Python Programming

Python Programming for Beginners

By Adam Stewart

Table of Contents

Introduction

Getting started in coding can be tough. You may have looked at a few of the most popular coding languages, such as C++ or Java and been a bit scared by what you saw. The pages may have been filled to the brim with letters and symbols that you just didn't understand, and you became frustrated and just wanted to walk away. Many people are scared of programing and feel like it is just too hard for them. But with the Python programming language, you will find that it can be easier than ever to learn about coding and to even read it like a professional.

This guidebook is going to give you some of the basics that you need to get started with Python programming. We will start out a bit talking about what Python programming is as well as some of the steps that you should take in order to download the program, if it isn't already present on your computer, and give you some more information to really understand why this program is so great. We will then move on to some keywords that will be useful to you when starting out with the program and even talk about the benefits and the drawbacks of using Python for all your coding and programing needs.

The rest of the guidebook is devoted to talking about some of the different things that you can do within the Python program as well as some examples of how each of these would work. We talk about adding comments into the code, working with strings and integers, and even spend some time working with variables so that they will show up right in the program. It is a great idea to experiment a bit with the process. Python makes it easy to test out your strings so that you can figure out what is going to work and what needs some more practice.

Getting started in programing can seem like a challenge. You may worry that you aren’t going to be able to figure it all out and all of those crazy programing languages may have scared you away in the first place. This guidebook is going to spend some time looking at the Python language and exploring how easy it can be to get started with this simple program.

Chapter 1: Learning About Python

The computer world has brought in many different types of people. Some are interested in making money creating their own programs to sell to others. Some just like to mess around and learn different things about how the computer will work. And still others have devoted their lives to programming, making it the product that brings home their income each month whether they work at repairing computers, work in a corporation to keep the computers safe, or doing some other aspect of computer technology.

When it comes to computer technology, nothing is going to be simple. Before you can even get a program to work on the computer, it needs to receive the right code to make it work. There are several options for code creation that a computer tech can choose including Java, C++ and Python. Here we will explore a bit about Python and why it is often preferred over the other two programming options.

Before you are able to start using Python to take over your programming needs, it is important to start learning more about it and all of the great benefits you will receive when using this program. Python is a high level programming tool, which means that it is easy to use and read, even as a beginner. The philosophy behind the code is readability and it has a type of syntax that allows the programmer to express their concepts without having pages of code along with it. Compared to using other popular codes, like Java and C++, this can make Python much easier to complete.

The philosophy of this code language is simple enough to use. It believes that a simple design is so much better than a complex one and that readability is important. This is a great language for beginners to get started on because they will actually be able to read and understand the code they are putting in. With other options, they may have to spend a lot of time trying to get the code just right, adding in many other symbols to get it to work. But with Python, it is kept much simpler and you may find that it is easier to read through the lines and see what you are doing.

Some of the features that you may like with Python include:

- An elegant syntax which will make the programs so easy to read.
- Language that is easy to use so that the program will work without a lot of bugs. If you are doing ad hoc programming tasks or prototype development because it works well without issues with maintaining the program.
- Has a large library that will work with other programming tasks such as changing files, searching for text, and connecting with web servers.
- Python is really interactive. This makes it easier for you to test out small bits of code to see if they work. You can also bundle it with a development environment called IDLE.
- If you would like to expand the programming language, it is easy to extend into other modules like C or C++.
- Python programming can be run on any unit including Unix, Linux, Windows, and Mac OS X.
- The software is free. You won't have to pay anything to download and use Python in your own life. you can also make modifications and redistribute this product. It is under a license, but it is an open source license so others are able to use it.
- Even though Python is a simple programming language, it does contain some advanced features like list comprehensions and generators.
- Errors can be caught quickly in this programming. Since data types are dynamically typed, when you mix types together that don't match, it will

raise an exception for you to notice.

- You can group the codes into packages and modules if needed.
- There is a wide variety of basic data types that you can choose from including dictionaries, lists, strings, and numbers.

The Origins of Python

The beginning of the modern Python programming started in December of 1989. The creator of this program was Guido van Rossum who began programming as more of a hobby. At the time, van Rossum was working on a project with the Dutch CWI research institute, that was later terminated. Van Rossum was able to use some of the basics of this new language, known as the ABC language, in order to work on Python.

The main strength of this language is that it is really easy to extend upon to make more complex, or keep simple, and it was able to support multiple platforms. Both of these were important during the days when personal computers were becoming popular. And since Python was designed to communicate with different file formats and libraries, it became a hit as well.

Python has grown quite a bit since its inception and more tools have been added to make the programing more functioning. In addition to making Python easy to use, van Rossum has been working on initiatives that encourage the education of coding to everyone, not just a select few. Using Python to use coding can make things easier and helps to get rid of some of the fears associated with the complex computer codes since it doesn’t look so scary.

Over the years, van Rossum decided to make Python open sourced. This allowed all to gain access and make changes to Python so that if something happened to van Rossum, all would not be lost. Thanks to having Python open sourced, Python 2.0 was released during 2000 to make it more community oriented and to have a transparent development process. There are a few newer versions of Python 2.0 still being used, but Python 3 has been taking the world by storm and most anticipate this will be the normal one used within the next few years.

Python 3

This version of Python was released in 2008. It is not simply an update to the program, but a complete change in it. While there are a lot of great features that come with this version, it doesn’t have a backward compatibility so you will have to make a choice between Python 2.0 and Python 3. To make things easier, the programmers did make a little marker within the program that would show a coder what needed to be changed between the two programs when uploading. Despite this, most have stuck with Python 2.0 for now.

Why Use Python?

