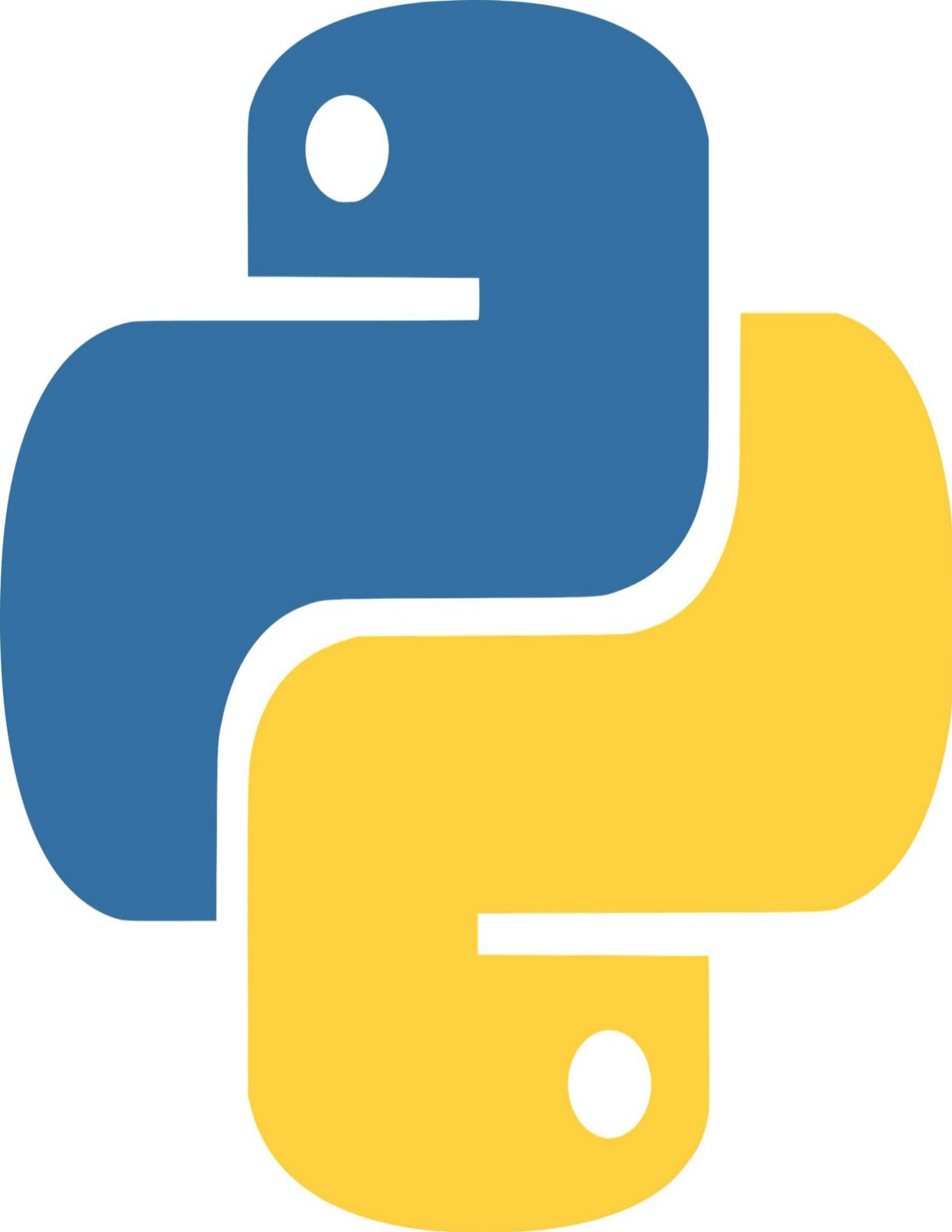

Python Programming for Beginners

Easy Steps to Learn the Python Language and Go from Beginner to Expert Today!

Table of Contents

Chapter 1: Starting Out With Python

When it comes to the world of coding, there is so much to learn about. There are different coding languages that you can work with, different commands that you can learn that will make the program act in a certain way, and so much more that you can learn in the process. Sometimes as a beginner, it is confusing to understand how the coding languages work. Maybe at some point you have seem some complicated code and felt that it is too hard for you to learn how to do. Maybe you have listened to others talk about code at some point, and it just seemed to go over your head. While there is some technical aspects that are found with many coding languages since they do work to tell the computer how to behave, once you learn some of the basics, it doesn't have to be such a hard thing to learn.

If you are interested in learning how to do some basic coding, and to perhaps change it to something more advanced down the line, then you need to learn how to work in the Python programming language. This is one of the easiest of the programming languages because it is based on English (so there aren't any complicated words to learn along the way) and it is set up to be really easy to read. Even if you have no experience with coding in the past, you will find that reading through this language can be pretty simple.

There are many different coding languages that you will be able to work with in your time in this field and some of them may have more options or more power compared to working with Python. C++ is a popular coding language that does a lot of hacking and other commands on the computer while a program like Java and JavaScript is perfect for setting up web pages online. But for a good overall program that can get you started in coding, and will build up some of your confidence in how this process works, Python is one of the best that you will

find.

When working in Python, you will be able to notice its philosophy fairly early on. This programming language was built on the idea of being simple to use. Rather than making things complicated and too hard to hand, you will be able to learn this language quickly and a lot of the readability that you need is found inside of Python. As a beginner, you will appreciate being able to look at the code and understand what is there, rather than just taking our word for what you write. With other options, you may worry that you missed out on a little part or that you are doing it all wrong just because it is set up to be so complicated; this just isn't the case with Python and most beginners appreciate how easy the design of this language is.

As you start to work with Python, you will notice that there are quite a few features inside that make this program one of the best for beginners to learn with. Some of the benefits and features that you will enjoy when working on Python include:

- The syntax is elegant and easy to read, even for the beginner.
- The language is really easy to use so not only will you be able to read through it, but you can also avoid a lot of the bugs and problems that come with the other coding languages.
- The Python library is really large, so you will be able to look here for guidance for some of the tasks that you want to work with inside of Python. The library is going to include commands that help with connecting to web servers, changing files, searching for text, and so much more. As a beginner, you should spend some time looking through this library because it can provide you with much of the information you need to get to learning some of these codes.

- The Python interface is one of the most interactive out of the coding languages. This can make it easy for a beginner to test out their code to see what is working for them. If you want to experiment a bit and see if one thing works over another, the Python language makes that possible.
- At times, you may want to expand out the work that you are doing into something more complex or powerful. Python works great with some other coding modules, including C++ and C so you have this option down the road.
- It doesn't matter what kind of programming you are using, or operating system, on your computer. Python is set up to work with all of the operating systems including Mac, Windows, Unix, and Linux.
- The Python program is free. You can download it onto your computer and not have to worry about paying anything on it. This is an open source program so you can download this for free and keep it for your own personal use or you can choose to make changes and develop a new update if you feel that brave.
- There are some advanced features that come in this programming language, even for it being so simple. You will enjoy some of these features, such as generators and comprehensions as you get more advanced in your coding knowledge.
- It is easy to find errors that occur in the Python language. Since the different data types are going to be dynamically types, when these types come together and get mixed, it is going to raise up an exception that you will need to deal with inside of your code.
- You will enjoy that there is the possibility to group together codes into modules and packages whenever you need.
- And finally, you can enjoy that there are many data types that are allowed inside of the Python program. You can pick from the data

types of numbers, lists, strings, and dictionaries.

Setting up the Python program

It is pretty easy to get the Python programming to work on your system. There are a few parts that you are going to need to place inside the program to make it work first, but overall you will be fine. First, you need to make sure that you have some kind of text editor in place. This is the area where you will write out the codes that you want to work with inside of the Python language. It doesn't have to be something complicated. When you are working on a Windows computer, you could use the Notepad feature to be your text editor and then there are a few other options that you are able to use to be the text editor to go along with the Python language.

Once you have chosen the text editor that you would like to use, it is time to download your Python programming. Since this is free, you will be able to pick the version that you like the most and then get it down on the computer in no time. You can go and visit python.com to get a list of the versions that are available and then you will be able to pick out the one that you would like to use. While you are downloading this onto the computer, make sure to download the IDLE along with it because your program is not going to work without this environment being in place. Once you have this all on the computer, it is time to work on Python and start making some of your own codes.

Reasons to Use Python

As we mentioned a bit before, Python is just one of many programming languages that you are able to choose to use. There are many of them out there and some are going to be more complex and have more power than you could imagine and much more than you will be able to get from Python. It often

depends on the kind of work you would like to get done with the coding language which will determine which of the types that you choose. With all these other options, why would you consider working with Python over one of the others. Here we will discuss some of the benefits of choosing Python rather than one of the other coding languages, especially when you are just getting started in coding for the first time.

Readability

One of the first things you will notice when you get started with Python is that it is really easy to read through. It is based off the English language and the first time you take a look at it, you may give a sigh of relief because it will look easy to read through. Python is also an easy language to learn how to format as long as you know some of the basic rules. Anyone is going to be able to learn how to write in this language and it is easy to make something, even as a beginner, that other programmers can read and understand.

Libraries

When you are new to programming, the library of your chosen coding language can be really important. These libraries are going to contain a lot of the commands and syntaxes that you will need and that work inside the Python language. Since this is an open sourced program, there are thousands of codes inside of the library and other programmers are always developing more that can be placed inside.

This is good news for someone who is a beginner. If you are interested in learning a new code inside the system or you aren't sure where to put something inside the code you are developing, you can head to the library and often find the information that you need. This can even be nice when you want to save time

because you could insert one of the codes that you find through the library into your own code and put it to work.

