

Graphic Era Hill University



MOOCS Based Seminar

DEEPAKAR SHARMA

COURSE: BCA

STUDENT ID : 20041299

UNIVERSITY ROLL NO : 2092014



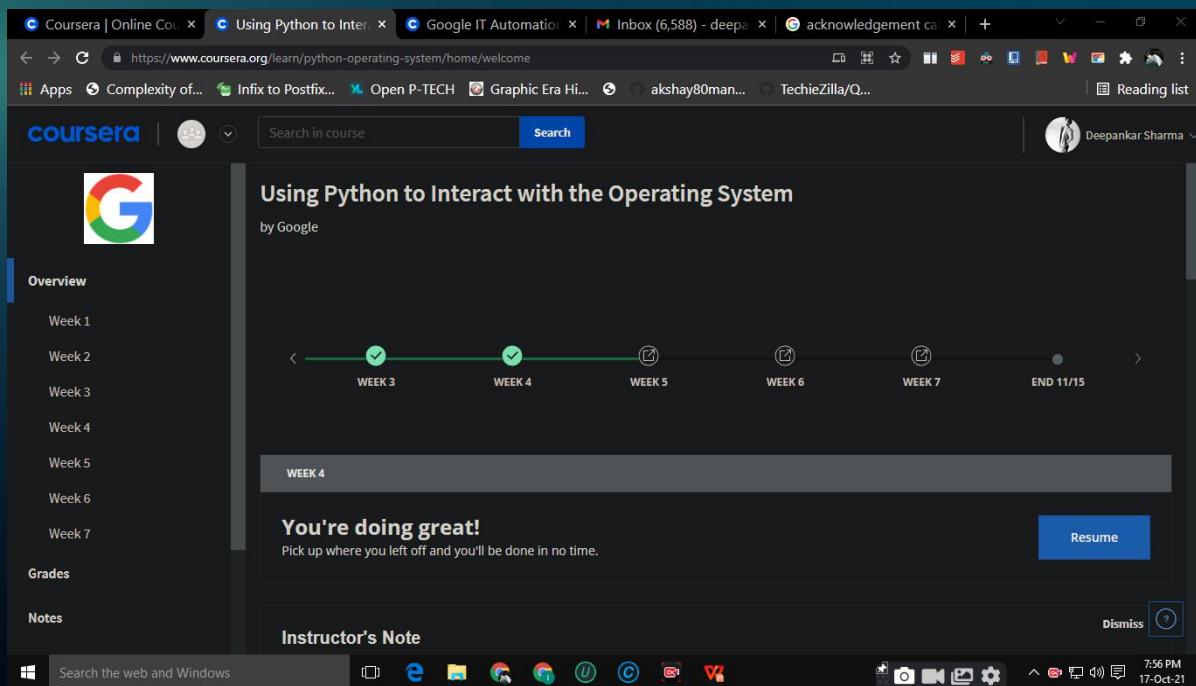
PRESENTATION

Using Python to Interact with the Operating System



Google IT Automation with Python Professional Certificate

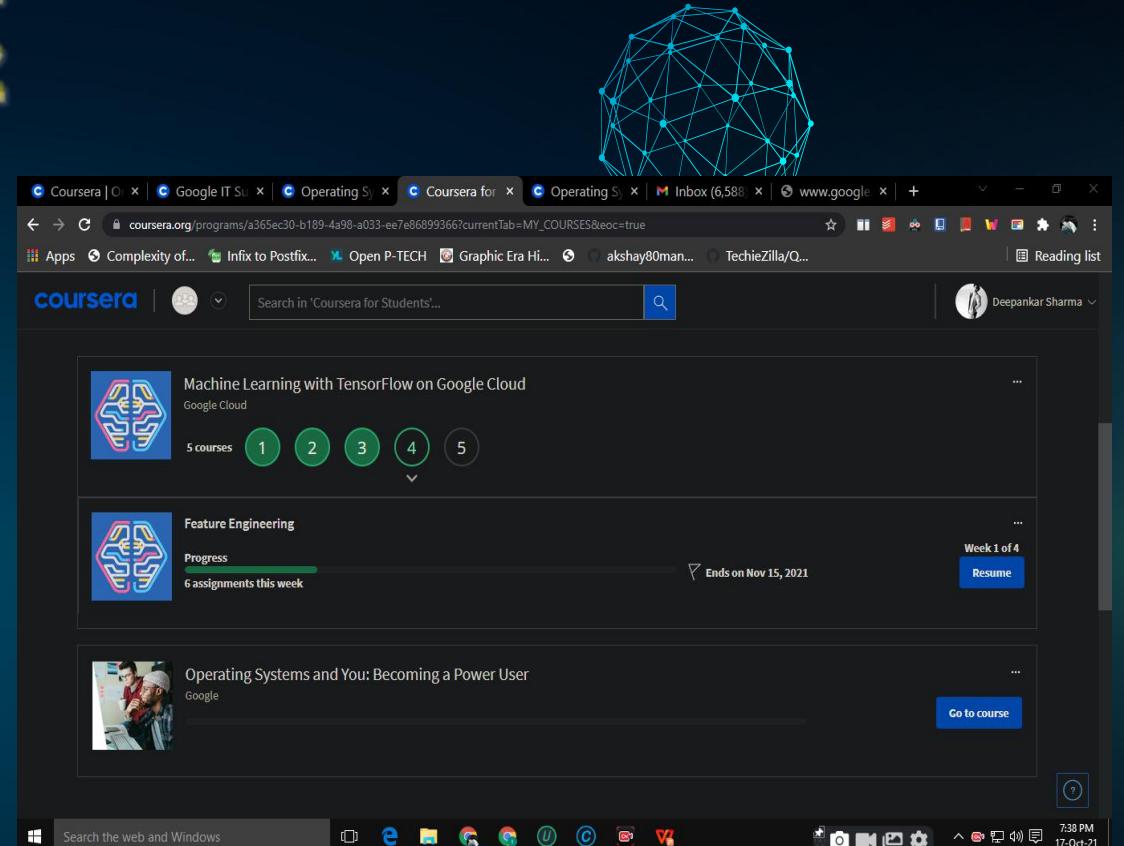
Using Python to Interact with the Operating System



