

# MOOCS PRESENTATION

## Using Python to Interact with the Operating System

DEEPANKAR SHARMA

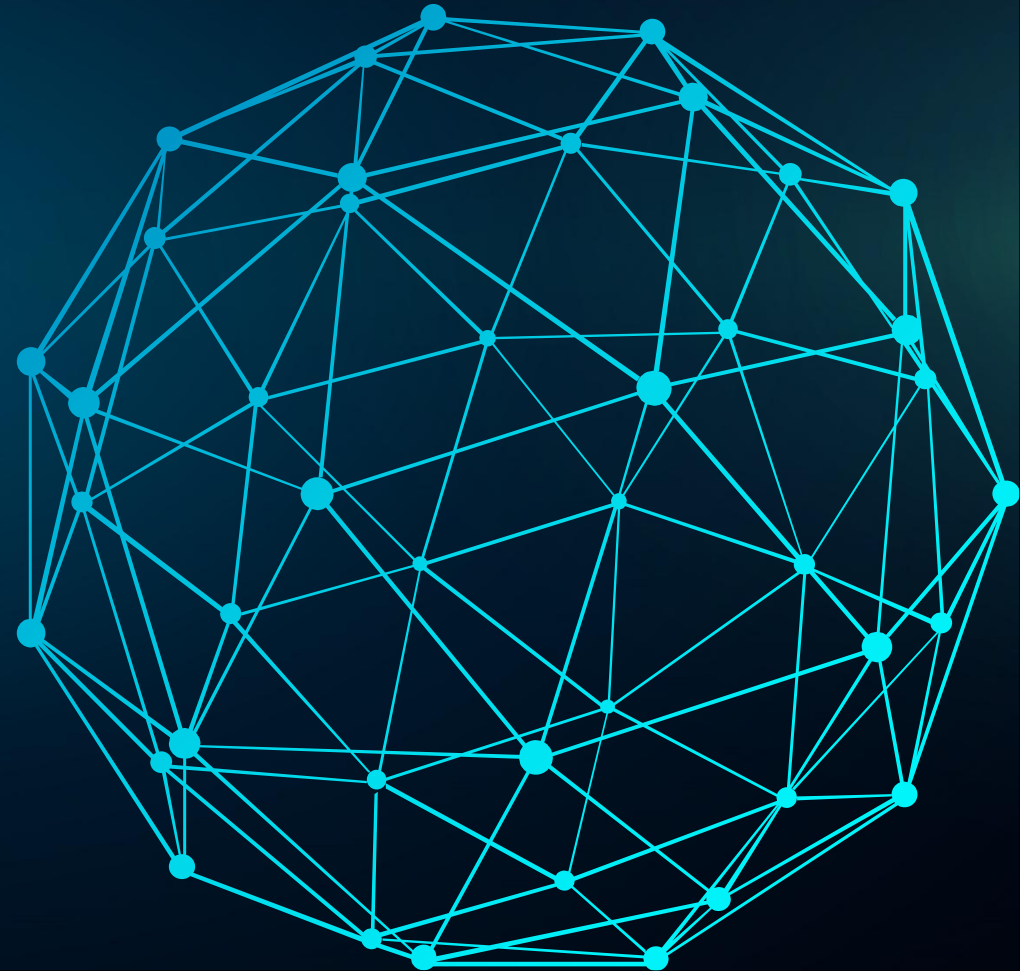
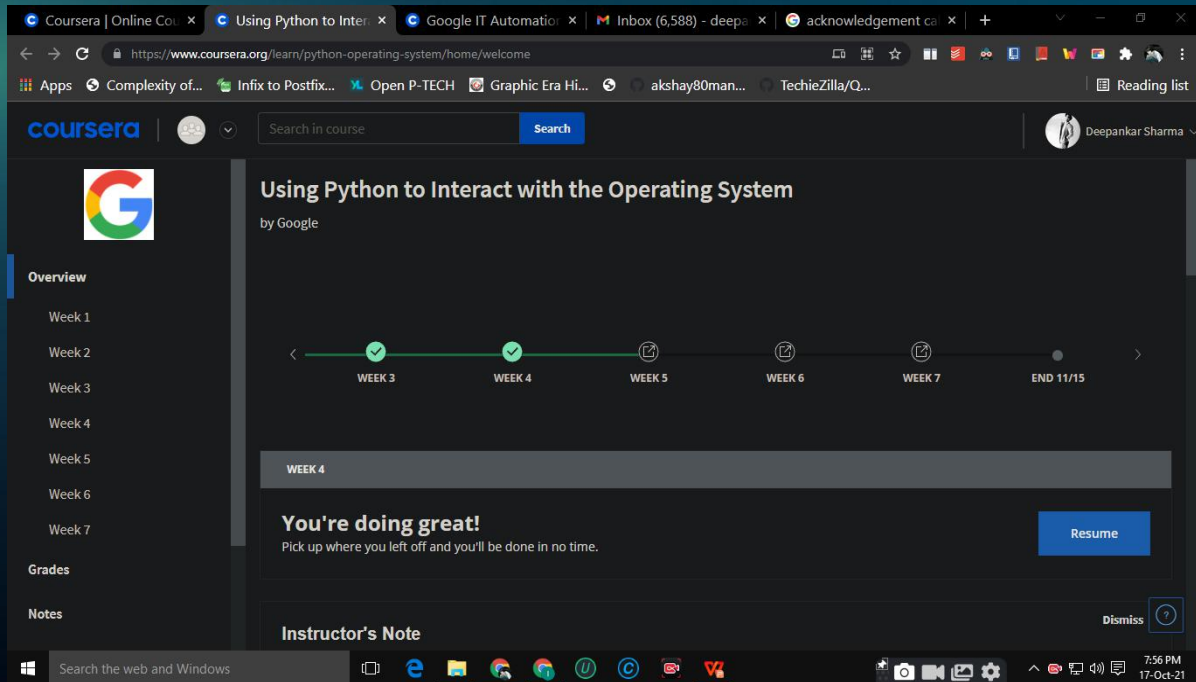
STUDENT ID : 20041299

