third-party module has many functions to control the mouse and keyboard.

pyautogui.size() returns the screen resolution, pyautogui.position() returns the mouse position. These are returned as tuples of two integers.

pyautogui.moveTo(x, y) moves the mouse to an x, y coordinate on the screen.

The mouse move is instantaneous, unless you pass an int for the "duration" keyword argument.

pyautogui.moveRel() moves the mouse relative to its current position.

PyAutoGUI's click(), doubleClick(), rightClick(), and middleClick() click the mouse buttons.

dragTo() and dragRel() will move the mouse while holding down a mouse button.

If your program gets out of control, quickly move the mouse cursor to the top-left corner to stop it.

There's more documentation at pyautogui.readthedocs.org.

Lesson 49

PyAutoGUI's virtual key presses will go the window the currently has focus.

typewrite() can be passed a string of characters to type. It also has an "interval" keyword argument.

Passing a list of strings to typewrite() lets you use hard-to-type keyboard keys, like ‘shift' or ‘f1'.

These keyboard key strings are in the pyautogui.KEYBOARD\_KEYS list.

pyautogui.press() will press a single key.

pyautogui.hotkey() can be used for keyboard shortcuts, like Ctrl-O.

lesson 50

A screenshot is an image of the screen's content.

The pyautogui.screenshot() will return an Image object of the screen, or you can pass it a filename to save it to a file

locateOnScreen() is passed a sample image file, and returns the coordinates of where it is on the screen.

locateCenterOnScreen() will return an (x, y) tuple of where the image is on the screen.

Combining the keyboard/mouse/screenshot functions lets you make awesome stuff!