Acknowledgement

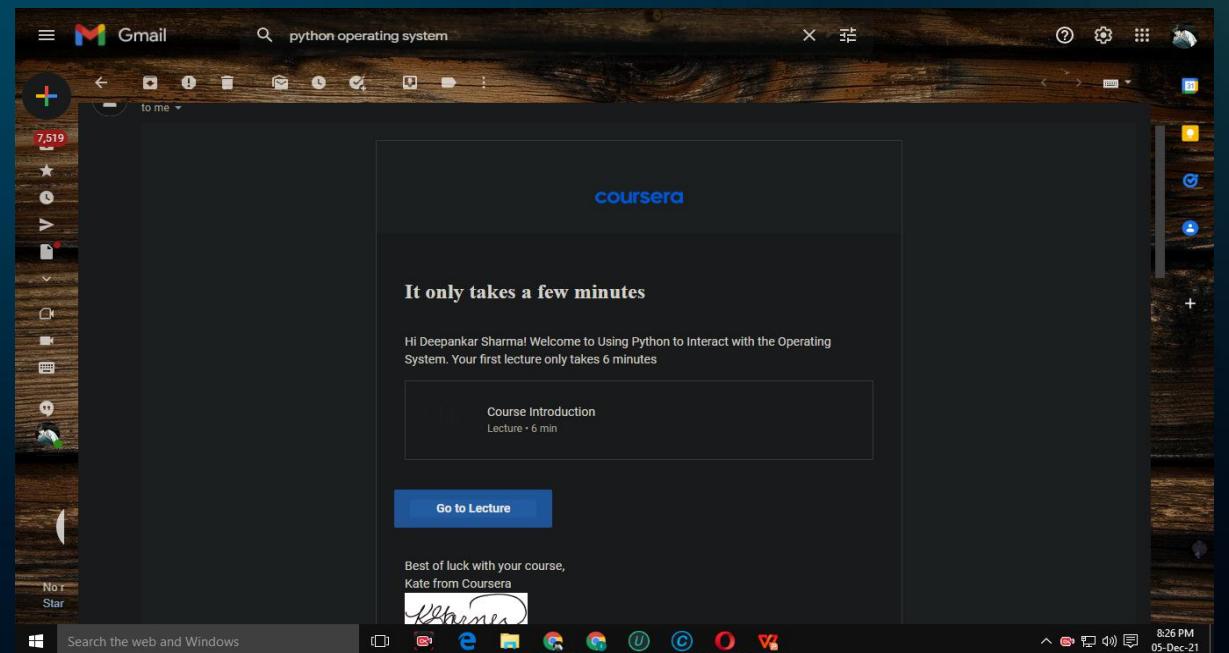
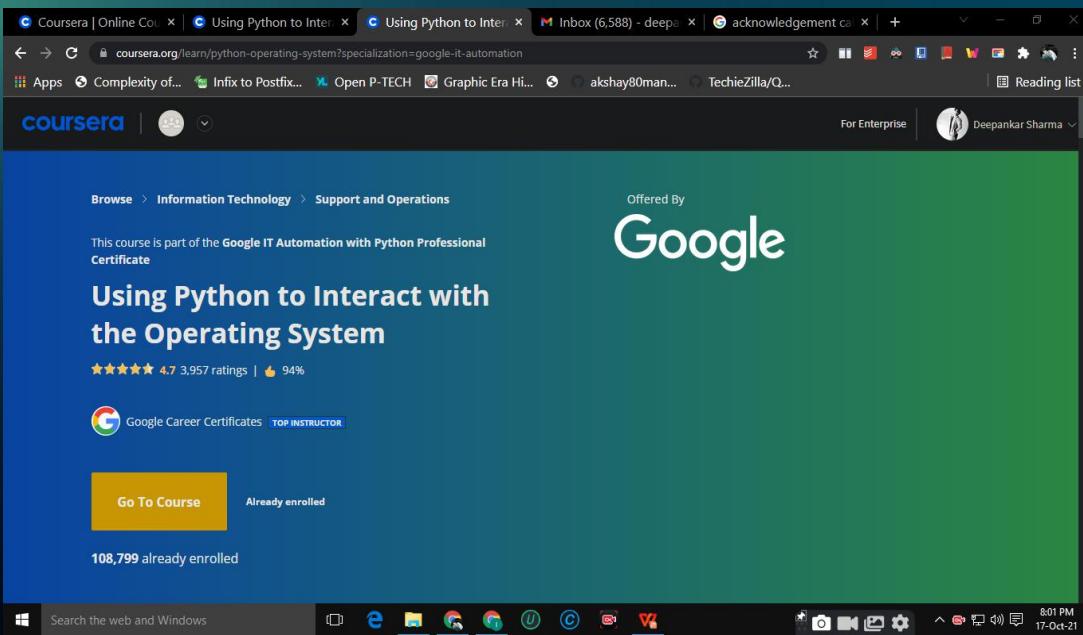
I would like to express my special thanks of gratitude to my teacher Mayurika Ma'am , who gave me the golden opportunity to do this wonderful MOOCs seminar on the topic "Using Python to Interact with the Operating System". And also for providing with all the facilities that were required. The fact is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