As you can probably guess, there are several different computer coding programs that you can choose to use. But while there are some benefits to using these other programs, Python is one of the best options out there. It is easy to use, has a lot of options for you to choose from, and it can even be used over a variety of platforms without having to change things up. Some of the benefits that you will love with Python include:

Readability

Python is designed to work with the English language, making it easy to read. There are also strict rules in terms of punctuation on the program so you aren't just looking at brackets all over the place. Python also makes sure that the programmer knows how to format everything thanks to a set of rules that are in place, making it easy for everyone to create a code that others can follow.

Libraries

Python has been around for over 25 years now and since it is one of the easiest codes to learn how to use, there have been quite a few different codes written using the system. The good news is that this system is open sourced so that the code is available for any programmer to use. You can install the Python program in your own system and use it for your own personal use. Whether you are using the codes to finish off a product or to write some of your own codes, the library of Python is easy to use. The codes that you want will be installed into the libraries and since the program has been around for a long tie, they are going to cover pretty much whatever you want from automating your server to making changes to a picture.

Community

Since Python is so popular, the community for Python is pretty big. There are conferences with lots of networking and workshops available for this programming products and lots of places you can visit, both online and offline, to ask questions or to learn more about the program. You may want to consider checking out a few of these places if you are a beginner with Python as it can help you to learn more and even to meet some new people.

If you are interested in getting started with coding, Python is one of the best options that you can make. It is simple to get started on and since it will work on a variety of different platforms, it is sure to work on your personal computer. Since it is easy to read, you will find that coding doesn’t have to be a challenge and you can create your own, or learn from others in no time.

Chapter 2: The Benefits and Negatives of Python

```
            default='Y',
            )
    global_scale_setting = FloatProperty(
            name="Scale",
            min=0.01, max=1000.0,
            default=1.0,
            )

    def execute(self, context):

        # get the folder
        folder_path = (os.path.dirname(self.filepath))

        # get objects selected in the viewport
        viewport_selection = bpy.context.selected_objects

        # get export objects
        obj_export_list = viewport_selection
        if self.use_selection_setting == False:
            obj_export_list = [i for i in bpy.context.scene.objects]

        # deselect all objects
        bpy.ops.object.select_all(action='DESELECT')

        for item in obj_export_list:
            item.select = True
            if item.type == 'MESH':
                file_path = os.path.join(folder_path, "{}.obj".format(item.name))
                bpy.ops.export_scene.obj(filepath=file_path, use_selection=True,
                                        axis_forward=self.axis_forward_setting,
                                        axis_up=self.axis_up_setting,
                                        use_animation=self.use_animation_setting,
                                        use_mesh_modifiers=self.use_mesh_modifiers_setting,
                                        use_edges=self.use_edges_setting,
                                        use_smooth_groups=self.use_smooth_groups_setting,
                                        use_smooth_groups_bitflags=self.use_smooth_groups_bitflags_setting,
                                        use_normals=self.use_normals_setting,
                                        use_uvs=self.use_uvs_setting,
                                        use_materials=self.use_materials_setting,
```

Python is a great program to use whether you are a beginner in the programming world or you have been into it for some time. Many of those who are just looking at getting started with programming will jump right in with Python and make it their own. It is simple to understand and can be used by anyone who is ready to get started with coding. This chapter is going to take some time to explain some of the positives, as well as some of the drawbacks, of using Python for your programming language.

The Benefits of Python

Python is probably one of the best programming languages that you can choose to use. Beginners are going to love how easy it is to turn on this program and start writing their own codes, even without experience, and there is plenty to enjoy when you are a professional, or an expert, as well. Some of the benefits that you will get when you get started with Python include:

Easy to use and read

When it comes to programming language, there are none that are as easy to use as Python. Other languages are kind of clunky and hard to look at. You may take a look at them and notice that they have tons of brackets and even words that you won't even recognize. It is enough to scare away someone who isn't used to programming at all just because all the words look a bit intimidating.

Python is a bit different. Instead of all the crazy brackets, it makes use of indentations, causing an easier to read page that isn't such a mess. Instead of words that you can't understand, it uses English. The other special characters are kept to a minimum so that you can look at the page of code and not feel like you are going to be overwhelmed in the process.

This is one of the easiest programming tools that you can use. It looks nice on the page and will use plenty of white spaces, when it can, to make it easier to read what you should know. There are also plenty of places with comments so you can get clarification if a program is too confusing for you. Overall, it is one of the best programming languages to use to really get ahead or even to learn about programming.

Uses English as the main language

Since English is the language that this program is based off, it is really easy to read. There aren't a lot of words that you won't get and you won't have to spend time trying to figure out what it is telling you. The program is all in English and you will love how simple this can make things.

Already present on some computers

In some cases, Python is already present on your computer. Mac OS X systems as well as those with Ubuntu will already have Python preloaded. You will simply need to download a text interpreter to get started. In terms of using Python on Windows computers, all you need to do is download the program. Python works with all of these programs, even if it isn't installed right from the beginning.

Can work with other programming languages

In the beginning, you will most likely only use Python on its own. It is a great program to learn with and grow with. But over time, you may decide that you want to try something new that Python can't do on its own. Luckily, Python is able to work with several other programming languages, such as C++ and JavaScript, so you can mess around, learn some more, and really get the code that you are looking for, even if Python is not able to do all the work.

Can test out things with the interpreter

When you download Python, you are going to have to download a text interpreter too. This will make it easier for Python to read through your information. You can use simple products that are sometimes already on your computer, such as Notepad from Windows or look for another interpreter that may be a bit easier.

Once you pick out the interpreter that you would like to use, it is time to get to work writing the code. Some of those who are new to coding may feel worried about trying to get the code to work. This is another spot where Python can make things easier. It will be able to take the words that you are typing and spit them back out, with the help of the interpreter, in just a few seconds. You can test what you are doing while you are working on it!