Community

One of the nice things about working in the Python language is that you will be able to find a good community to work with. Python is one of the most popular coding languages and because of this, it has a large community. You will be able to find national conferences that will contain many of the Python products and you can find forums both online and offline where people are constantly asking questions and getting advice.

As a beginner, it is a good idea to check out some of these communities. They are full of more advanced users, as well as other beginners looking for advice, and if you get stuck on something, want to learn something new, or have issues with the programming, this is the place where you will find your answers.

Usability

When you are beginning with a new programming language, especially if you don't have any experience with the coding field, you want to make sure that the program you are using is really usable. You don't want to spend months trying to learn how to use the new code because this can get frustrating and make you want to give up. Yes, there is a little work in learning a new code, but most people want to feel that it is something they can do, with a bit of work, rather than something that they will never be able to figure out.

With the Python language, you are going to find that you are working with a language that is really usable. It doesn't have a lot of tricks inside of it and you will find that you could even read through some of the coding before learning

anything about it. Add together a bit of knowledge of this programming language, and you will be able to make it your own in no time.

Python is one of the easier coding languages to learn how to use and after a bit of practice, you will find that you can write out some of your own codes in no time. It can be a lot of fun to learn a brand new coding language and Python is going to be one of the best tools that you can use to get this done. If you are a beginner who wants to start making some of your own coding, this is the right coding language to get you started on.

Chapter 2: How to Work with Your Files in Python

Now that we have some of basics of working in Python and understand why you may want to go with this programming language compared to one of the other ones, it is now time to learn some of the basics that come along with this coding language. There are going to be many times when you are creating something in Python and you will want to store this data in a way that is easy to access at a later time. You may even want to have it display in the program, as a part or a whole, and you want to make this work out easily for you.

When you want to store the data in this way, you should make a new "file", but you will have times when you can reuse the same code over again inside the programming language. Let's take some time in this chapter to create some of these files so that they will stick around when you are creating a new code in Python.

There are a few types of operations that you are able to use and have carried out when you are working inside of your code and in the file mode. You can think of this like the same process that you accomplish when you are working in Word on a Windows computer. You will need to save the document inside of Word when you are done and when you want to have it come back up at a later time, but rather than saving a document, you are going to do this saving with some of the code that you write out instead. Some of the operations that you are able to do with your files inside of Python include:

- Create a new file
- Close up a file
- Moving a file
- Editing a file so that it has more code inside of it.

All of these will provide you with some choices when it comes to working inside of your files and when making sure that the code is where you would like it to be. Learning how to make these all work can make it easier to work in Python, but after you do them a few times, you will find that this can be a really simple process.

Create your new files

So the first thing that we are going to take a look at is how to create a new file. If you would like to write onto a file before you get it created, you need to simply open up the file and choose which of the modes that you want to write out your code in. There are three basic choices that you can use when opening up the file and these are going to help choose what exactly you are doing inside of the program. The three modes that you can make for this include mode(x), append(a), or write(w). Any time that you want to make some changes to the file, such as originally writing inside of it, you will want to use the write(w) mode to get this started.

If we are working on creating a new file, by writing out statements or strings, like what will happen with most of our binary files, you will need to just stick with the write(w) method. This one is great because you will be able to write out the code that you want to use while creating a new file, and it keeps things pretty simple and all together.

The write(w) function is pretty simple to use and there are so many things that you can create with this function. You can get a new file started by writing things out on the code or you can make some changes to what is already inside the file. It all depends on what you would like to see inside of your code when you are all done. The following is a good example of how you would use the write(w)

function to help you to get a new file started inside of Python:

```
#file handling operations
#writing to a new file hello.txt
f = open('hello.txt', 'w', encoding = 'utf-8')
f.write("Hello Python Developers!")
f.write("Welcome to Python World")
f.flush()
f.close()
```

When you are working on this particular code, you are telling the compiler that you want all the things that are created in this file to end up in the current directory since you didn't tell the system a different location where you wanted to send everything. So if you would like to find this, you would be able to find this hello.txt file inside your current directory. Take a moment to look for this file and then try to get it to open. You should notice that there will be a message that comes up on your screen that says "Hello Python Developers! Welcome to Python World!" as long as you typed everything into the code properly.

Now that we have written a simple code inside of this language, it is time to take it all to the next step. You could choose to rewrite some of the parts of this language so that they will show up with different messages on the screen or in a different way. You will need to make some changes to the code that you are using, but this is still a pretty simple process to complete. Below is a simple example of what you are able to do when you want to make some changes to your syntax in order to rewrite your code:

```
#file handling operations
#writing to a new file hello.txt
f = open('hello.txt', 'w', encoding = 'utf-8')
```

f.write(“Hello Python Developers!”)
f.write(“Welcome to Python World”)
mylist = [“Apple”, “Orange”, “Banana”]
#writelines() is used to write multiple lines in to the file
f.write(mylist)
f.flush()
f.close()

The above example is going to be a simple example of how you will be able to make some changes inside of your file since you just need to add in an additional line. You will be able to write in as many of these lines as you would like to get some good results. If you typed all of this inside of your compiler and asked the compiler to interpret the results, you would get this message to come across your screen: “Hello Python Developers! Welcome to Python World. Apple Orange Banana.”

Working on your binary files

In some instances, when you are working on your data, you may want to write it out as a binary file. This may seem a bit complicated at the beginning, it is actually a pretty simple to work with. You will simply be writing out your data as a sound or image file rather than a text file in this process. You can easily change any of the text that you are working on into a binary file, regardless of the file type that you were using in the beginning, you just need to follow the right steps to get it changed. You must remember to supply the data into the object form so that the compiler is able to expose this as a byte. A good way to write out the syntax for doing this includes:

 # write binary data to a file
 # writing the file hello.dat write binary mode

F = open(‘hello.dat’, ‘wb’)
 # writing as byte strings
f.write(b”I am writing data in binary file!/n”)
f.write(b”Let’s write another list/n”)
f.close()

Once this has been written into the compiler, you can then bring out your notepad so that you can see what is all written for the program. It is important to remember that you must decode and encode your functions so that it is easier to read and write with these codes once you are in the binary file mode. To get this to happen, you will just need to write out the following syntax:

 # write binary data to a file
 # writing the file hello.dat write binary mode
f = open(‘hello.dat’, ‘wb’)
text = “Hello World”
f.write(text.encode(‘utf-8’))
f.close()

Opening up a new file

So far we have taken a look at how you are able to design the file so that it will save for you to find later. But now it is time to move over to opening up the files that you have saved so that you can use them again. Any code that you have saved this far has been stored on some disks inside whichever computer is using them, so you just need to find them, bring them back up, and then you are able to read all the way through them. A good syntax that you should use when you want to open up the file includes:

 # read binary data to a file

#writing the file hello.dat write append binary mode

```
with open(“hello.dat”, ‘rb’) as f:
        data = f.read()
        text = data.decode(‘utf-8’(
print(text)
```

the output that you would get form putting this into the system would be like the following:

Hello world!
This is a demo using with
This file contains three lines
Hello world
This is a demo using with

This file contains three lines.

You can use this to open up any of the files that are on your system. This is a simple syntax and makes it easier to find the files that you would like and open them up so that you can either read them again, make some changes, or use the codes that you have written. As you can see, this is a simple process, part of why Python is so popular to use, and you will be able to use it to open up any of the files that are on your system.

Moving your files

In addition to writing out a new file, saving it, or opening up one of the files that you saved, there are also going to be times when you will want to move one of the file locations. You may want to put them in a new place that is easier to find

or you may have made a new folder that you want to put a few of the files inside for easier access. There may even be times when things don't match up the way that you would like, such as words that are not spelled right in the code or something else, and making some moves could make it easier to work within the files of your system.

You can always choose to move some of the file locations so that they end up in the right spot, or so that they are easy for you to find on the system. You will just need to take some time to go inside and find the file and tell your code where you would like to have it moved. Once your program finds the new location, you can easily go inside, make the changes, and get them all moved. This may sound a bit complicated in the beginning, but you will find that it can be pretty easy once you try it out.

So the trick here is to find out where the original file is currently located before you are able to move it. For the most part, if you didn't specify where the file should go, it is going to be in the current drive until you move it around. You can go to the current drive, or wherever you stored the information inside the system, and open up the file using the tricks that we did earlier. Now that you are inside the system, you will be able to make some changes to the code, choose where you would like to move it (such as giving it a new name or finally specifying where you would like to place it), and so much more.

Learning how to work within the files on your system is a great way to get some more experience within Python. This may seem like a simple method that you are able to work on and like it shouldn't need a code or other steps to accomplish, but it is really a good way to start learning how to use the system and how to not only save some of the code that we will be writing in this guidebook, but also to open it up later when you would like to use it again as well as how to move the code to a new location if needed. There is just so much

that you are able to do when in Python when it comes to some of the file and code that you are writing, and learning how to do some of these simple steps is the right way to get the program written fast.

Chapter 3: What are Exceptions and How Do They Work in Python?

While there are a lot of things that can make working inside the Python language more enjoyable, one thing that you will enjoy the most is the big library that comes with it. This is going to make it so much easier to get the codes that you are working on to come out the way that you would like. When working inside of this Python library, you are basically working with things that are known as regular expressions. These are the parts that are responsible for executing any task that you want without any glitches and sometimes they will even help to handle the searches that you are doing. There are many times when you will need to use these regular expressions to help filter your text and you can use them to check on the strings of text to see if they do match up with some of the regular expressions you are working on.