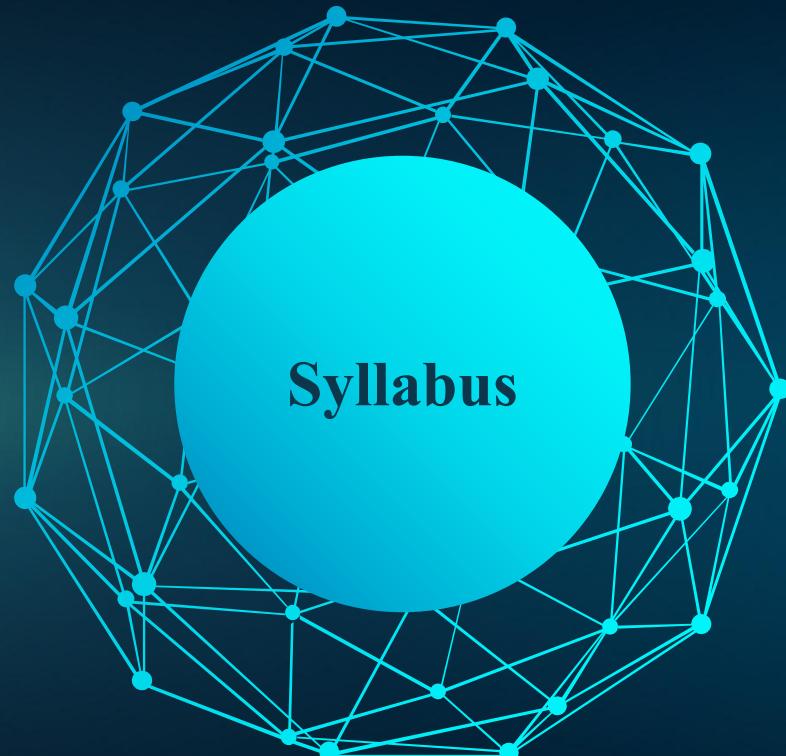
REGARDS,
DEEPAK SHARMA



By the end of this course, we are going to be able to manipulate files and processes on your computer's operating system. we also have learned about regular expressions -- a very powerful tool for processing text files -- and we get practice using the Linux command line on a virtual machine. And, this might feel like a stretch right now, but we 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 also explain how to set up our own developer environment in our machine. This is a key step in being able to write and deploy powerful automation tools.

Registration Email





Getting Your Python On



Managing Files with Python



Regular Expressions



Managing Data and Processes



Testing in Python



Bash Scripting

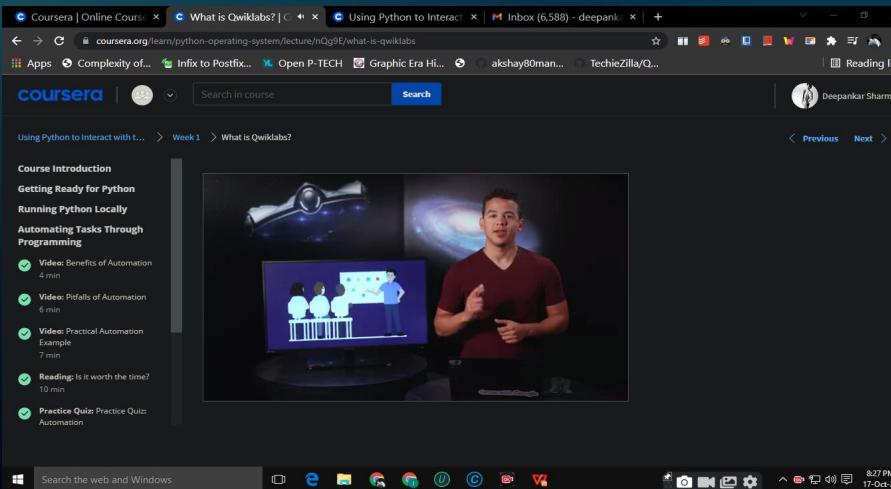
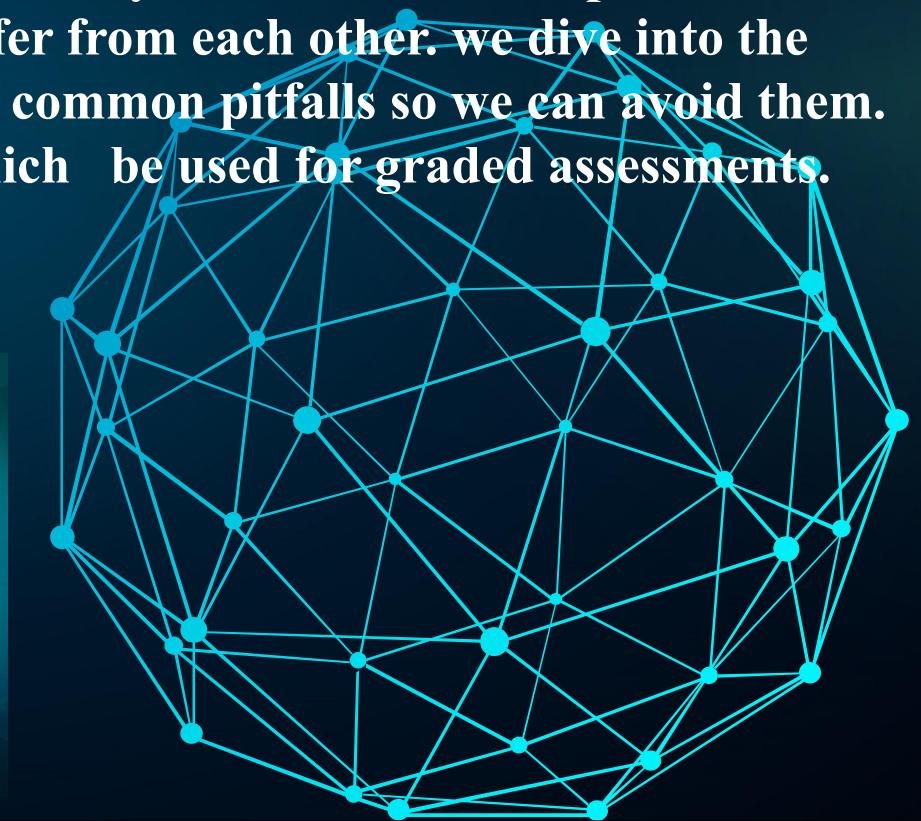