There are so many benefits of using the Python program. Beginners are going to love how readily available this program is and how easy it is to learn some of the simple commands in no time. Even those who have been programming for some time will be impressed by how this all works!

The Negatives of Python

While there are a lot of reasons to love Python, it is important to realize that there are a few negatives that you should watch out for. These negatives include:

It doesn't have a lot of speed

For those who are looking to work with a program that has a lot of speed, Python may not always be the best option for you. It is an interpreted language so this will slow it down compared to some of the other options that are compiled languages. However, it does depend on what you are translating.

There are certain benchmarks with the Python code that can run faster using PyPy compared to other codes.

Luckily this issue with a slow speed and Python is being remedied. Programmers are working to make the interpreting speed of Python faster so that you won't have to compare it with the others so much. Over time, the hope is that Python will be able to work at the same speed as C and C++ or even some of the newer programming languages that are coming out.

Not present on most mobile browsers

Python is a great option to use if you have a regular computer. It is available on many desktop and server platforms to help you create the code that you are looking for. But it is not ready to go into mobile computing. Since there is such a big increase in revenue and people going into the mobile industry, it is sad that this programming language hasn't kept up with the trends like others.

Perhaps in the future Python will decide to go into the future and develop a version that will be able to work well with various mobile devices. Until then, programmers will have to be satisfied with using it on their desktop and laptop computers.

Restrictions with the design

If you are looking to work with a program that has a lot of design options, the Python program may not be the right option for you. The design language is not up to what you will find with some of the other option. Since you are working with a program that is dynamically typed, it takes more testing and can have more errors that will only show up when you are running the program.

The global interpreter lock means that you can only have one thread access the internals of Python at a time. This may not be as important anymore since it is easy to spawn the tasks out to different processes, but the design is not as nice as some of the other options that you would like.

A good way to work with the design is to remember that indentation is important with Python. Other programming languages are going to use a lot of brackets to show the difference in lines and information inside the program, but Python is going to rely on indentations. Make sure to be careful with using this to avoid issues and errors that can come up.

Python can be one of the best programs that you use to write your own codes and have some fun. While there are a lot of benefits to using this program, especially compared to some of the other ones that aren't as easy to read, it is important to understand both the positives and negatives of each option before you jump in!

Chapter 3: Common Terms You Should Know with Python

Before you get too far into your programming with Python, it is important to understand some of the words that can make the programming easier to understand. This chapter is going to take some time to look at the different words that are common in Python programming, and which we do talk about a bit in this guidebook, to help avoid some confusion and to help you get started with your first code.

- Class—this is a template that was used for creating user-defined objects.
- Docstring—this is a string that will appear lexically first expression inside a module, function, or class definition. The object will be available to documentation tools.
- Function—this is a block of code that is invoked when using a calling program. It is best used in order to provide a calculation or an

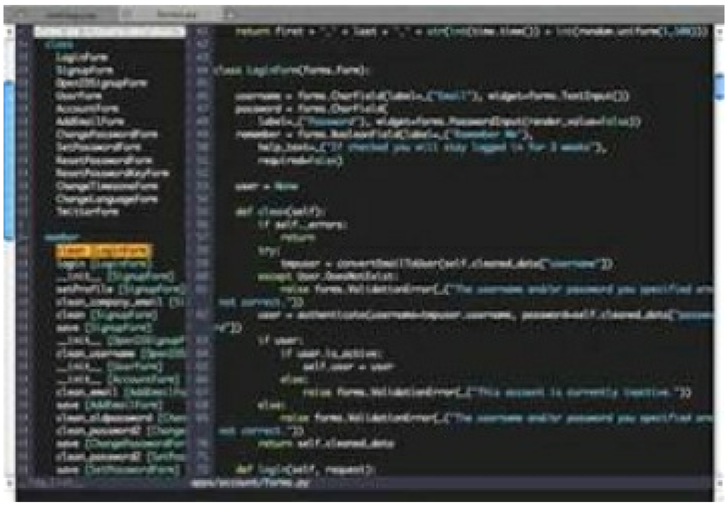
autonomous service.

- IDLE—this stands for Integrated Development Environment for Python. This is the basic interpreter and editor environment that you can use along with Python. It is good for those who are just beginning with this and can work for those on a budget. It is a clear example of code and won't waste a lot of time or space.
- Immutable—this is an object within the code that is assigned a fixed value. This could include tuples, strings, and numbers. You can't alter the object and you will need to create a new object with a different value and store it first. This can be helpful in some cases, such as the keys in a dictionary.
- Interactive—one thing that a lot of beginners like about Python is that it is so interactive. You can try out some different things in the interpreter and see how they will react right away in the results. It is a good way to improve your programming skills, test out a new idea you have and more.
- List—this is a datatype within Python that is built in. It contains a mutable sequence of values that are sorted. It can include immutable values of numbers and strings as well.
- Mutable—these are the objects that will be able to change their value within the program, but which are able to keep their original id().
- Object—within Python, this is any data with a state, such as a value or an attribute, as well as a defined behavior, or a method.

- Python 3000—Python 2 and Python 3 are the main two types of Python that are available. Many people have stuck with Python 2 since Python 3 does not have any backwards capabilities and they like using the databases on the older version. Python 3000 is a mythical option of Python that does allow this backward capability so you can use it and the Python 2.
- String—this is one of the most basic types that you will find in Python that will store the text. In Python 2, the strings will store text so that the string type can then be used to hold onto binary data.
- Triple quoted string—this is a string that has three instances of either the single quote or the double quote. It could have something like '"I love tacos"'. They are used for many reasons. They can help you to have double and single quotes in a string and they make it easier to go over a few lines of code without issues.
- Tuple—this is a datatype that has been built into Python. This datatype is an immutable ordered sequence of values. The sequence is the only part that is immutable. It can contain some mutable values, such as having a dictionary inside it, where the value's can change.
- Type—this is a category or sort of data that is represented in the programming languages. These types are going to differ in their properties, they including immutable and mutable options, as well as in their functions and methods. Python includes a few of these including dictionary types, tuple, list, floating point, long, integer, and string.