Once you learn how to use the regular expressions inside of Python, you are going to be set. The syntax that you are working with will stay the same whether you are working in just Python or if you combine Python with one of the other coding languages that you want. Plus, the syntax for those other coding languages is going to be similar to the coding that you find inside of Python so once you learn how to do one of the coding languages, it should be so much easier to work on some of the others.

At this point, we have just looked a bit at what these could all be about, but you may be confused as to what the regular expressions are all about. One way that you can see these regular expressions and how they work is to find a word that you spelled a bit differently inside of the text editor. This could be a misspelling or some other issues, but let’s say that you spelled word blue and then the next one as bleu. Even though these are going to have different spellings, you will be able to use your text editor to make the two work together correctly.

Now, when you want to take these regular expressions and make them work inside of your program, you must make sure that you are importing the right expressions from the Python library. A good place to do this is right at the beginning of your program to keep things organized. When you open up the program, you can go through the process and import all the expressions that you need out of the library. This not only helps you to write out the code that you should be using, but it is also easier on those who are trying to read your program later on.

There are many expressions that are found inside the Python library and you will need to take a look at them and figure out the one that is right for your needs. Often these expressions are going to be used at the same time as the statements so that these statements are able to execute in the proper way. This means that you need to have a good understanding of how each expression works because if you use the wrong one with your statements, you could end up with the wrong execution happening.

Luckily, there are many expressions that you are able to use inside of your code and you will be able to find the one that will help the program work the way that you would like. You simply need to learn how these expressions work and place the right ones in the code and then you will see that your commands will quickly come to life.

Basic Patterns

In addition to helping you to set up some of the commands that you want your statements to execute, you will also find that these regular expressions are good at helping you to make up and specify the patterns that you should have within your code. There are a number of different patterns that can be used inside of the

Python programming and it is a good idea to know which ones you can use for your coding. These include:

1. Ordinary characters. These are characters that will match themselves exactly. Be careful with using some of these because they do have special meanings inside of Python. The characters that you will need to watch out for include [], *, ^, $
2. The period—this is going to match any single except the new line symbol of '\n'
3. 3. \w—this is the lowercase w that is going to match the "word" character. This can be a letter, a digit, or an underbar. Keep in mind that this is the mnemonic and that it is going to match a single word character rather than the whole word.
4. \b—this is the boundary between a non-word and a word
5. \s—this is going to match a single white space character including the form, form, tab, return, newline, and even space. If you do \S, you are talking about any character that is not a white space.
6. ^ = start, $ = end—these are going to match to the end or the start of your string.
7. \t, \n, \r—these are going to stand for tab, newline, and return
8. \d—this is the decimal digit for all numbers between 0 and 9. Some of the older regex utilities will not support this so be careful when using it
9. \ --this is going to inhibit how special the character is. If you use this if you are uncertain about whether the character has some special meaning or not to ensure that it is treated just like another character.

Make sure that you learn how these patterns work because they are going to really help you to communicate with the interpreter and let it know what you are

working on inside the code you create. You should learn how some of these symbols work and practice inside the interpreter to see what is going to come up when you try out different things. If you use the symbols in the correct way, you will find that your code is going to execute beautifully and there shouldn't be any problems. On the other hand, if you mess up on a sign or on something else within the code, you may find that you get an error message, the interpreter gets a bit confused, or the wrong messages come up on your screen.

Doing a query with the help of expressions

At this point, we are going to work on using these expressions in order to create a query. There are a few different queries that you will be able to use and it often depends on what you want the regular expression to do before you pick the method that you will use. There are three types of methods that work for this kind of action and they include:

re.findall()
re.match()
re.search()

Right now these are all just some words in the code that you may not understand all that well, but in the next few parts, we are going to look at what each of them mean, how these options are a bit different from each other, and help you to learn when you are supposed to use each one in your code writing.

The search method

Using the search method in your coding is pretty easy. You will just need to use the search() as the syntax and you are ready to go. This one is going to be nice because you are able to match together things anywhere and you won't have to

be restricted with just finding matches that are at the beginning of the string like you would with some of the other query options. If you want to look through the whole string and see if there are any matches to what you are looking for inside that string, the search() function is the one that is right for you. To get a good idea of what is going on with this option, check out the following syntax:

```
import re
string = 'apple, orange, mango, orange'
match = re.search(r'orange', string)
print(match.group(0))
```

When you take some time to place this code into your interpreter, you will find that the output is going to be "orange". You will be able to find the pattern whether it is at the beginning of the code or later on, you need to remember that with the search() function, it is only going to find the code for you once. In this example the search function found that there was one match of orange because that is all that is there. But, you could have ten more oranges in this code, and it is only going to show up the word "orange" once because that is the word that has a match.

Match Method

Now we are going to move on to the second query option that you can work with. This option is going to be slightly different compared to the search function that we did above, but it can also be helpful inside of the Python language. Using the match function is going to work in that it will find the matches that show up right at the beginning of the string. If there is a match that occurs anywhere except right at the beginning of the string, it is not going to show up in the code.

So when you take a look at the code that we had in the last section, if you used

the match function, you would find that nothing would come up in the output. Yes, there is the pattern of using orange throughout the code, but since the word orange is not the first word in the string, the match function is not going to work like this. And since apple doesn't have a match throughout the rest of the string, you will end up with no results. If apple was recurring, or if you moved the orange around to be the first word in the string, then these could be the outputs to these codes.

This option is basically going to be helpful if you have a term at the beginning of the string and you want to do a quick look through the statement to see if there are any patterns that match up with the first word. It can save you some time when looking for the patterns that you want to use.

The findall method

The next command that we are going to look at when we are in Python and we want to use regular expressions is the findall method. If you are working with a statement and you would like to release all of one word out of this string, using the findall method is going to be one of the best choices. While this sounds complicated, it is basically going to work like the search function, but instead of just giving you one result, it is going to tell you how many of that pattern are repeating inside the code.

So, taking a look at the code we have been using, you could use the findall method to find all of your oranges. It will recognize that there is a pattern inside of the code and then it will release all of the items that match this pattern. With the pattern that is above, if you used the findall method, you would get two oranges showing up at the output. You can expand this out as much as you would like depending on how many of these end up inside your code. If there is a pattern of oranges and there ends up being a hundred oranges in the pattern, then

the findall method would pull up a list that had 100 oranges for you to see.

This can be useful if you have a smaller string and you would like to know how many are inside the pattern. But if the string gets pretty long, you probably don't want to use it because who wants to have a list of 100 or more oranges on their screen. As mentioned above, it all depends on what you would like to get out of this code.

As we discussed inside this book, there are many regular expressions that you will be able to work with. They are going to help you to work inside the library of Python and can be amazing for getting you on the right track when it comes to finding patterns. Depending on what you would like to do within the code, the search, match, and findall method can really help you to get the information that you want out of the code and to get the right message to display on the screen for your needs.

Chapter 4: What Does Object Oriented Programming Mean?

As you work with programming, you may hear a few times abut Object Oriented Programming, or OOP. This is a new approach that many programming languages, like Python, are using in order to keep the program organized during development. It is basically designed as a way to eliminate some of the pitfalls that were found in conventional methods of programming.

The OOP is going to do this by using the best program features in a more structured way as well as some new powerful concepts. While it may seem a bit confusing to you, this is a new way that you will be able to design some of your programs, how you are able to use Python along with some of the other programs, and why Python is so easy to use, even for the beginners.

When most programming languages first came out, they were developed using the procedural approach. While this did get the job done and made some amazing code for many years, it did have a lot of flaws inside of it. These flaws were the things that often made beginners in coding run away and not want to try it out.

Because the older programming approach was harder to use, the OOP was developed. With OOP, you will find that data is treated as an important development rather than allowing the data to just flow freely throughout the system. It is going to keep the data tied up with the function that will operate it, helping to keep the data safe and ensure that no accidental modifications occur from outside sources.

For you, this simply means that the programming language is easier to use and

still has all the power that you will look for. Some of the features that you will really enjoy when using programming languages that have OOP include:

- Pulling the emphasis away from the procedure and putting it more on the data.
- The programs will be divided up into objects.
- The structures of data in these programs are designed so that they can be characterized by their objects.
- The functions that are used to operate the data of an object are going to tie together inside of this data structure.
- The data will be hidden so that none of the external functions will be able to access this data.
- The objects will be able to communicate with each other thanks to having common functions.
- The new data and their functions can be added at any time if you want.
- You will notice that when you are designing the program, it is going to follow the bottom up approach.

Some concepts to understand in OOP

There are many concepts that you can learn about when you want to work with OOP. Most of the ones that you should learn at this point to make your codes better include:

- Classes
- Objects
- Inheritance
- Data encapsulation and abstraction
- Dynamic binding

- Polymorphism

Each of these are pretty unique to OOP and will ensure that you are going to be able to use this great program in your own code. Let's take some time to look at these concepts and how they are going to work to make your code more powerful and work properly.

Classes

When you are thinking about classes, you should think of them like the blue prints or the design of the Python program. The whole set of data and code for the objects that you are using can be defined when the class is there to help. For the most part, the objects are going to be variables of the type of class that you want to work with.

When the user has taken the time to define their class, they are then able to go on and create as many objects as they would like to belong within that same class. The objects will then be associated with the data type of class that they were originally created within.

For the most part, the class is just going to be a collection of objects that are grouped together because they have some similarities that link them together in some way. They can have the same actions, the same colors, or the same use. You are able to determine the way that the objects are classified together. For example, you could place all fruits in one class, all the objects that are the color blue, all kids toys, or something else that will link the objects together.