Final Project



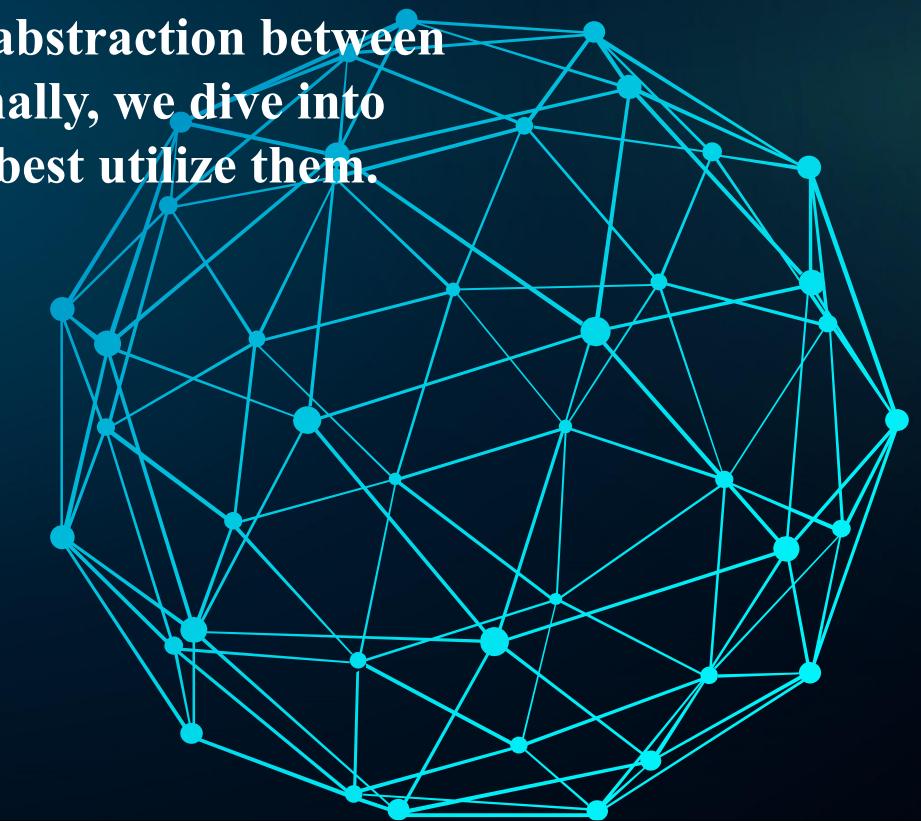
Getting Your Python On

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

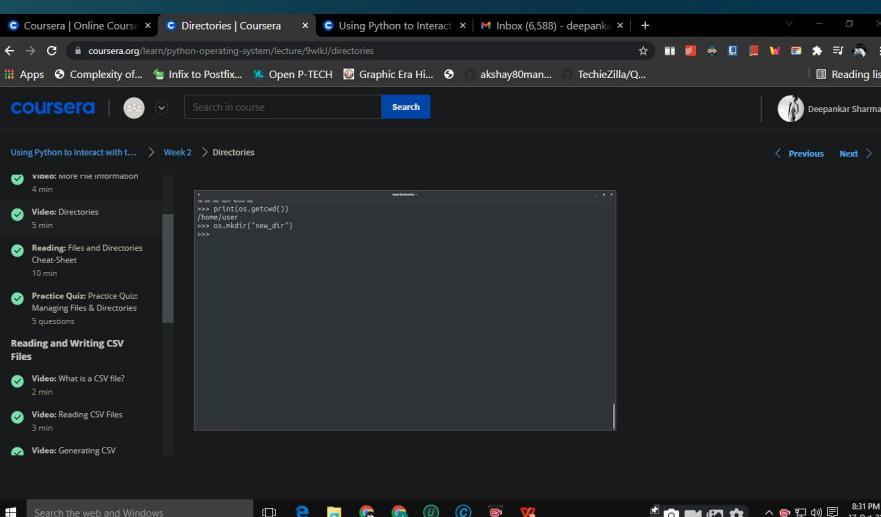




Managing Files with Python

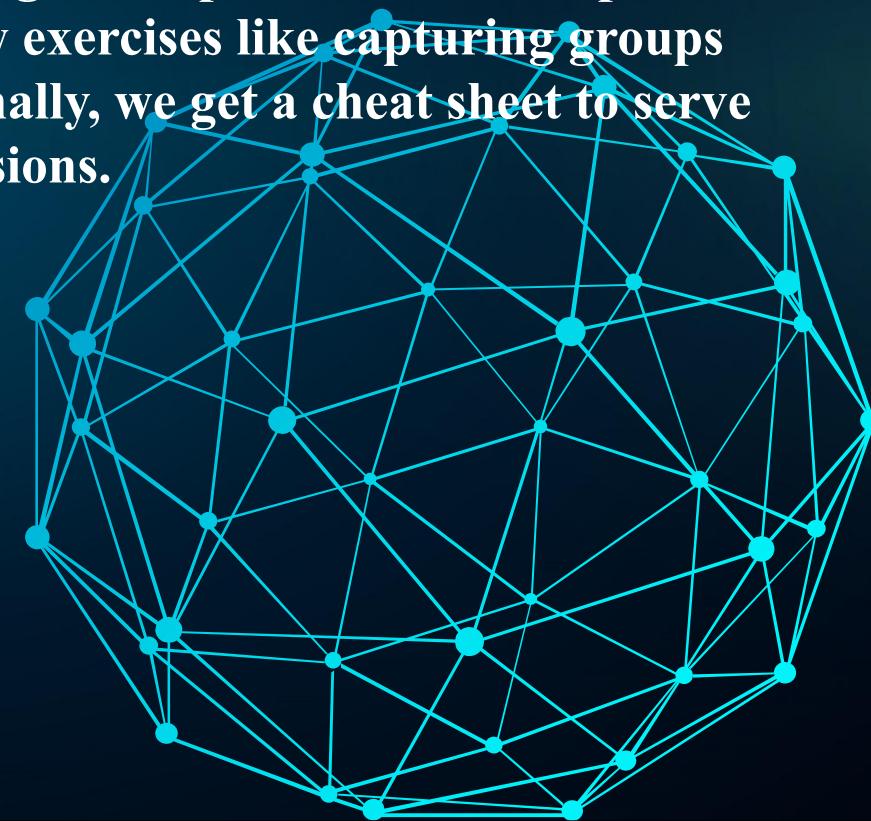


In this module, we learn about reading and writing to files and the commands that enable us to do this. we learn the importance of managing files and how we can navigate through different directories. we understand how to work with files and how there is a layer of abstraction between Python and the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