UNIVERSITY ROLL NO : 2092014



# Google IT Automation with Python Professional Certificate

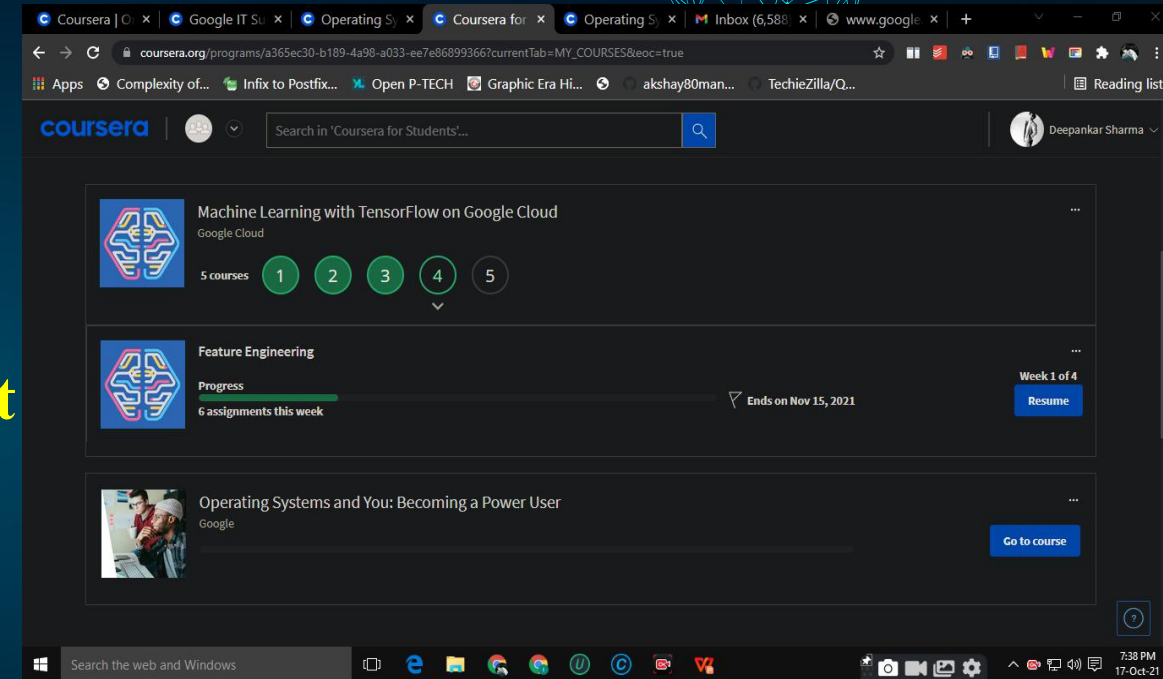
## Using Python to Interact with the Operating System