-

Chapter 4: Getting Started with Python



Now that we know some of the benefits of choosing this program, it is time to get started with it. Before you are able to learn some of the great steps that are needed to make this program create code for you, it is time to set up the environment. For those who have a computer with Mac OS X or Ubuntu, you will already have Python installed on the system This can make things easier to get started as you will just need to click on the icon to get started.

Windows computers will need to install Python. While Python works just fine on Windows computers, it doesn’t come preinstalled so you will need to do this. The following steps work for Windows 7 to 10:

- Download Python—you can choose between Python 2 or Python 3. Both are fantastic options; it just depends on which one will get the job done for you.

- Click to run the Python Installer. When you get to the options, choose to Customize Installation.
- You will see a box pop up. Click on every box that is under Optional Features and then continue.
- On the next screen, look for the Advanced Options and then choose where you would like to have Python install.

Once you have gotten this far, the next part is to set up your PATH variable. This is going to allow the user to include directories for all the components and packages that are needed. To do this step:

- Open up the Control Panel on the Windows computer.
- Look up Environment.
- Under System Environment Variable, click on Edit. Then click on Environment Variables.
- You may have to look a bit for the next part, but look for User Variables. You can then either create a new one or edit an existing path.
  - To create a new path, select PATH as the name and add it to the directories that are there. Make sure that each Variable Values is separated with a semicolon.
  - If you want to edit your existing path, you need to make sure that each value is on a different line. Click on New and then put your

directories on different lines.

- Now you can open your command prompt. To do this click on Start then Windows System and then Command Prompt.
- When the command prompt opens, you can type in “Python.” This will load up the Python interpreter. You can then type in Exit and hit Enter to get back to the command prompt.

Text Editor

You will not be able to program Python without having the text editor in place on your computer. If you are using Windows, the Notepad function will work. Make sure that you are not using Word though, it is not considered an editor and your code is not going to save on the system properly. If you are considering getting a version of Notepad, you will notice that Notepad ++ is the best one to use on a Windows computer and Text Wrangler is the best to use for Mac. To set them up, do the following:

Windows

Download and then install Notepad ++

Once it is downloaded, open up the settings and click on Language Menu and

Tab Settings

Tick the box that is beside Expand Tabs

Make sure that the value is at 4.

Click again to Close.

Mac

Download and then install Text Wrangler

You won't need to register to install the software, just click Cancel if there is a box that comes up asking for this.

Otherwise, follow the other instructions that come on the screen to set this editor up.

Once the program is on your computer it is time to learn more about the coding and functions that you can enjoy on Python.

Getting IDLE

While you are setting up Python, make sure that you download the IDLE or the

Integrated Development and Learning Environment. This should download along with Python if you are setting it up, but make sure to check into this while you are going through the process. This is the environment that you are going to work with when you are on Python and it can make things easy. If you don't want to mess around with finding another environment or you want to make the process as easy as possible for you as a beginner, this is the option for you.

The main features of using IDLE with your Python programming include:

- Integrated debugger with persistent breakpoints, call stack visibility, and stepping to make things easier
- Python shell that will highlight the syntax
- Multi-window text editor that can help with the indentation, highlighting, and completing the code.

Now, you can choose to use another environment, like those that we discussed above if needed, but since this one often comes as an option with Python and it is designed to work well with this system, there are many people who choose to go with this option. That being said, there have been some issues in the past with IDLE having trouble focusing, won't copy some things, and some clients don't like the interface design. You may want to try out this program ahead of time and see if it is the right one for you or if you would like to use one of the options above.

Getting Python set up on your computer is a pretty easy process. There are already several types of computers that have the programming language already present so you won't have to do any work and the rest of them simply need a quick download to complete. You can wait just a short amount of time to get Python on your computer and then you are good to go and try out some of the codes you want to make.

Chapter 5: Learning the Basics of Python Programming

Now it is time to get to know a bit more about Python programming and how you can make it work for you. You will need to learn a bit more about the different keywords and the variables that come with Python so you are able to write the words that you want and make the program perform in a certain way. Let’s take a look at some of these basics of Python programming so you can get started with your new code right away.

Keywords

When you are working on a new computer coding program, you are going to notice that each computer language will have certain keywords. These are the words that are meant for a specific command or purpose in the language and you should try to avoid using them anywhere else. If you do use these words in other parts of your code, you may end up with an error alert or the program not working properly. The keywords that are reserved for Python include:

- And
- Pass
- Or not
- Nonlocal
- None
- Lambda
- Is
- In import
- If
- Global
- From
- For
- Finally
- False
- Except
- Else
- Elif
- Del
- Def

- Continue
- Class
- Break
- Assert
- As
- Yield
- With
- While
- Try
- True
- Return
- Raise

Identifier Names

When you are creating a new program in Python, you are going to work on creating quite a few entities, a combination of functions, classes, and variables. All o these will be given a name that is also known as an identifier. There are a few rules that you need to follow when forming an identifier in Python including:

- It should contain letters, either uppercase or lowercase or a combination of the two, numbers, and the underscore. You should not see any spaces inside.

- The identifier can't start with a number
- The identifier can't be a keyword and it shouldn't include one of the keywords inside.

If you break one of these rules, the program will close on you and will show a syntax error. In addition, you need to work on making identifiers that are legible to the human eye. While the identifier may make sense to the computer and get through without causing issues on the computer, a human is the one who will read through the code to use it themselves. If the human eye doesn't understand what you are writing in a certain place, you could run into some issues. Some of the rules that you should follow when creating an identifier that will be readable to the human eye include:

- The identifier should be descriptive—you should pick out name that is going to describe what is inside the variable or will describe what it does.
- You should be careful with using abbreviations that aren't necessary because these always make things that are difficult.