Regular Expressions

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



A screenshot of Visual Studio Code showing a Python code editor. The code is demonstrating regular expression usage with the `re` module. It includes examples for matching words like "way" and "python", and for matching patterns within words like "9ways" and "a-zA-Z0-9ways". The code editor interface is visible, including the menu bar, toolbars, and status bar.

```
File Edit Selection View Go Run Terminal Help 01_regexipynb - 02 Using Python To Interact With OS - Visual Studio Code
EXPLORER ... 01_regexipynb x
OPEN EDITORS
01_regexipynb x
02 USING PYTHON TO INT...
01_regexipynb x
week 03 > 01_regexipynb > M practice problems > import re&lt;def check_time(text): pattern = r'^[0-9]*-[0-5][0-9]apm$|^([0-9]*-[0-5][0-9])apm$</s> result = re.search(pattern, text, re
+ Code + Markdown | ▶ Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.9.2 64-bit
None
<re.Match object; span=(0, 6), match='python'>
<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-z>way as it can match any a-z character
print(re.search(r'[a-z]way', "This is not the only way")) # match word <a-z>way as it can match any a-z character
print(re.search(r'[a-z]way', "suyway is muway")) # match word <a-z>way as it can match any a-z character with a-z
print(re.search(r'[a-zA-Z0-9]way", "9ways to run")) # match word <a-zA-Z0-9>way as it can match any a-zA-Z0-9 character
print(re.search(r'[a-zA-Z0-9]way", "Awayttttt")) # match word <a-zA-Z0-9>way as it can match any a-zA-Z0-9 character
print(re.search(r'[a-zA-Z0-9]way", "whey")) # match word <a-zA-Z0-9>way 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-Z0-9>way 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-Z0-9>way as it can match any a-zA-Z0-9 character

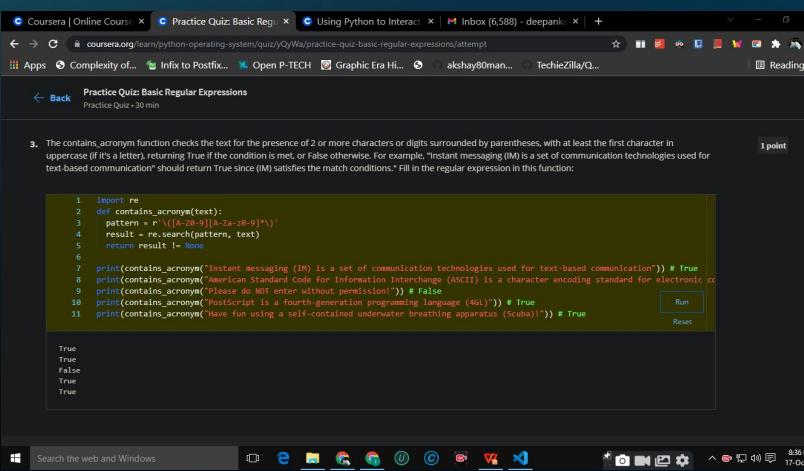
<re.Match object; span=(17, 21), match='hway'>
None
<re.Match object; span=(2, 6), match='cway'>
<re.Match object; span=(0, 4), match='9way'>
<re.Match object; span=(0, 4), match='away'>

Python
[11]
... > OUTLINE
Search the web and Windows
e 8:24 PM 17-Oct-21
```