When you are making the classes, you should ensure that they make sense. You want the things that are in the class to be there because they make sense. If you put in a firetruck, an ice cube, a car, a bush, and a slide, it may be hard for

someone to figure out what the class is all about. But if you pick something like a Christmas tree, Santa Clause, elves, and presents, the user would be able to determine that these are items that are for the Christmas season.

Objects

We discussed this a bit above, but objects are basically the items that are going to be placed inside the classes and they will be the basic entities to run time inside of the OOP program. These can represent almost anything from a table with a bit of data, a person, an item, a place, or even a building. There are not limitations on what can be considered an object. It just needs to be something that you want the interpreter to execute.

Some of the other things that you may want to classify as an object in your code include user defined content such as vectors, time, and lists. In addition, if you even up having some issues with your programming, your interpreter is going to analyze the code in term of the objects that you use and how they are communicating with each other. The objects that you choose in your classes and in the codes need to be chosen carefully, ensuring that the objects you are using will match closely with each other as well as with other real world objects so that the interpreter is able to show them correctly.

Abstraction and data encapsulation

When you take the data and the function from one unit, or the class, you are doing the process that is known as encapsulation. This is one of the most amazing features that you will see within the class. The data, once it has gone through the process of encapsulation, will not be accessible to the outside world and only any functions that are inside that class will be able to access it.

This is going to ensure that no one else is able to make changes to your classes when you are done creating them, that only the function that is attached to that class will be able to make the changes in the code. These functions are good for the interface when you want the data and the program of the object to work together well. This insulation will help to keep the data away from access of the program and it is a process that is known as either data hiding or information hiding.

Another option that you may want to use is abstraction. The process of abstraction is the act of representing some essential features of the class without adding in the explanation or the details about the background of that class. The classes that you are using are going to use abstraction to help keep it safe and will be defined by the different attributes such as size, wait, function, and cost to name a few.

Inheritance

Inheritance is a great feature that you are able to use on Python and we will discuss how to go about this process a bit more in a later chapter. It is going to be used to help you to use features from a previous code and expand it to work with a newer code, one that you are able to make different changes to over time.

When you are on OOP, the concept of using inheritance is the idea that things are able to be reused within the program. This means that you will be able to pick out a parent that you would like to use in the code and then change it into the child code before making the changes that you need to continue on the code.

This is going to save you a lot of time. You will be able to use one of the classes that you already developed in order to make something new. You will not need to go through and make a whole new code completely since you will be able to use some of the information that you already have. Plus, with inheritances, you will be able to make changes. You are not stuck with just using the same parent code each time, the parent code is going to give you the basis that you need, but you can add more power and functions to the original code.

Polymorphism

The next thing that you are able to work on in your code is to do a polymorphism. You are going to be able to create an operation that is able to take on behaviors that come from different instances. The behavior that comes up is going to depend on the types of data that you plan to use in the operation, but you are able to make the code work based on the functions that you choose. A good example of this is when you have a class that is about shapes and then you give it a draw method in there as well.

You would use this kind of method when you want your object to do more than one thing during the execution. Perhaps you want the ball to be able to roll and bounce when it comes out or something else that would work well with your coding. You can take just one of the objects and get it to do many different things. The best part about this is not only will the code help the object to do many different actions, but you will not need to use as many lines of codes to make it happen, helping your code to stay organized and neat looking.

Dynamic binding

It is also possible to work with dynamic binding when you are working on your OOP language. Binding refers to the linking of a procedure call to the code to be

executed in response to the call. But when you are working with dynamic binding, this means that the code will be associated to the given procedure, but the call is not known until you call this out at the run time. You are often going to see this happen with some of the other features of OOP including with inheritance and with polymorphism.

The OOP options are one of the best to use in your programming language because they are simple, there is a lot that you are able to do with it, and it is easier to use without as many complications like you would get with some of the older coding languages. Python as well as some of the other modern programming languages will use this option, making it one of the best to use if you want to have safe programming that is easy for you to make and the user to use properly. And as this chapter shows, there are a lot of great things that you will be able to do with your code when you use a programming language that utilizes OOP.

Chapter 5: The Importance of Classes and Objects in the Python Code

We spent quite a bit of time talking about objects and classes in the last chapter and how they are going to work together. The classes are going to make it easier for you to place different objects together so that you are able to pull them out at the same time when they are needed within the code, but both are so important to making the code work out the way that you want.

Keep in mind that while you can make the objects of a class be anything that you like, you should consider placing ones that are similar in the same class. This just makes it easier to keep things together and can give your code a more coherent look. With a bit of organization and thinking ahead, you are going to be able to keep all the parts of your code in order so that you get the interpreter to read them exactly the way that you want.

Before we get too far with creating classes, there are a few things that you need to keep in mind including:

- Objects that are found inside the same class are going to have the same structures, but they are allowed to differ a bit. For example, if you have a class of vehicles, you can put any vehicle you want inside of it including cars, vans, trucks, and more.
- How to create objects by using classes with the help of instantiating objects.
- Classes are the basic blueprint or design for the objects and they are going to tell the interpreter how to run the program.

How to create a class?

Creating one of these classes in Python is not something that is complicated. When you are working on one of these class statements, you are going to work on creating a new definition for this as well. You should lace the name of the class right after the keyword and then the superclass will be inside the parenthesis. Make sure that you follow it all with a colon to keep up with good coding practices. A good example of how this is going to work with the right syntax:

```
class Vehicle(object):
#constructor
def_init_(self, steering, wheels, clutch, breaks, gears):
self._steering = steering
self._wheels = wheels
self._clutch = clutch
self._breaks =breaks
self._gears  = gears
#destructor
def_del_(self):
     print("This is destructor….")

#member functions or methods
def Display_Vehicle(self):
   print('Steering:' , self._steering)
   print('Wheels:', self._wheels)
   print('Clutch:', self._clutch)
   print('Breaks:', self._breaks)
   print('Gears:', self._gears)
#instantiate a vehicle option
myGenericVehicle = Vehicle('Power Steering', 4, 'Super Clutch', 'Disk Breaks',
```

5)

myGenericVehicle.Display_Vehicle()

The output that you are going to be able to get from putting all of this information into your interpreter includes:

(‘Steering:’, ‘Power Steering’)
(‘Wheels:’, 4)
(‘Clutch:’. ‘Super Clutch’)
(‘Breaks:’, ‘Disk Breaks’)
(‘Gears:’, 5)

When you look through this example you are going to notice that there are a lot of the parts. It had the object definition, the methods definition, the attributes, the destructor function, the class definition, and the function. Let’s take a look at how each of these work and why they are so important within your code.

Class definition and object instantiation

You are going to need both of these parts as your syntax. They are going to help tell the code what you want to have happen. Here are some of the syntaxes that you are going to need in order to make sure that the interpreter recognizes what you are doing.

Class definition syntax will be the following:

```
class subclass[(superclass0]:
        [attributes and methods]
```

The object instantiation syntax will look like the following

object = class()

When you want to invoke the methods and attributes within the syntax you will use the following options:

object.attribute
object.method()

Special attributes that you can use in Python

There are some attributes that are special inside of Python. These are good to know about because they will make it easier to work on the code and you will be able to use them with the knowledge that the interpreter has an idea of how to read them. Some of the attributes that you should learn how to use within some of your codes include:

__dict___ this is the dict variable of a class name space
__doc__ this is the document reference string of class
__name___ this will be the class name
__module__ this is the module name and consists of the class
__bases__ this is the tuple that will also contain all of the superclasses

Let's take a look at how this may work within one of the syntaxes that you may choose to work with.

```
class Cat(object):
        itsWeight = 0
        itsAge = 0
```

```
    itsName = “”
    defMeow(self):
    print(“Meow!”)

    defDisplayCat(self):
    print(“I am a Cat Object, My name is”, self.itsName)
    print(“My age is”, self.itsAge)
    print(“My weight is”, self.itsWeight)

frisky = Cat()
frisky.itsAge = 10
frisky.itsName = “Frisky”
frisky.DisplayCat()
frisky.Meow()
```

When you are using this as your syntax in the interpreter the result that you will get on the screen is:

```
(‘I am a Cat Object, My name is’, ‘Frisky’)
(‘My age is’, 10)
(‘My weight is’, 0)
Meow!
```

Accessing the members of your class

With the syntax that we used above, we defined the object of the cat as being Frisky by using the dot operator in order to access the members of that object. This means that in order to get the right age assigned to Frisky, you will need to write out the function like frisky.itsAge=10. You can do this with all of the different objects that you want to assign.

With the example that we used above, there were a lot of variables that you are able to access from the class object. But this is not always the most convenient way to go out and get all of the variables. There are some other methods that you are going to be able to use in order to get the members of a class to come back out. The accessor method is one of the best because it is going to be able to provide the information or encapsulate it into the syntax.

```
class Cat(object)
    itsAge = None
    itsWeight = None
    itsName = None
    #set accessor function use to assign values to the fields or member vars
    def setItsAge(self, itsAge):
    self.itsAge = itsAge

    def setItsWeight(self, itsWeight):
    self.itsWeight = itsWeight

    def setItsName(self, itsName):
    self.itsName =itsName

    #get accessor function use to return the values from a field
    def getItsAge(self):
    return self.itsAge
    def getItsWeight(self):
    return self.itsWeight

    def getItsName(self):
    return self.itsName
```

objFrisky = Cat()
objFrisky.setItsAge(5)
objFrisky.setItsWeight(10)
objFrisky.setItsName(“Frisky”)
print(“Cats Name is:”, objFrisky.getItsname())
print(“Its age is:”, objFrisky.getItsAge())
print(“Its weight is:”, objFrisky.getItsName())

The output that you are going to get from all of this will be the following:

(‘Cats Name is:’, ‘Frisky’)
(Its age is:’, 5)
(‘Its weight is:’, 10)

With the example above, you are putting the accessor method to work in order to get the variables that you want to work. This is often a process that you are going to use for data encapsulation or data hiding. Encapsulation will be done if the variables end up being private or protected, but since you are working with the Python language, all the variable members will be default into the public.