While there are a lot of ways that you can write out your code, you should be careful and stick with one rule throughout. For example, both MyBestFriend and mybestfriend work in the coding world, but pick one that you like and do it the same each time that you work in the program to avoid confusion. You can also add in underscores into this or numbers, just be careful that you keep things consistent.

Flow of Control

When working on the Python language, you are going to write out the statements in a list format, just like you would when writing out a shopping list. The computer will start with the first instruction before working through each of them in the order that you make them show up on the list. So you will need to write out the controls that you want just like you would for your grocery shopping list to make sure that the computer is reading it properly. The computer will only stop reading through this list once it has done the final instruction to completion. This is known as the flow of control.

This is an important way to get started. You want to make sure that your flow of control is even and smooth for the computer to read. This will make it easier to get the program to do what you would like without as many issues and ensures that the computer program doesn't get stuck, cause issues, or have something else go wrong.

Semi-colons and Indentation

When you look at some of the other computer languages, you will notice that there are a lot of curly brackets used to arrange the different blocks of code or to begin and end the statements. This helps you to remember to indent the code blocks in these languages to make the code easier to read, although the computer will be able to read the different codes without the indentations just fine.

This type of coding can make it really difficult to read. You will see a lot of unnecessary information that is required for the computer to read the code, but can make it hard on the human eye to read this. Python uses a different way of doing this, mostly to help make it easier on the human eye to read what you have. You are going to need to ident the code for this to work. An example of this is:

```
# this function definition begins a new block
def add_numbers (a, b):
c= a + b
# as is this one
```

```
return c

# this function definition begins a new block
if it is Saturday
print (It’s Tuesday!”
# and this one is outside the block
print (“Print this no matter what.”)
```

In addition, there are a lot of languages that will use a semicolon to tell when an instruction ends. With Python though, you will use line ends to tell the computer when an instruction will end. You will be able to use a semi-colon if you have a few instructions that are on the same line, but this is often considered bad form within the language.

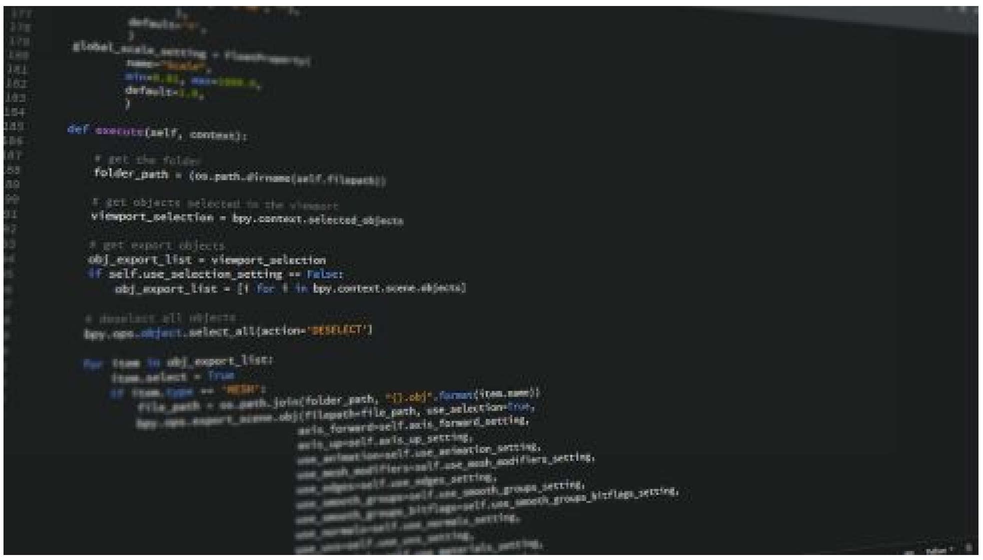
Letter Case

Most computer languages will treat uppercase and lowercase letters the same, but Python is one of the only ones that will be case sensitive. This means that the lower case and upper case letters will be treated differently in the system. Keep in mind as well that all the reserved words will use lower case except for None, False, and true.

These basics are going to make it easier to get started on the Python programming. You need to take a bit of time to go through the program in order to get familiar with it. You aren’t going to need to become an expert, but getting familiar with some of the text interpreter and some of the other parts of the program can make it easier to use and you can learn how the different buttons will work even before you get started. Try out a few of the examples above first to help you get started.

Python works to keep things as basic as possible because it understands that most of its users are going to be beginners or those who are tired of other complex languages. As you can see here, and in the following chapters, there are simple commands that you will be able to put forward in order to get the program to work a specific way. Study these and you can make a great program without quite as much work.

Chapter 6: A Bit More on Comments



```
            defaults=[illegible],
            )
    global_scale_setting = FloatProperty(
            name="Scale",
            min=0.01, max=1000.0,
            default=1.0,
            )

    def execute(self, context):

        # get the folder
        folder_path = (os.path.dirname(self.filepath))

        # get objects selected in the viewport
        viewport_selection = bpy.context.selected_objects

        # get export objects
        obj_export_list = viewport_selection
        if self.use_selection_setting == False:
            obj_export_list = [i for i in bpy.context.scene.objects]

        # deselect all objects
        bpy.ops.object.select_all(action='DESELECT')

        for item in obj_export_list:
            item.select = True
            if item.type == 'MESH':
                file_path = os.path.join(folder_path, "{}.obj".format(item.name))
                bpy.ops.export_scene.obj(filepath=file_path, use_selection=True,
                                        axis_forward=self.axis_forward_setting,
                                        axis_up=self.axis_up_setting,
                                        use_animation=self.use_animation_setting,
                                        use_mesh_modifiers=self.use_mesh_modifiers_setting,
                                        use_edges=self.use_edges_setting,
                                        use_smooth_groups=self.use_smooth_groups_setting,
                                        use_smooth_groups_bitflags=self.use_smooth_groups_bitflags_setting,
                                        use_normals=self.use_normals_setting,
                                        use_uvs=self.use_uvs_setting,
                                        use_materials=self.use_materials_setting,
```

There are a lot of things that you can do in Python. It is one of the most interactive options that you will run into when getting started in programming and since it is so easy to use. In this chapter, we will take some time to discuss more about comments and some of the other aspects of Python so you are able to get started and make your codes amazing in no time.