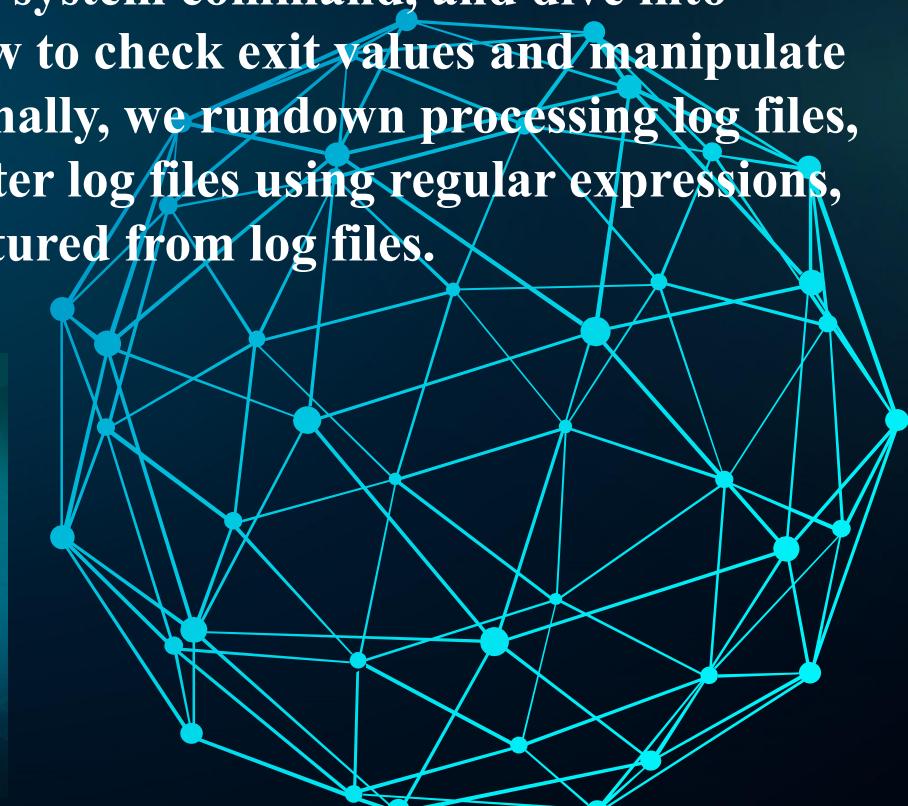
Managing Data and Processes

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



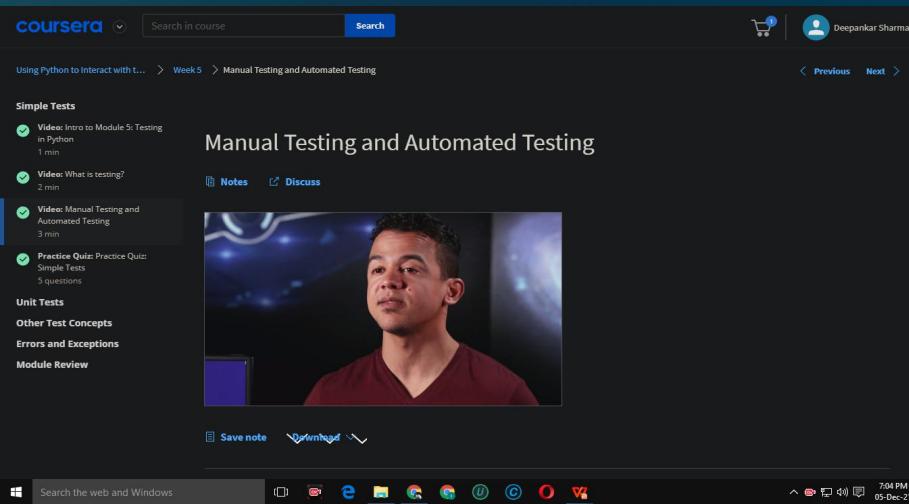
```
1 import re
2 def contains_acronym(text):
3     pattern = re.compile(r'([A-Z][A-Z0-9]*\b)')
4     result = re.search(pattern, text)
5     return result != None
6
7 print(contains_acronym("Instant messaging (IM) is a set of communication technologies used for text-based communication"))
8 print(contains_acronym("American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

True