# Acknowledgement

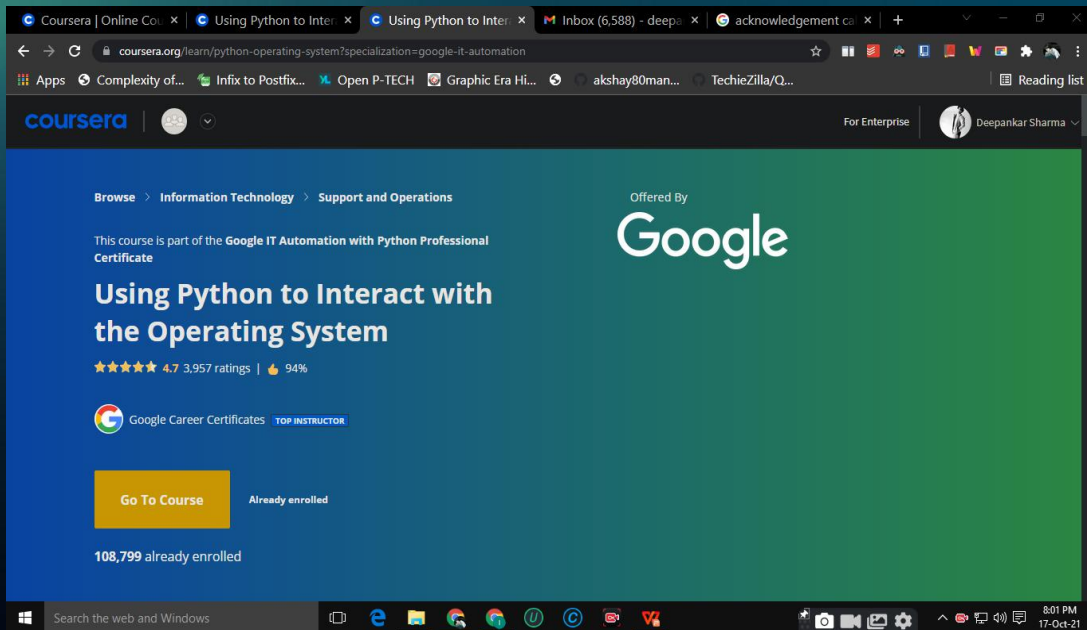
I would like to express my special thanks of gratitude to my teacher Mayurika Ma'am , who gave me the golden opportunity to do this wonderful moocs seminar on the topic “Using Python to Interact with the Operating System”. And also for providing with all the facilities that were required. The fact is I am learning lots of new skills during this course and I'll surely implement this knowledge in my real time problems too.