In Python programming a comment is one that will start with the # sign and then will continue on until you get to the end of the line. For example:

# This would be a comment

print(“Hi, how are you?)

This would tell the computer to just print “Hi, how are you?” All comments are ignored in the Python interpreter because it is more of a footnote in the program to help the programmer, or others who may use the code, special things about the code. They are basically there to say what the program is supposed to do and how it will work. It is a bit more detailed and can be helpful without getting in the way of how the code works.

You will not need to leave a comment on every line, just when it is needed. If the programmer feels that something needs explained better, they would put in a comment but don’t expect to see it all over the place. Python doesn’t support any comments that will go across several lines so if you have a longer comment in the program, figure out how to split it up into different lines with the # sign in front of each part.

Writing and Reading

Some programs are going to show the text you want on the screen, or they can request certain information. You may want to start out the program code by telling the reader what your program is all about. Giving it a name or a title can make things easier so the other coder knows what is in the program and can pick the right one for them.

The best way to get the right information to show up is show a string literal that will include the “print” function. For those who don’t know, string literals are basically lines of text that will be surrounded by some quotes, either a single or double quote. The type of quote that you use isn’t going to matter that much, but if you use one type in the beginning of the phrase, you should use it at the end. So if there are double quotes at the beginning of your phrase, make sure that you keep up with the double quotes at the end as well.

When you want the computer to display a word or phrase on the screen, you would simply have “print” and then the phrase after it. For example, if you want to portray “Welcome!” you would do

Print(“Welcome!”)

This will make it so that “Welcome” shows up on your program for others to use. The print function is going to take up its own line so you will notice that after putting this in, the code will automatically place you on a new line.

If you would like to have the visitor do a certain action, you can go with the same kind of idea. For example, say you want the person to input a specific number so that they can get through the code you would use the string:

```
first_number = input(‘put the first number in’)
```

When using the input feature, you won’t automatically see it print on a new line. The text will be placed right after the prompt. You will also need to convert the string into a number for the program to work. You don’t need to have a specific parameter for this either. If you do the following option with just the parentheses and nothing inside, you will get the same result and sometimes makes it easier.

Files

For the most part, you will use the print function to get a string to print to the screen. This is the default of the print function, but you can also use this same function as a good way to write something onto a file. A good example of this is

```
With open(‘myfile.txt’, ‘w’) as myfile:
Print(“Hello!”,file=myfile)
```

Now this may look like a simple equation, but there is quite a bit that is going on in the string above that you should watch out for. In the spot *with* you opened up the myfile.txt to write on and then assigned it to the variable called myfile. Then in the second part, you wrote in Hello! To the file as a new line and then the *w* told the program that you will only be able to write the changes when the file is open.

Of course, you don't have to use the print function to get it to do the work that you want. The *write* method will often work well too. For example, you can replace the print with write like the example below to get the same things.

```
With open('myfile.txt', 'w') as myfile:
myfile.write("Hello!")
```

So far we have learned how to print a string of words into the program and even how to save them to a specific file. In addition to those options, you can use the read method in order to open a specific file and then to read the data that is there. If you would like to open and read a specific file, use this option:

```
With open('myfile.txt', 'r') as myfile:
```

data = myfile.read()

with this option, you will be able to tell the program to read the files contents into variable data. This can make it easier to open up the programs that you would like to read.

Built In Types

Your computer is capable of processing a lot of information including numbers and characters. The types of information that the Python program will use are known as types and the language will contain many different types to help make things easier. Some of these include string, integers, and floating point numbers. Programmers can even define these different types using classes.

Types will consist of two separate parts. The first part is a domain that will contain a possible set of values and the second part is a set that contains the possible operations. Both of these can be performed on any value. An example of this is that if you have a domain that is a type of integer, it can only contain integers inside it including addition, division, multiplication, and subtraction.

One thing to note with this is that Python is a dynamically typed program. This means that there really isn’t a need to specify the types for the variables when you create it. The same variables can be used to store the values of different types. Despite this, Python still needs you to have all the variables with a definitive type. For example, if the programmer tried to add in a number to a string, the Python program would recognize this and show an error. It won’t try to figure out what you wanted; rather it will just exit without trying.

Integers

If you want to use integers as a type, you need to keep them as whole numbers. These can be positive or negative numbers, as long as there are no decimals with these numbers. If you have a decimal point in the number, even if the number is 1.0, you will need to use it as a floating point number instead. Python is able to display these integers in the “print” function, but only if it is the sole argument.

Print(3)

# Let’s add two numbers together

Print(1+2)

If you are using integers, you will not be able to place the two right next to each other. This is mainly because of how Python is a strongly typed language and won’t recognize them if you combine them together. If you would like to put the number and the string together, you need to make sure that the number has turned into a string.

Operator Precedence

One thing that you need to keep track of when you are working in Python is operator precedence. For example, if you have 1+2//3 Python could interpret it as (1+2)//3 or 1+(2//3). Python has a method that will help you to order the operation properly so that you get the right information to come up. For example, when it comes to integer operation, Python is going to handle everything that is brackets first. Then it will handle the things that have**, then *, and then //, then %, +, and finally -.