Keep in mind that you are not going to find words like public, protected, or private when looking at the accessibility of your members. These are not used in Python because all of the members are going to be considered public by default. All of your attributes are going to be public so keep this in mind when getting started.

Property

When you are talking about properties in Python, you are talking about the part

that gets and then sets a value. It is going to work similar to some of the other methods in Python, but you will find that the syntax is easier to learn and use compared to some of the others. A property is going to be assigned just like you would assign a variable, and this will cause the setter method to be executed.

With the cat class that you used above, you already have the setter and getter inside the syntax. The setter property will be able to assign the value to the member variables while the getter property is going to return a value from the variables of the member.

Static Methods and Instances

When you use the static method, you are able to call out things without having to create a new object or class or using the new class names. This is because this kind of method is not going to accept a self instance.

But when you are using the instance method, you are going to need to use an object and a class and in order to use these, you will need to create both the class and the object before you are able to call it out. These are two different methods of calling things out and you will need to determine which one you want to use based on whether you want to create a new object when working on your syntax.

Objects and classes are really important when it comes to creating your syntaxes. The objects are going to function in the role of defining what you are using inside the statement. You are able to define them in any way that you want, but make sure that they make sense and that the interpreter is able to figure out what you want it to do based on your objects.

If you are curious at how these work, you should try out a few of the examples that we have in this chapter. Try out the original options first and then mix and

match them to get some great results.

Chapter 6: What are Exceptions and How Do They Work in Python?

As you get to working on the Python programming language, you will notice there are times when exceptions are going to happen. Depending on the type of exception, you could get an error message on the computer. Any time that abnormal conditions occur inside of your code, you will need to use these exceptions in order to tell the interpreter how it should react to these conditions. There are also times when the statements and inputs work out just fine within your language, but because of the program that you are trying to design, you will need to add in an exception that is just for that program. This chapter will take some time to look at how the Python program is going to work with exceptions so that the code works the way that you would like.

Any time that you are working on your code and you would like to show that a condition that is abnormal occurs inside the code, it is time to use the exceptions. There are a few of these kinds of conditions that the code won't allow. For example, if you are writing out your code, you may find that you put the wrong statement inside or you don't spell it out correctly; these could create an abnormal condition because your Python interpreter is not able to find what you are trying to do and an error message may come up.

In addition to having abnormal conditions occur with the code, there are times when a programmer will be able to add in some of their own exceptions inside of the code. In these cases, the interpreter will look at the input that the user places inside and will see that there is nothing wrong, but because of how your code is written, you want to make that exception show up. For example, if you have someone who is using your code who is 17 and you don't want to let people who are under 18 to place something into the code, you could make this into an exception, telling the code how you want it to behave in this situation even

though this would be something that the program would allow.

When you look through the Python program, you can see that the library has a few exceptions that are already in place. When you want to add in a few of these exceptions into your code, it is helpful to have a few of them inside the library to save you time and to show you how they are going to work. There are a few different types of exceptions that are inside the Python library, including where the program is going to have issues when you try to divide a number by zero (which is going to show an error message when you try to do this), or when you are trying to read past the end of your file.

The nice thing about exceptions in Python is that even when one of these errors is going on, you can make some changes to make it work out better. For example, instead of having the computer proclaim that an error is going on when you try to divide by zero, you could change this to give a message that shares why this is not allowed. Instead of stating that there is an error, you could make a message show up that says something like “you are trying to divide by zero! This is not allowed.” This tells the user why they aren’t allowed to do something inside of the program rather than just hoping they can figure out what the error message is all about.

If there are some exceptions that you would like to have done inside your code that aren’t found inside of the Python library, this is an option as well. You will learn how to raise some of your own so that an error, preferably with a message that you set up, will come on the screen and will let the user know that you aren’t allowing that answer for some reason.

When you are working on exceptions inside of Python, there are a few options that you will find inside of the library that you should look over and learn how to use. They are going to make exception handling easier and you will really need

to understand and use them properly to get the exceptions to work, whether they are exceptions set out by the program or ones that you are creating. Some of the statements that you will use inside of Python for exception handling in your codes includes:

- Finally—this is the action that you will want to use to perform cleanup actions, whether the exceptions occur or not.
- Assert—this condition is going to trigger the exception inside of the code
- Raise—the raise command is going to trigger an exception manually inside of the code.
- Try/except—this is when you want to try out a block of code and then it is recovered thanks to the exceptions that either you or the Python code raised.

Right now, these may not make much sense for how you are going to use them to handle exceptions, but we are going to take some time to talk about each one and discuss how it will work for your particular needs inside the code.

Raising exceptions

Raising an exception is an easy concept. When you are working in your code and you notice that there is an issue with it, or the program is trying to do something that won't work within the confines of Python, your Python program will raise an exception for this kind of conduct. This is due to the fact that the program will not understand how it is supposed to handle these issues. Sometimes, you will find that the exception is simple because you may have named something the wrong way and you just need to change that within the code. Or there cold be an issue with the user trying to divide something by zero, which isn't allowed within the system. A good example of how this is going to

look includes:

```
x = 10
y = 10
result = x/y #trying to divide by zero
print(result)
```

The output that you are going to get when you try to get the interpreter to go through this code would be:

```
>>>
Traceback (most recent call last):
    File “D: \Python34\tt.py”, line 3, in <module>
    result = x/y
ZeroDivisionError: division by zero
>>>
```

In this example, the Python program is raising up the error since you are taking a number and trying to divide it by zero. This is something that the Python language so you are going to see the error come up. As we mentioned before, when this error comes up, it is going to look a bit confusing to the user. They may not understand why there is an error coming up, so you will be able to make some changes to have the code look the way that you want. Having a new message show up on the screen is going to make it easier to see will seem a little friendlier than the other thing.

Making

These messages are going come up on the screen any time that the user puts in something that may not be allowed inside the system. These can be a bit nicer compared to having the error come up and if you create the message to be the right way, you will find that it can actually help out the user. With the message,

the user will be able to see the things that they did wrong inside the code and they can go through and make some of the changes to get it all to work out the right way. Here is something that you could write out in the code in order to get a message to come up, rather than just the error from before:

```
x = 10
y = 0
result = 0
try:
        result = x/y
        print(result)
except ZeroDivisionError:
        print("You are trying to divide by zero.")
```

If you look closely at this syntax, you will see that this is going to be pretty similar to what you found in the previous example, but there is one change that is inside. The program will still see that you are handling an exception, but you will be able to send out a message to the user letting them know that something is wrong in the code that they are writing. Instead of following the top example and having the error message come up, you will have the message come up that says (in this example) "You are trying to divide by zero." This is going to leave a look that is much cleaner and will give the user a chance to go back on and change the input they are doing.

Making your own exceptions

In the example above, we took a look at how to handle the exceptions that come up if your user is doing something that the Python system doesn't allow. In those examples, we took a look at what happens when the user tries to divide by zero within the program. But there are some steps that you can take any time that you

would like to make an exception into the program, even if it is something that Python does allow. There are some programs that you can create where you may not want to allow a certain answer or so on, and making sure that this is handled as an exception in the program will ensure that it is going to work the way that you want.

For example, if you want to make sure that the user isn't allowed to place certain numbers into the system, you would want to make sure that there is an exception added in. If you want to make sure that the user is able to add in two or more guesses for the part they are working on, you could also create an exception for that as well. The trick here is that the Python program wouldn't see something wrong with what the user is putting into the system, but because of the way that you design your program, there are going to be issues if you allow everything. A good example of this is dealing with ages. There is nothing wrong with the person stating that they are 20, but if you are on a gambling site that only allows people 21 and older to play, you will want to make an exception that says that anyone under the age of 21 is not able to get onto the site.

You are able to set up any exception class as long as it meets with the Python rules and also follows the rules of the program that you are trying to create. If there is a certain thing that you don't want to happen inside of the code, such as having certain answers, letters, numbers, and other things picked, you need to add in these as an exception to the program. Here is an example that you can follow that will help you to create some of these exceptions:

```
class CustomException(Exception):
def_init_(self, value):
        self.parameter = value
def_str_(self):
        return repr(self.parameter)
```

```
try:
        raise CustomException(“This is a CustomError!”)
except CustomException as ex:
        print(“Caught:”, ex.parameter)
```

Using this particular syntax is going to help you to get the message “Caught: This is a CustomError!” and it is going to show up whenever the user places in some of the wrong information, based on the conditions that you set out. This is a great way to show that there is a custom exception that is going on inside the program, especially when this is an exception that you set up by yourself rather than one that is through the program.

Keeping this in mind, you are able to make any changes to the wording that you would like to use inside this as well. You aren’t stuck with just leaving the message that we have listed above. You can use this as a good starting point, or choose to change it up so that it works for your particular program. If you are keeping people out of the program because it is a gambling site and you can’t allow anyone under 21 to come into the game, for example, then you may want to leave a message about that in there.