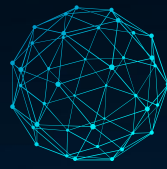
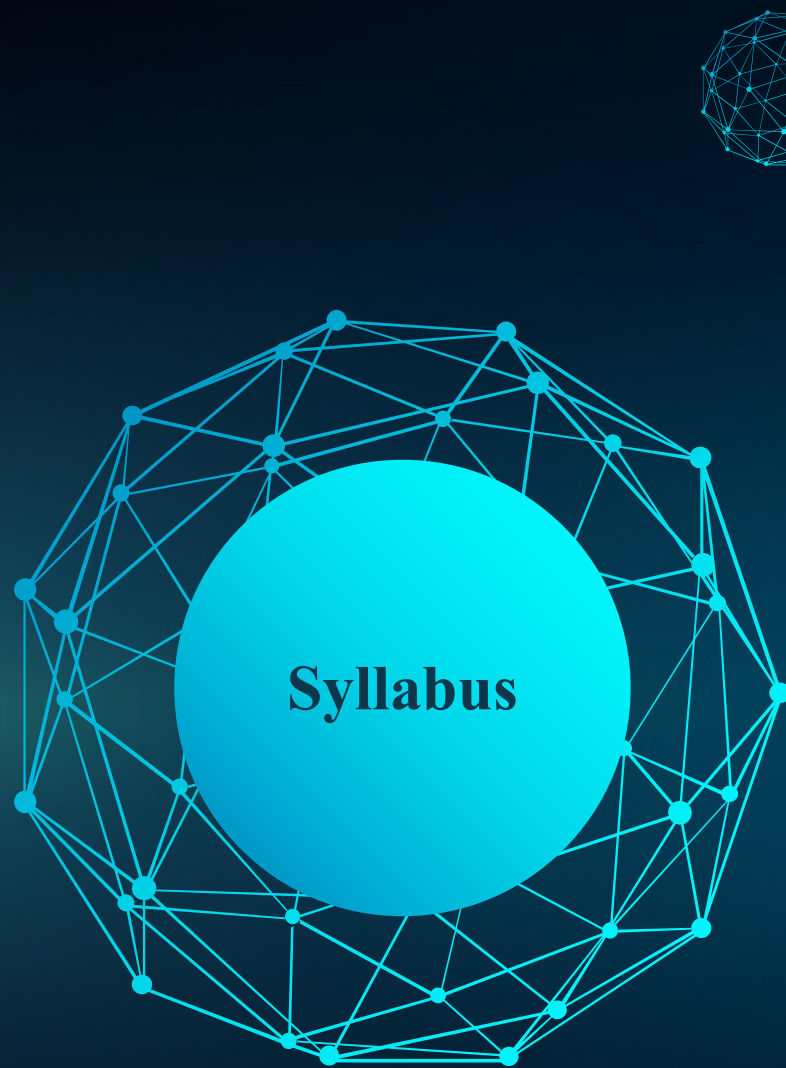
REGARDS,  
DEEPANKAR SHARMA





**By the end of this course, you'll be able to manipulate files and processes on your computer's operating system. You'll also have learned about regular expressions -- a very powerful tool for processing text files -- and you'll get practice using the Linux command line on a virtual machine. And, this might feel like a stretch right now, but you'll also write a program that processes a bunch of errors in an actual log file and then generates a summary file. That's a super useful skill for IT Specialists to know. We'll also explain how to set up your own developer environment in your machine. This is a key step in being able to write and deploy powerful automation tools.**