If you are writing an expression that has a number of operations in it, you will need to keep those signs in mind. This will tell Python how to go through the numbers so that you can get the right answers at the time. Keep in mind that most arithmetic operators are going to be left associative so write it out that way for Python to read. The only exception is the ** feature. For example:

```
# ** is right-associative
2**3**4
# will be evaluated right to left:
2**(3**4)
```

Strings

While a string may seem like something complicated, in Python they are basically a sequence of characters. They are going to work the same way as a list does, but they will contain a bit more functionality that is specific to the text.

Formatting strings can be a challenge when it comes to writing out your out your code. There are some messages that aren't going to be fixed string and sometimes there are values that are stored inside variables inside it. There is a way to get this to work right for string formatting. An example of this is:

```
Name = “Janet”

Age = 24

Print(“Hello! My name is %s.” % name)

Print(“Hello! My name is %s and I am %d years old.” % (name, age))
```

The symbols that have a % first are called placeholders. The variables that go into these positions will be placed after the % in the order where they are placed in the string. If you are doing just a single string, you will not need a wrapper, but if you do have more than one of these, you need to place them into a tuple, with a () enclosing it. The placeholder symbols will start with different letters, depending mostly on the variable type you are using. For example, the age is going to be an integer by the name is a string. All of these variables are going to be converted into the string before you can add them into the rest.

Escape Sequences

Escape sequences can be used as a way to denote special characters that can be hard to type on your keyboard. In addition, they can be used to denote characters that can be reserved for something else. For example, using and in the sequence can confuse the program so you may use the escape sequence to replace that like the following example:

Print(“This is a line. \nThis is another line.”)

Triple Quotes

We have spent a bit of time talking about both single and double quotes, but there are times when you may need to bring in the triple quote. This is used when you need to define a literal that will span many lines or one that already has a lot of quotes in it. To do this, just use a single and double together or three singles. The same rule applies with the triple quote as with all the others. You will need to star and end the phrase with the same one.

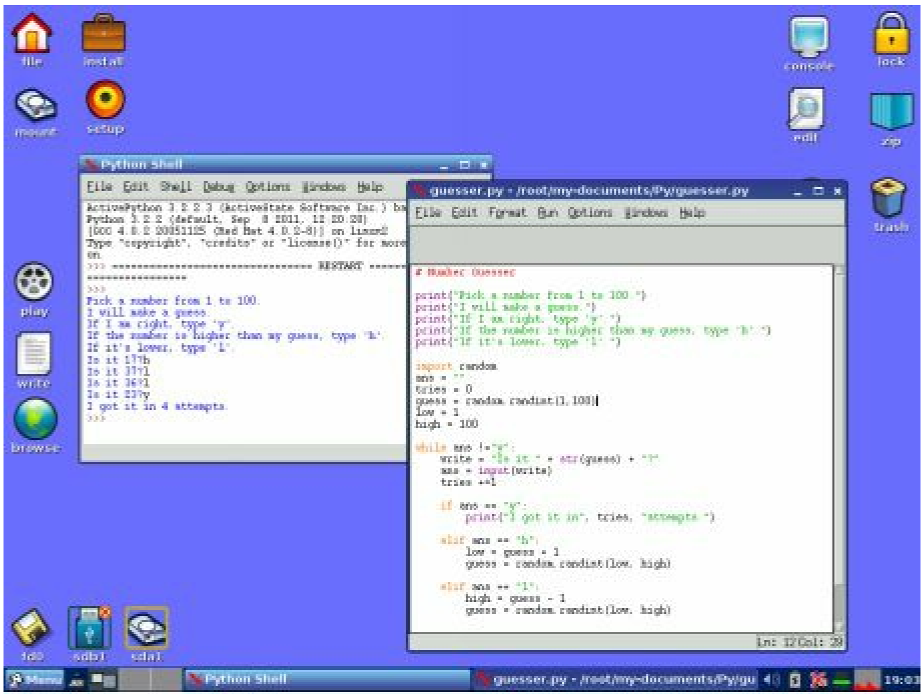
String Operations

One of the string operations that you may use a lot is a concatenation. This is used in order to join a pair of strings together and you will notice it is there with the + symbol. There are a lot of functions that Python is able to help you with and they will work with the strings to create a variety of operations. They are going to have some useful options that can do a lot more in the Pythons program

In Python program, strings are called immutable. This means that once you create the string, it is not capable of being changed. You may have to assign a new valuable to a specific variable that exists if you are looking to make some changes.

There is so much that you are able to learn about when it comes to getting started with Python. It may be a simple language, but you want to be able to learn how it works, how to write things down properly, and even how to leave a comment for others to understand when they are looking through the code. It may seem a bit intimidating in the beginning, but before too long, and with some practice, you will get it down and be writing your own code in no time.

Chapter 7: Variables and What They Do in Python



```
ActivePython 3.2.2.3 (ActiveState Software Inc.) ba
Python 3.2.2 (default, Sep  8 2011, 12:20:28)
[GCC 4.0.2 20051125 (Red Hat 4.0.2-8)] on linux2
Type "copyright", "credits" or "license()" for more
on.
>>> ================================ RESTART ======
================
>>>
Pick a number from 1 to 100.
I will make a guess.
If I am right, type 'y'.
If the number is higher than my guess, type 'h'.
If it's lower, type 'l'.
Is it 17?h
Is it 37?l
Is it 36?l
Is it 23?y
I got it in 4 attempts.
>>>
```

```
# Number Guesser

print("Pick a number from 1 to 100.")
print("I will make a guess.")
print("If I am right, type 'y'.")
print("If the number is higher than my guess, type 'h'.")
print("If it's lower, type 'l'.")

import random
ans = ""
tries = 0
guess = random.randint(1,100)
low = 1
high = 100

while ans !="y":
    write = "Is it " + str(guess) + "?"
    ans = input(write)
    tries +=1

    if ans == "y":
        print("I got it in", tries, "attempts.")

    elif ans == "h":
        low = guess + 1
        guess = random.randint(low, high)

    elif ans == "l":
        high = guess - 1
        guess = random.randint(low, high)
```

The next thing that we are going to discuss are variables. Variables are basically the labels that will denote where in your computers memory something is going to be stored and they can also hold values. When it comes to programming that is typed with statistics, the variables will each have a value that is predetermined and each variable is only going to hold the value of that type. Python has made it a bit easier because you can use one of your variables in order to store different types.