When you are working with these exceptions, you will find that they are going to be defined as being able to do the same things that most of your classes are able to do, but it is best to keep these as simple as possible when you are the programmer, changing a few of the attributes to make sure that the right error is extracted out of the exception, but not adding in much more than that or it can get confusing.

It is possible to create some modules inside of your program or your code that will bring up two or more errors all at the same time. The best way to do this is

to create one class that will be the base for which the definition of the exemptions will be listed out. This is going to help sort out the exceptions that you want to work with. you can then go through and create some subclasses that will take care of all these different classes.

Doing some cleanup actions

Another of the clauses that you would like to use is the ones that are considered the try statement. This is one that is also known as the finally clause, because you are going to use this in order to clean up the actions that are going through your program. These are often called the clean up actions because once you go through and execute them, they can clean up the code, make sure that it is ready for the next part, and make everything run in a smooth manner.

This is one of the clauses that will be used both on the way in with the code and then again on the way out. You can choose it to clean up on the way out whenever the other clauses are still left behind in the statement, return statement, continue, or a break. To see how this is going to work, take a look at this example for clean up actions inside of your Python statements:

```
def divide(x, y):
        try:
        result = x/y
except ZeroDivisionError:
print("Division by zero!")

else:
        print("result is", result)
finally:
```

print(“Executing finally clause”

divide(2,0)

When you go through and execute this syntax, you will see the results:

>>>
Division by zero!
Executing finally clause
>>>

This is one of the useful clauses to know how to use because it is great for ensuring that all of the statements will clean up and that when you are done with the code, nothing is going to be left behind. It can be really helpful any time that you are worried about how your program will end or there is some kind of issue between switching from one part of the code to the other. It is also often used when you are dealing with exceptions and you need to allow the program to move on, even if the exception has been brought up. You will need to experiment w a bit with these kinds of clean up actions because they will make it possible to have smooth transitions between your code and will make it work better than before.

Working with exceptions can be a lot of fun inside of Python. They will add in some new stuff that you will be able to do, such as helping to bring up messages instead of error signs when working with the exceptions that are already inside of Python and watching to see which of the exceptions you are going to bring up all on your own to work with the program. You may find that as you go along, there are going to be more and more instances when you want to bring up exceptions and make your program more powerful while getting it to work the exact way that you want.

Chapter 7: Working with Conditional Statements in Python

Another thing that you are able to work with inside of the Python language is that you are able to teach the computer to make some big decisions for you. There are times when you will set up your conditions into the program and then the user will put in their input. At times these answers are going to be true based on the conditions that you set and other times they will put in some information that is going to false based on your conditions. You will need to set up the computer so that it can decide what it should do based on what you have put into the system and the answer that comes up.

This chapter is going to show you some of the basics that come from teaching the program you are making to do its own decision making. This will add some more power to the programs and makes it easy to work inside of your program while having a few different options come up on the screen.

First, let's take a look at the different types of decision control commands that you can make inside of your program. Some of the most common that are found inside of Python include the switch statements, the if statements, and the if…else statements. There are also variations of these that you can work with that allow for more options inside of the code.

As a beginner, we are going to start out with the most basic of these decision control commands. This is the if statement and it is going to work on the basis of true or false. You will set out the conditions that are true inside of your code. Then the user is able to put in any information that they would like. If the program determines that their answer is true based on your conditions, your message will show up on the screen or they will be able to proceed through the program some more. On the other hand, if the input of the user is determined to

be false, the system is going to just shut down and stop (we will take a look at how you can leave a separate message for when the answer comes out true later on in this chapter).

This is pretty basic and allows the system to just put up one message based on whether the input of user is true based on your conditions. A good example of these if statements would include the following syntax:

```
age = int(input("Enter your age:"))
if (age <=18):
        print("You are not eligible for voting, try next election!")
print("Program ends")
```

So when we take a look at this syntax, there are a few points to watch out for. If your user comes onto the site and puts that their age is 18 or under, they have matched true to the conditions that you have set. The program will see that the input is true and will provide them with the statement "You are not eligible for voting, try next election!" on the screen of your user. But something a bit different is going to happen whenever the user places an answer that is above 18, such as 25 into the program.

When the user puts in an answer such as 25 into the program, the compiler is going to read that this input is considered false based on the conditions that you have set. Since this is a basic formula for coding, there is nothing set up for when the answer is false. The compiler is not going to put out the statement above because that only shows up when the conditions are considered true, but when the input is considered false, the compiler has no instructions with the if statement on what to do in this situation. For now, the compiler is just going to leave a blanks screen and will not continue on with the program.

As you can see, this is not the most efficient method that you can use with making decisions inside of your program. While it can help you to just get the true answers that come in from your input, and it is a good place to get started when you are trying to write out some of your first codes, what are you going to do when someone enters an answer that is not considered true based on your conditions? Do you really want someone to come to your code and then get a blank screen, or the program shutting down, because their age or some other information doesn't line up with your conditions, even if the information is right? Your user may be older than 18 and it doesn't make sense to close down the program because it doesn't meet with your conditions.

Luckily, there are some methods that you are able to use that will make sure that one message is going to show up when the conditions are met and then another message can show up when the conditions are not met. This ensures that the user is able to see some kind of statement or message regardless of the answer that you put up. These are going to be the if…else statements.

The if…else statement is basically going to be in charge of making some exceptions to the rules to help the program keep on running. You can just add in a small little part in order to get the program to work well with the if…else statements and to ensure that the right messages are going to show up regardless of what answer they place into the system. A good syntax for working with the if…else statement includes:

```
age = int(input("Enter your age:"))
if (age <=18):
    print("You are not eligible for voting, try next election!")
else
    print("Congratulations! You are eligible to vote. Check out your local polling station to find out more information!)
```

print(“Program ends”)

now with this example, there are going to be two options that will come up on your system. If your user inputs that they are 18 or a younger age, the first statement “You are not eligible for voting, try next election!” will come up on the screen. On the other hand, if your user inputs an age that is above 19, the second message the “Congratulations! You are eligible to vote. Check out your local polling station to find out more information!” will come up on your screen. This is a good way to allow the user to input any age into the system that they would like and then they will get the corresponding message that tells them what is going on.

Now this is a pretty basic version of using the if…else statement. You can add in more than one option if you would want. If you want to make it so that you have three or four different age groups that people are separated out into, you can make this happen as well. You would just need to add in another else portion as well as the information that will tell the program what message or statement that will come up based on the input of your user.

Once you have had some time to get more familiar with the if statements, it is time to move over to the if…else statements. These will add in some more options to the whole program and will ensure that your program looks all put together and like it is working properly. No one wants to go to a program, put in an answer, and then find out that the screen goes blank, but this is something that can happen when you are working with the if statements because they are too simple to handle some of the different variations that happen in coding. Luckily, you are able to use the if…else statement is able to fix this process and even as a beginner you will be able to use it to make a great looking program.

Working with elif statements

In addition to working with the if statement and the if…else statement, you can also create some elif statements. These are nice because they are going to give the user some choices that they are able to make. Each of the choices will have a process associated with it and when the user makes a decision, these processes will occur that correspond with the choice. This is one of those options that are popular inside of games that let the user choose what they would like to do inside the game. It is possible to add in as many of the elif statements as you would like into the code, you just need to make sure that you have the right syntax in place and that the functions are going to work out the best. To see how the elif statements are going to look in their most basic format, take a look at the following:

if expression1:
statement(s)
elif expression2:
statement(s)
elif expression3:
statement(s)
else:
statement(s)

This is the basic syntax that you are able to use when working inside the elif statements and you will simply be able to place whatever information you would like into it and set up the answer that you would like to work with each of the parts. As mentioned, this is one of the basic forms of the elif statement. You will be able to make some of the changes that are needed into the program in order to get many different choices for the user to enjoy and you can choose which conditions need to be met as well as what responses you would like the come up

based on the answers that the user is providing to you. So let's take a look at how this would look when it is used in a code and has been expanded out a bit more.

```
Print("Let's enjoy a Pizza! Ok, let's go inside Pizzahut!")
print("Waiter, Please select Pizza of your choice from the menu")
pizzachoice = int(input("Please enter your choice of Pizza:"))
if pizzachoice == 1:
    print('I want to enjoy a pizza napoletana')
elif pizzachoice == 2:
    print('I want to enjoy a pizza rustica')
elif pizzachoice == 3:
    print('I want to enjoy a pizza capricciosa')
else:
    print("Sorry, I do not want any of the listed pizza's, please bring a Coca Cola for me.")
```

This code is pretty simple for you to get started with, but let's look at what it is all going to stand for and figure out how the user would be able to do it. This syntax is listing out a few options that the user is able to go with and they can pick the one that they like. For example, the user can look at this code and pick out the number 2 in order to tell the system that they wanted the pizza rustica. If they would rather stick with a Coca Cola rather than having any pizza at all, they would be able to pick that choice as well. You would be able to add in as many of these choices as you would like to go with the elif statements. If you would like to have ten options because this goes with the program you are creating, that is fine, it would just expand out in the same method as you see in this section.

The if statements are some of the best commands that you can learn how to work with inside of the Python programming language. These statements make it easier for the user to make some decisions in what it would like to do and you

can place in the different options that you would like to be available for them based on their answers. You can start out with a basic if statement and just have one answer be the right one, with nothing showing up if the user input ends up being false, or you can make it a bit more complex by adding in some options based on whether the conditions are met or not. And then the elif statements come into effect and allow the user to pick from numbered options to get what they want.