Getting Your Python On



Managing Files with Python



Regular Expressions



Managing Data and Processes



Testing in Python



Bash Scripting

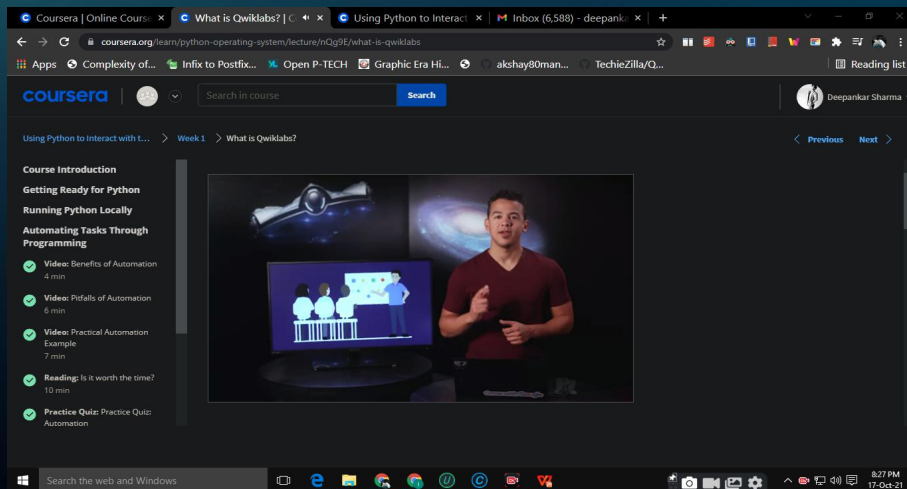
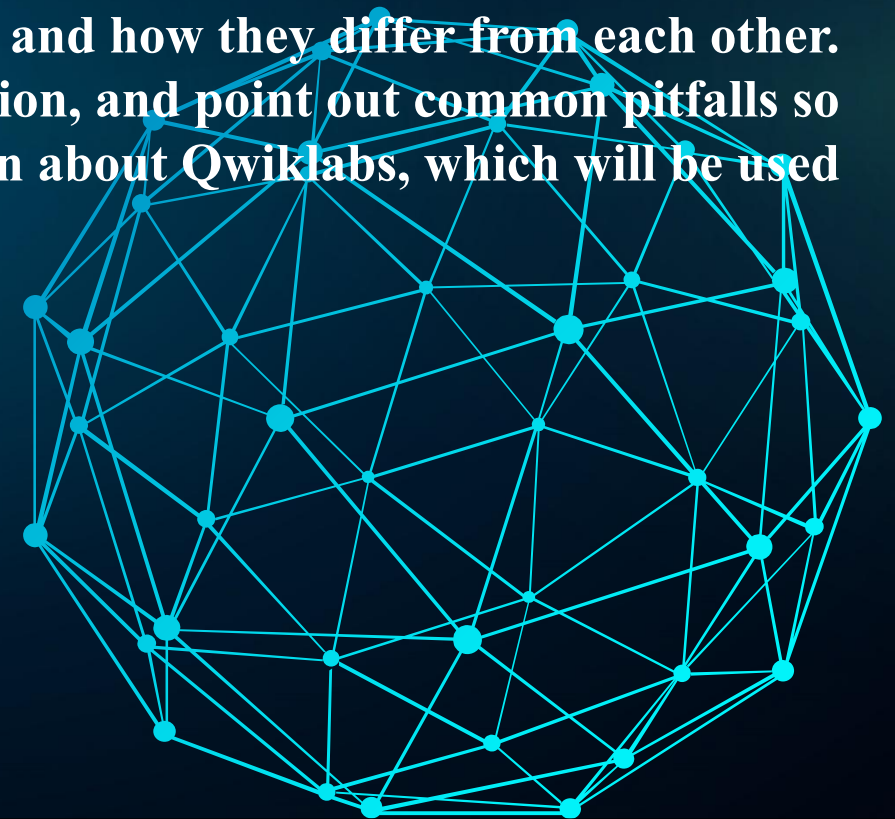


Final Project



# Getting Your Python On

In this module, you'll learn about the different types of operating systems, and how you can get your python code ready to interact with the operating system. We'll learn about getting your environment set up and installing additional Python modules that will help you along the way. We'll rundown interpreted versus compiled language, and how they differ from each other. We'll dive into the benefits of automation, and point out common pitfalls so you can avoid them. Finally, we'll learn about Qwiklabs, which will be used for graded assessments.