Think about your calculator for this one. The variable is going to be like the memory function in this calculator. It will hold onto a value so that you can retrieve it any time that you want, but when you store in a newer value, the older one will be erased. The only difference is that you will be able to have a large number of variables and each of them will have different values, each of them being referred by their own name.

With Python you will be able to define a variable by giving the label a value. For example, you can name a variable *count* and have it an integer value of one. You would show this by simply writing

```
count = 1
```

Note that with this syntax, you can assign a value to the variable with the same name. If you try to access values in a variable that hasn't been defined, the Python interpreter won't read through this. It will just exit out of the program and give you an error.

You can choose to define a few different variables in one line, but this is not the best practice for you to use. For example, you could do this:

```
# Let's define three variables at the same time:
count, result, total = 0, 0, 0
```

And while that is the correct way to do things, it is much better to show it like this:

```
# This is the same as:

Count = 0

Result = 0

Total = 0
```

It is much easier to read the second way and will ensure that the Python program is going to understand what you want it to say.

Understanding the scope of a variable

You won't be able to access every variable from all parts of the program and not every variable will be the same length. The way that you defined the variable is going to determine where and how long you will be able to access this variable. The section of your program where you can access the variable is going to be known as the "scope" and the time that the variable will be available is known as the "lifetime".

Global variables are those that are defined within the main file body and you will be able to see these variables throughout the entire file as well as inside a file that will be able to import the specific file. These variables have far reaching effects and because of this, you may notice some consequences that you didn't notice. This is why most people won't use global variables, or they will use them sparingly. You should only add stuff into the global namespace if you plan to use them globally, like with functions or classes.

On the other hand, if you define a variable inside of another variable, it will be called a local variable. This one has the ability to be accessed from where it is defined and will only exist when that function executes. These are only going to be available in certain areas of the program and can't be found or used elsewhere.

The assignment operator

We have discussed this option a bit throughout the book, but haven't really given it a name. The assignment operator is the equals sign or the (=). It is going to be used in programming to assign the value to the right of the statement to the variable that is located t the left. Sometimes the variable will be created first. In cases where the value on the right is from an expression, such as an arithmetic expression, the evaluation will take place before this assignment happens.

Keep in mind that the (=) is not going to be a mathematical sign in programming. You can add things to the number and make all sorts of changes that wouldn't make sense if you thought of this sign as a mathematical one. Rather it is an assignment operator so that the statement will be turned into the part on the right.

When you assign the first value to this variable, you are going through the process of initializing. The definition of a value assignment and variable are carried out in the single step in this programming, although it is sometimes done in two steps with some of the other programming languages. But since it is done in one step, it is less likely that the user will make a mistake or receive an error in the process.

Modifying values

In some programming languages, you will be able to define a special variable that has a value that has been set. This means that the value can't be changed. These are called constants in the programming language. For the most part, Python is not going to allow for these kinds of restrictions, but there is a convention that is used to help ensure that some variables are marked to indicate that the values aren't supposed to be changed. To show this, the names will be written in CAPITAL letters with underscores between each word. An example of a variable that is a constants include:

```
NUMBER_OF_DAYS_IN_A_WEEK=7
NUMBER_OF_WEEKS_IN_A_YEAR=52
```

Of course, there are no rules to say you have to put the right number at the end. You could say there are 8 days in a week if you want because the Python program won't be keeping track, but it is best to just keep it accurate in case other coders would like to use it.

These can be really helpful to you in your string. Sometimes in the program, for example, you will want to change the maximum of a number that is allowed in the program. This may work fine for a bit, but maybe later on you need to increase or decrease this number. Without setting up a constants, you are going to have to go through and make quite a few changes to get everything matched up. But with a good constants in order, you can just go back to one place and get it all fixed up.

Understanding how the strings work in your program can make a big difference in the success that you see with this program. You need to learn where they are stored, what the rules are that govern each of them, and how to make them work in a specific part of the program. With a bit of practice, and using the guidelines above, you will get this down in no tie and can be an expert too!

Conclusion

Learning how to get started with computer programing can seem like a big challenge. There are many different programming options that you can go with, but many of them are hard to learn, will take some time to figure out, and won't always do all of the stuff that you need. Many people fear that they need to be really smart or have a lot of education and experience in coding before they are able to make it to the coding level they want. But with Python, even a beginner can get into programming.

Python has made it so easy to get started with coding whether you are a beginner or have been in this business for some time. The language is based in English so it is easy to read and it has gotten rid of a lot of the other symbols that make coding hard to read for others. And since it is user domain, anyone can make changes and see other codes to make things easier.

This guidebook has spent some time talking about the different functions that you can do in Python and how easy it is for a beginner to get started. You will find that this process is easy and you can learn it with a little bit of practice. It is easy to use, works across a lot of platforms, and even the newer Mac systems come with this already downloaded.

When you are ready to get started on programming, or you want to find a program that is going to do a lot of great things without all the hassle, make sure to check out Python. This is one of the most popular options when it comes to programming and you are going to find that it is easy to read and learn, even if you have no idea how to start in the first place. Use this guidebook to learn some of the basic functions and to learn a bit more about the Python program.