The if statements will add in a lot of power to the program that you are working with. They are a complex part of the Python system, but even a beginner is able to learn how to work with them and all the options that they will allow inside of your code will be amazing. Try out a few of the syntaxes and codes that we have available in this chapter and see how it feels to write out these statements in the command prompt as well as learning how to use these statements to write some of your own code.

Chapter 8: Learning How Inheritances Work in Python

The next thing that we will learn how to work with inside of Python are inheritances. These are really great and were an addition that was added to the OOP languages. You will be able to make your code look a lot cleaner and nicer because you will not have to rewrite all of it over again. Inheritance is going to allow you to go through and copy the important parts of one code and translate them into a new part of the code, adding in the things that you want.

You are able to use inheritances to make things easier, to help keep things together, and you can add in a few inheritances together to ensure that you are getting all the parts together for a new part of the code. Here is a good example of what you may see when you are working with inheritances in your code on Python.

```
#Example of inheritance
#base class
class Student(object):
        def__init__(self, name, rollno):
        self.name = name
        self.rollno = rollno
#Graduate class inherits or derived from Student class
class GraduateStudent(Student):
        def__init__(self, name, rollno, graduate):
        Student__init__(self, name, rollno)
        self.graduate = graduate

def DisplayGraduateStudent(self):
        print”Student Name:”, self.name)
```

```
        print("Student Rollno:", self.rollno)
        print("Study Group:", self.graduate)

#Post Graduate class inherits from Student class
class PostGraduate(Student):
        def__init__(self, name, rollno, postgrad):
        Student__init__(self, name, rollno)
        self.postgrad = postgrad

        def DisplayPostGraduateStudent(self):
        print("Student Name:", self.name)
        print("Student Rollno:", self.rollno)
        print("Study Group:", self.postgrad)

#instantiate from Graduate and PostGraduate classes
        objGradStudent = GraduateStudent("Mainu", 1, "MS-Mathematics")
        objPostGradStudent = PostGraduate("Shainu", 2, "MS-CS")
        objPostGradStudent.DisplayPostGraduateStudent()
```

When you type this into your interpreter, you are going to get the results:

```
('Student Name:', 'Mainu')
('Student Rollno:', 1)
('Student Group:', 'MSC-Mathematics')
('Student Name:', 'Shainu')
('Student Rollno:', 2)
('Student Group:', 'MSC-CS')
```

Overriding your base class

There may be times when you will need to override the bass class so that you are able to take it into the new derived class. The basic meaning of doing this process is that you are going to replace the parental behavior so that it can now be present inside of your child class.

You will be able to use the parental features that you want and then add features and make changes inside of the child class, making the new class work the way that you want. With this method, it is easy to copy over another class, avoid duplicating some of the same codes, and still keeping the ability to enhance or customize the code that you want.

Overloading

Another task that you may want to do when working on these inheritances is overloading. When you are overloading you basically just use one identifier in order to define two or more methods inside of a class that differ in the input as well as the output parameters. The overload methods are going to often be used when they will execute the same task conceptually, but there are some differences in the parameters that you set. Overloading is also going to give you the ability to define two or more functions with the same names, but which have different signatures.

In most cases, you are not going to need to use this kind of function. Since Python is considered a dynamically typed program, it is easy to support optional arguments inside the function, so you won't usually have to worry about using the overloading process.

If you do have a reason to use the overloading process, you will need to make sure that you download the right module to help you get this done. A good place to go to get a good module to finish this up is

https://github.com/bintoro/overloading.py.

Method Overloading

In one of the newer versions of Python, there is a new feature that has been added that will make it easier to do method overloads. This is going to make it possible to perform different operations depending on what type of statement you are using in your first argument.

When you are making a generic function, you will have one that is composed of different functions that are implementing the same operation over different types. Which type of implementation you are choosing or which one you should use during this time is often determined by the dispatch algorithm. When the program chooses the implementation from just one of your arguments that is on its own, this is called a single dispatch.

To define this generic function, you will need to use the right decorator. This is usually the @singledispatch decorator. Keep in mind that this dispatch is going to happen just with the first argument.

Multiple inheritance

Another feature that you may want to work with inside of Python is multiple inheritance. We have already discussed how to go through and do one inheritance and how you will be able to use the features of the parent code to work with the child code while also making the corrections that are needed. But you can take this a step further and do multiple inheritance.

The multiple inheritance is a feature where a class is going to have two or more parent classes. As long as you do this properly, you will be able to use as many

parent classes as you would like within the new child class. What basically happens is that you are creating a new class, ClassC, and it is created from the features of ClassB. ClassB was designed from ClassA, meaning that ClassC is going to have some features from both ClassB and ClassA to make it work. You will be able to make adjustments at each level, but you will get some of the features of the parent class into the child one.

Keep in mind that when you are working with multiple inheritances, you will not be able to do circular inheritance with Python. But it is possible to use two or more parent classes in order to create the new child code that you would like to use in Python.

There are many times when you are going to want to use inheritances inside of your code because it is easier. You will be able to use some of the methods and features from codes that you already wrote and then put them into a new one, adding in some extra features and making changes as you would like. It is even possible to use a few different method parents to make new ones if this will work the best inside of your code.

There are many options that you are able to do with inheritances and these will make your code look amazing if you do it right. You can just keep carrying the inheritance down in the code as much as you would like, making changes each time or keeping them pretty much the same. It will all depend on the things that you want to happen in your code.

Chapter 9: Creating Loops in Python

While the decision control statements add some good power to the code and can help you to do a lot more than before, it is definitely a good idea to work with the loops inside of the Python program. These are helpful any time that you need the program to repeat something inside of the code, but you don't want to write the code out over and over again. For example, if you would like the code to list all the numbers from 1 to 10, it probably wouldn't be much fun to write out the code to tell the compiler to write out 1 and then write it again for 2, and so on until you reach 10. You could use the idea of looping in Python to get this all done in one block of code to save time and to make the code easier to read.

Loops are pretty easy to work with. They are basically going to tell the compiler to keep on reaidng the same block of code over and over until some condition is met. If you would like the code to count from 1 to 10, you would just need to tell the code to stop going once it got higher than 10 (we will look at some codes that show how this is done). You do need to be careful with this though because if you don't set out the condition ahead of time, the code is going to end up in a continuous loop and will get stuck. So always make sure that you set out the break of the code, or the part that will make the loop stop and move on to another part of the code to read.

There are several types of loops that you can work with to help make the program work the way that you would like. The two we are going to talk about in this guidebook are the while loop and the for loop.

What is the while loop?

One of the loop types that you are able to work on is known as the while loop.

This is the one that you will want to use whenever the code should go through the cycle a fixed amount of times. You don't want this code to go through the cycle indefinitely, but you want to make sure that it goes through the right amount of times. Often this is the one that will repeat at least one time before it sees whether the code is true or false so it can be helpful if you would like to use it in this manner. To understand better how this is going to work, let's look at a good example of the while loop:

```
#calculation of simple interest. Ask user to input principal, rate of interest, number of years.

counter = 1
while(counter <= 3):
    principal = int(input("Enter the principal amount:"))
    numberofyeras = int(input("Enter the number of years:"))
    rateofinterest = float(input("Enter the rate of interest:"))
    simpleinterest = principal * numberofyears * rateofinterest/100
    print("Simple interest = %.2f" %simpleinterest)
    #increase the counter by 1
    counter = counter + 1
    print("You have calculated simple interest for 3 time!")
```

Once you have the time to put this in your compiler and execute the code, you will see that the output is going to come out so that your user is able to place in the information, any information, that they wish to have computed. The program is going to be able to figure out the rates of interest as well as the final amount based on the numbers that the user places into the system. Right now this loop is set up to go through three times, but you are able to change it around and get it to do more loops if you would like.

How is the for loop different?

There are many times that the while loop will be your friend and can help you to get a lot of things done, but there are other times when the for loop is useful. The for loop is considered a more traditional way to write out the loop, but there are many times when you will want to use it.

In this kind of loop, the user won't be able to go in and give the right information or determine when the loop is able to stop. Rather, with the for loop, Python will go over the iteration in the order that they show up inside of your statement and it will show this information on the screen, without needing input from someone else, until it reaches the end. A good example of how this may work includes:

```
# Measure some strings:
words = ['apple', 'mango', 'banana', 'orange']
for w in words:
print(w, len(w))
```

Now, when working on the example above, you will be able to put it into your code and when it is executed, the four fruits will come out onto the screen, in the exact order that they are already written. Any time that you would like to have them come out on the screen in a slightly different order, you will just need to go back to the syntax and change that up. Once they are in the syntax and ready to be used inside of the code, you will not be able to make these changes.

The nested loop

The final type of loop that you may find helpful to use inside of your Python code is the nested loop. When you are using a nested loop, you are basically

taking one loop and placing it inside of another one and then allowing both of them to keep on running until they are done. There are several times when this can be useful to help you to get things done. A good example would be when you would like to write out a multiplication table that starts at 1 and goes all the way to 10. Here is an example of how that would work:

```
#write a multiplication table from 1 to 10
For x in xrange(1, 11):
    For y in xrange(1, 11):
    Print '%d = %d' % (x, y, x*x)
```

When you got the output of this program, it is going to look similar to this:

1*1 = 1
1*2 = 2
1*3 = 3
1*4 = 4
All the way up to 1*10 = 2
Then it would move on to do the table by twos such as this:
2*1 =2
2*2 = 4
And so on until you end up with 10*10 = 100 as your final spot in the sequence.