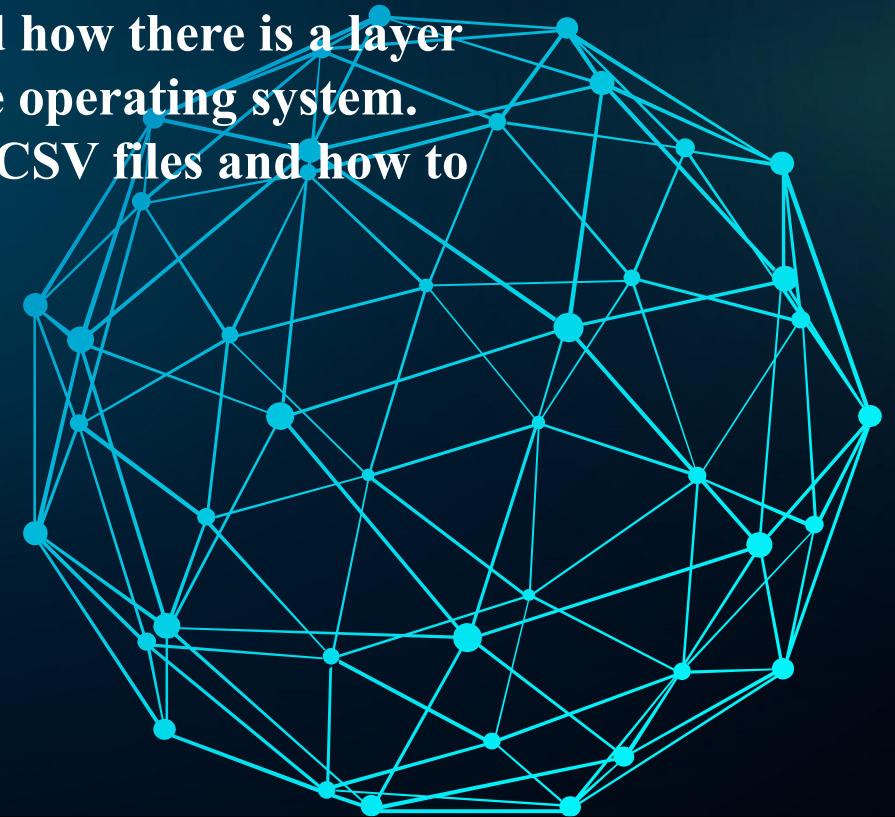
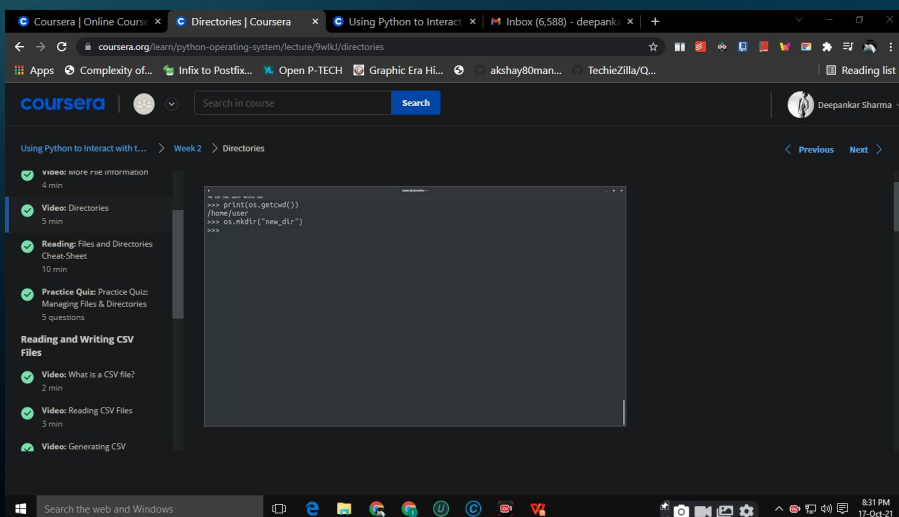






# Managing Files with Python

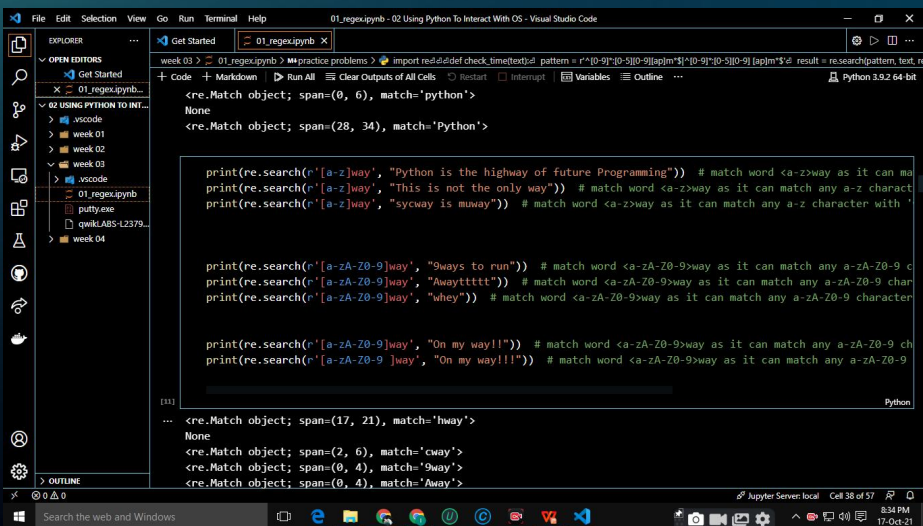
**In this module, you'll learn about reading and writing to files and the commands that will enable you to do this. We'll learn the importance of managing files and how we can navigate through different directories. We'll understand how to work with files and how there is a layer of abstraction between Python and the operating system. Finally, we'll dive into learning about CSV files and how to best utilize them.**