There are so many reasons why you would like to use a loop inside of your code. You will be able to use it as a way to clean up your code while getting the compiler to go through the same block of code over and over again. The compiler will continue to go through this code until the condition is no longer considered through and then it will move on to either end the program, or move on to the next part if there is more. This can open up a lot of things that you are able to do with your code while keeping things easy to use.

Chapter 10: The Importance of Those Operators

Inside of Python, it is common to run into operators that will help to make your code a bit stronger. There are many types of operators and they are responsible for helping you to do things such as mathematical equations, comparing different parts of the code, and even giving a value over to the variable. There are many of these that you are able to work with depending on what the code is trying to do so let's take a look at some of the operator types that you could use and see how they would work for you!

Arithmetic operators

The first type of operator that we will discuss is the arithmetic operator. These are the ones that would be able to help you do a mathematical equation, even soemthing as simple as adding together two oeprands, inside of the code. Some of the options that you have with the arithmetic operators includes:

- (+): this is the addition operator and it is responisble for adding together both of your values.
- (-): this is the substraction operator and it is going to be responsible for taking the right operand and subracting it from the left.
- (*): this is the multiplication operator and it is used to multiply two or more values in the equation.
- (/): this is the division operator and it is going to divide the value of the left operand from that on the right and gives you the answer.

You are able to use more than one of these at a time inside a statement of your code if that is needed. For example, you could add together three or more

numbers if you need to or you can add a few numbers and then subtract some others if it works for your code. The rule here is to remember that you need to multiply all the numbers first, then divide the ones that have this symbol before moving on to addition and subtraction in order to get the right answer.

Comparison operators

Another type of operator that you will be able to use is the comparison operator. This one is helpful if you would like to compare the two or more values, or even the statements, that are inside of your code. They are often used with Boolean expressions because they are going to return a true or false result; you will either have two numbers that are equal to each other or not equal to each other for example so this is a good way to figure this out. Some of the comparison operators that you may use in your coding include:

- (>=): this one means to check if the left hand operand is greater than or equal to the value of the one on the right.
- (<=): this one means to check if the value of the left hand operand is less than or equal to the one on the right.
- (>): this one means to check whether the values of the left side are greater than the value on the right side of the code.
- (<): this one means to check whether the values of the left side are less than the values that are on the right side.
- (!=): this is the not equal to operator.
- (==): this one is the equal to operator.

The Logical Operators

The logiacl operators are also going to work in order to help with evaluating the input that the user gives you with the conditions that you have set. There are

three main logical operators that you may be interested in using including the not, and, as well as the or and we are going to talk about them below:

Or: with this one, the compiler is going to evalue x and if it is false, it will then go over and evaluate y. If x ends up being true, the compiler is going to return the evaluation of x.
And: if x ends up being the one that is fase, the compiler is going to evaluate it. If x ends up being true, it will move on and evaluate y.
Not: if ends up being false, the compiler is going to return True. But if x ends up being true, the program will return false.

Assignment Operators

And finally, we will look at the basics of the assignment operator. This one is usually going to use up the equal (=) sign in order to assign some kind of value over to your variable. If you would like the variable to be equal to 100, that is how you would write it all out. At some point in the code writing process, you will need to include this in there to tell the compiler the value of the variable and the assignment operators are going to help to make this happen.

You can sometimes use this in order to assign more than one value to the same variable as long as you use the right signs. You would just need to use the same variable and make sur eto bring in the right assignment operator, and then the variable can hold on to as many values as you would like.

Working with operators can make your code a bit easier to handle. You will be able to add together variables or do other mathematical equations. You will be able to assign a value to your variables to help the compiler know what you are doing, and you will even be able to do some comparisons to help figure out how the system is supposed to work in your program. Take some time to write out a

few codes that use these operators and see just how simple it can be!

Chapter 11: Ready to Work with Some Projects?

Now that you know a few of the basics that come with the Python program, it is time to work on a few projects to make it even more interesting. This can help you to get more practice with what we have learned inside this book and ensures that you get some extra practice with writing out the code before getting started with your own code. In this final chapter, it is time to practice a few different codes that it is possible to try out, games and other guessing options, that you are able to try out with some of these Python codes. Let's take a look at all the fun things that you are able to do when writing in Python to help see some of the power that comes with this great code.

Creating your own magic 8 ball game

Think back to your childhood when you played with a magic 8 ball, asking it questions and seeing what answers you were able to get at the same time. It was a simple game, you had to be careful with some of the questions that you asked because there were only a few answers that you were able to get out of the system, but it could be a fun game. You are able to make this same kind of game using the codes that we have discussed in the Python coding language. The code that you would need to use to make your own magic eight ball inside Python includes:

```
# Import the modules
import sys
import random

ans = True
```

```
while ans:
    question = raw_input("Ask the magic 8 ball a question: (press enter to quit)")

answers = random.randint(1,8)

if question == ""
    sys.exit()

elif answers ==1:
    print("It is certain")

elif answers == 2:
    print("Outlook good")

elif answers == 3:
print("You may rely on it")

elif answers == 4:
    print("Ask again later")

elif answers == 5:
    print("Concentrate and ask again")

elif answers == 6:
    print("Reply hazy, try again.")

elif answers == 7:
    print("My reply is no")
```

elif answers == 8:

print(“My sources say no”)

Here we just provided you with 8 answers, but you can cut it down a few to make it easier or give twenty answers if that is what you would like. It is set up to be random and pick the answers differently each time, which can make it easier for you to get something different each time that you ask a new questions. This is a good way for you to use some of the topics that we discussed in this book and you can have some fun playing the game by asking it questions during the testing period.

Let the computer roll the dice

Another game that you are able to create is one where you tell the Python program to roll the dice for you. This one will also use the random idea that we did above because you want the computer to determine different numbers on the dice each time that you roll; no one wants to play the game and get the exact same numbers each time the do it so the random module will help to make sure that this happens. In this code, it is your responsibility to set up two different variables, the maximum and the minimum, so that the system will only pick out numbers that are on your code. For example, you will want to make sure that your minimum on this one is 1 and the maximum is 6. If you would like to make a special die, you could add in more numbers, but we will just keep it simple for this one.

You will notice in this example that we are working with the loop function. This allows the user to roll the dice as many times as they would like without restarting the code, rather than just doing it one time. We are going to use the “y” for yes so that the user is able to click on it and roll the dice again and again as they would like.

So, to create this new game, we would need to use the following code to tell the compiler what to do:

```
Import random
min = 1
max = 6

roll_again = “yes”

while roll_again == “yes” or roll_again == “y”:
        print(“Rolling the dice…”)
        print(“The values are…”)
         print random.randint(min, max)
        print random.randint(min, max)

roll_again = raw_input(“Roll the dices again?)
```

Creating a Hangman game

Here we are going to look at how to create the Hangman game with the help of your Python compiler. If you are a fan of this game, you are going to see that it is pretty easy to create this game all on your own. You will be able to use some dashes that are in a row in order to represent the word that you are trying to have the user guess and when the user does guess a letter that is inside the word, the script will be able to place this letter in the correct spot.

For the game that we are going to make, we will allow the player to guess 10 times. After the 10 times, or when the user guesses the word ahead of that many times, the game is going to end. You will be able to choose the variables and

make it a bit different, such as giving it the ability to let the user guess more or fewer times, but here we are going to keep things simple and stick with 10. The code that is needed in order to create this game includes:

```
# importing the time module
importing time

#welcoming the user
Name = raw_input(“What is your name?”)

print(“Hello, + name, “Time to play hangman!”)

print(“
“

#wait for 1 second
time.sleep(1)

print(“Start guessing…”)
time.sleep(.05)

#here we set the secret
word = “secret”

#creates a variable with an empty value
guesses = ‘ ‘

#determine the number of turns
turns = 10
```

```
#create a while loop

#check if the turns are more than zero
while turns > 0:

    #make a counter that starts with zero
    failed = 0

    #for every character in secret_word
    for car in word:
    #see if the character is in the players guess
    if char in guesses:
    #print then out the character
    print char,

    else

    # if not found, print a dash
    print "_",
    # and increase the failed counter with one
    failed += 1

    #if failed is equal to zero

    #print You Won
    if failed == 0:
    print("You Won")

    #exit the script
    Break
```

```
print

# ask the user go guess a character
guess = raw_input("guess a character:")

#set the players guess to guesses
guesses += guess

# if the guess is not found in the secret word
if guess not in word:

#turns counter decreases with 1 (now 9)
turns -= 1

#print wrong
print("Wrong")

# how many turns are left
Print("You have," + turns, 'more guesses')

#if the turns are equal to zero
if turns == 0

#print "You Loose"
print("You Loose")
```

As you can see, this is one that has some more complexities to it because of the nature of the game, but it has many of the different parts in it that you need to practice when using this code. You will be able to add in more parts, or take

them away and maybe try just having 5 practices or tries inside of it to make it easier, but you will be able to try it out and play a fun game with all of the information that you learn inside of this guidebook and the code that is above.

Making some of your own projects inside of Python is one of the best ways to work on learning the code. Many of these games bring together different parts of the information that we learned inside of this code so that you can see that they aren't just some random concepts that are talked about but you will never actually used. In fact, many of the options that we discuss are going to be used together inside the same code in order to make it functions. As you get used to seeing some more of these things together and you practice writing out codes like the one above, you will be better able to see how all of the parts need to be present in order for the code to work properly.

Conclusion

Thank you for taking the time to read my book! I hope you learned everything that you needed to know about Python coding and about getting started with the Python language.

If you liked what you read, please take a moment to leave a review after this book! Also, to keep up to date on my future books, check out my Facebook page at SAHM Writing!