# Regular Expressions

In this module, you'll learn about what a regular expression is and why you would use one. We'll dive into the basics of regular expressions and give examples of wildcards, repetition qualifiers, escapare characters, and more. Next up, we'll explore advanced regular expressions and deep dive on repetition qualifiers. You'll tackle new exercises like capturing groups and extracting PIDs using regexes. Finally, we'll provide a cheat sheet to serve as your go-to guide for regular expressions.



```
File Edit Selection View Go Run Terminal Help 01_regex.py - 02 Using Python To Interact With OS - Visual Studio Code
EXPLORER
  Get Started
  01_regex.py
  02 USING PYTHON TO INT...
    week 01
    week 02
    week 03
    week 04
  putty.exe
  qwik_ABS-12379...
  week 04
OUTLINE
  01_regex.py
  02 USING PYTHON TO INT...
  putty.exe
  qwik_ABS-12379...
  week 04
Python 3.9.2 64-bit

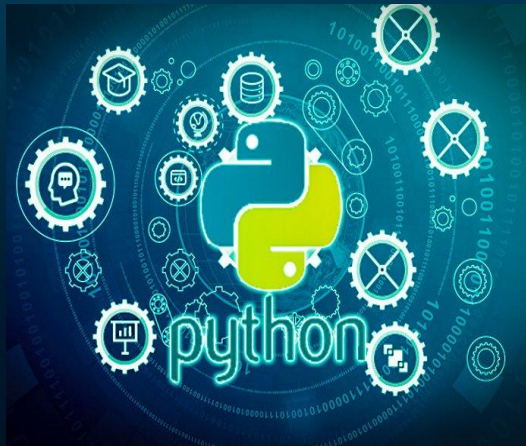
<re.Match object; span=(0, 6), match='python'>
None
<re.Match object; span=(28, 34), match='Python'>

print(re.search(r'[a-z]way', 'Python is the highway of future Programming')) # match word <a-zway> as it can ma
print(re.search(r'[a-z]way', 'This is not the only way')) # match word <a-zway> as it can match any a-z charact
print(re.search(r'[a-z]way', 'sycway is mway')) # match word <a-zway> as it can match any a-z character with '

print(re.search(r'[a-zA-Z0-9]way', '9ways to run')) # match word <a-zA-Z0-9way> as it can match any a-zA-Z0-9 c
print(re.search(r'[a-zA-Z0-9]way', 'Awayttttt')) # match word <a-zA-Z0-9way> as it can match any a-zA-Z0-9 char
print(re.search(r'[a-zA-Z0-9]way', 'whey')) # match word <a-zA-Z0-9way> as it can match any a-zA-Z0-9 character

print(re.search(r'[a-zA-Z0-9]way', 'On my way!!!')) # match word <a-zA-Z0-9way> as it can match any a-zA-Z0-9 ch
print(re.search(r'[a-zA-Z0-9]way', 'On my way!!!')) # match word <a-zA-Z0-9way> as it can match any a-zA-Z0-9

[11]
... <re.Match object; span=(17, 21), match='hway'>
None
<re.Match object; span=(2, 6), match='cway'>
<re.Match object; span=(0, 4), match='9way'>
<re.Match object; span=(0, 4), match='Away'>
```

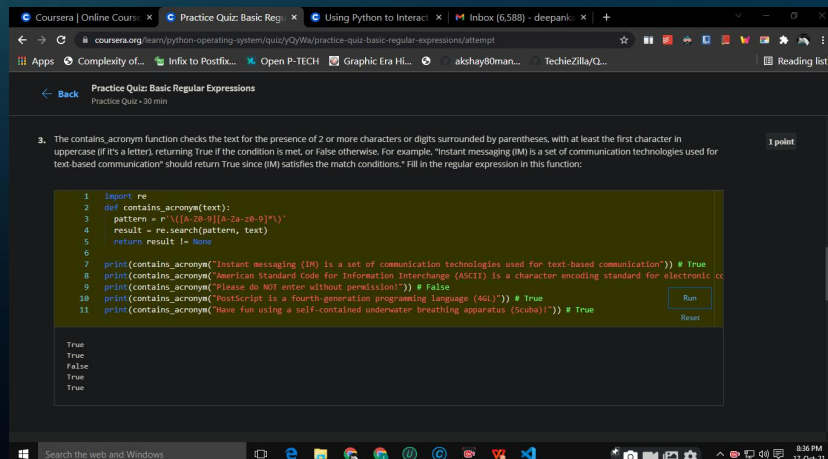






# Managing Data and Processes

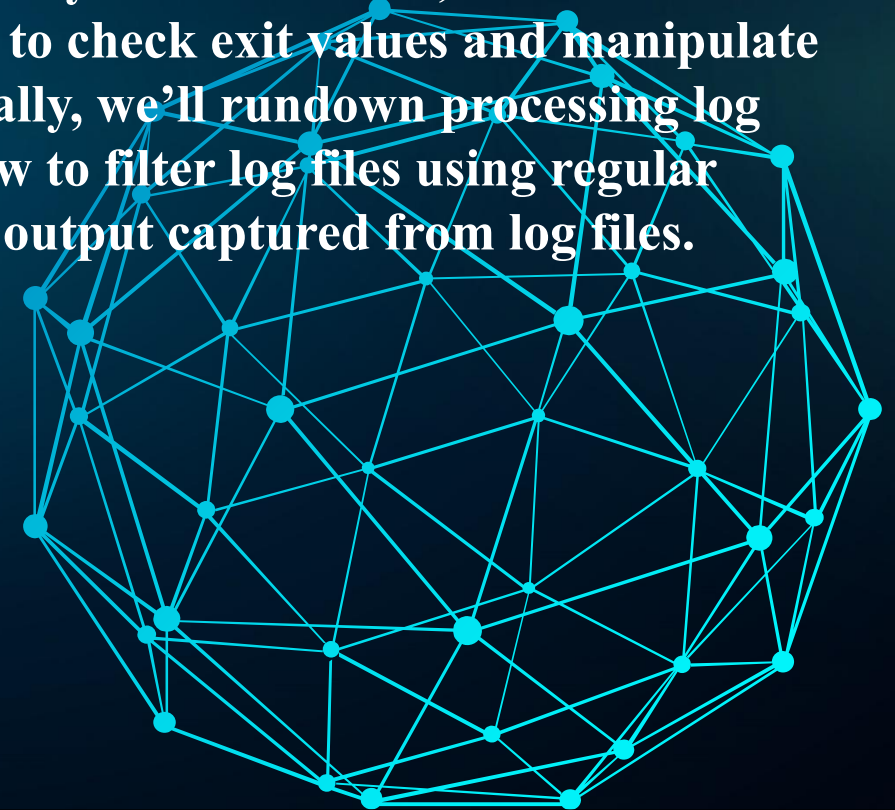
In this module, we'll learn about reading and writing to data files based on an interaction with the user. Along the way, we'll dive into standard streams, environment variables, and command line arguments. Next, we'll jump into Python subprocesses, including system commands and how they can be used. We'll review how to obtain output from a system command, and dive into subprocess management, including how to check exit values and manipulate the normal versus error exit values. Finally, we'll rundown processing log files, and will cover what a log file is, how to filter log files using regular expressions, and how to understand the output captured from log files.



```
3. The contains_acronym function checks the text for the presence of 2 or more characters or digits surrounded by parentheses, with at least the first character in uppercase (if it's a letter), returning True if the condition is met, or False otherwise. For example, "Instant messaging (IM) is a set of communication technologies used for text-based communication" should return True since (IM) satisfies the match conditions. Fill in the regular expression in this function: 1 point
```

```
1 import re
2 def contains_acronym(text):
3     pattern = r'\([A-Z0-9][A-Za-z0-9]*\)'
4     result = re.search(pattern, text)
5     return result is not None
6
7 print(contains_acronym("Instant messaging (IM) is a set of communication technologies used for text-based communication")) # True
8 print(contains_acronym("American Standard Code for Information Interchange (ASCII) is a character encoding standard for electronic communication")) # False
9 print(contains_acronym("Please do NOT enter without permission!")) # False
10 print(contains_acronym("PostScript is a fourth-generation programming language (4GL)")) # True
11 print(contains_acronym("Have fun using a self-contained underwater breathing apparatus (Scuba)")) # True
```

True  
True  
False  
True  
True





# Thank You

---

`print "RUN THE TRAP"`