True
False
True

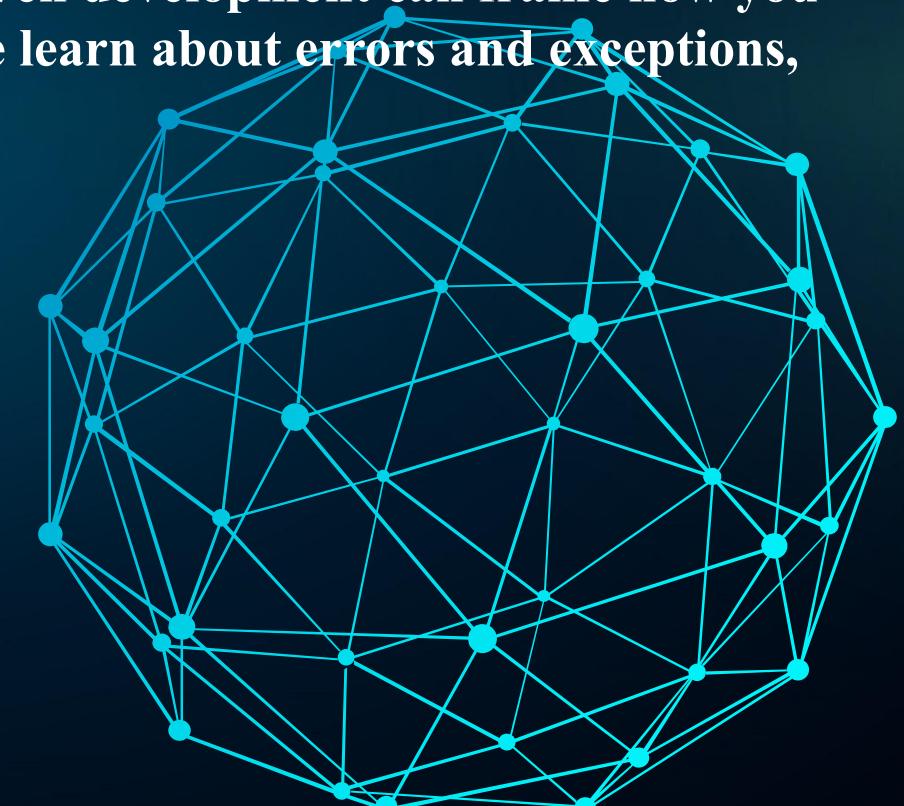


Testing in Python

In this module, we learn how to create tests in Python. We'll cover what testing is all about and dive into the differences between manual versus automated testing. Next, we'll explore what unit tests are intended to do and how to write them. Then, we'll learn about other test concepts like black box versus white box tests and how test-driven development can frame how you design and write your code. Finally, we learn about errors and exceptions, and how to combat them.

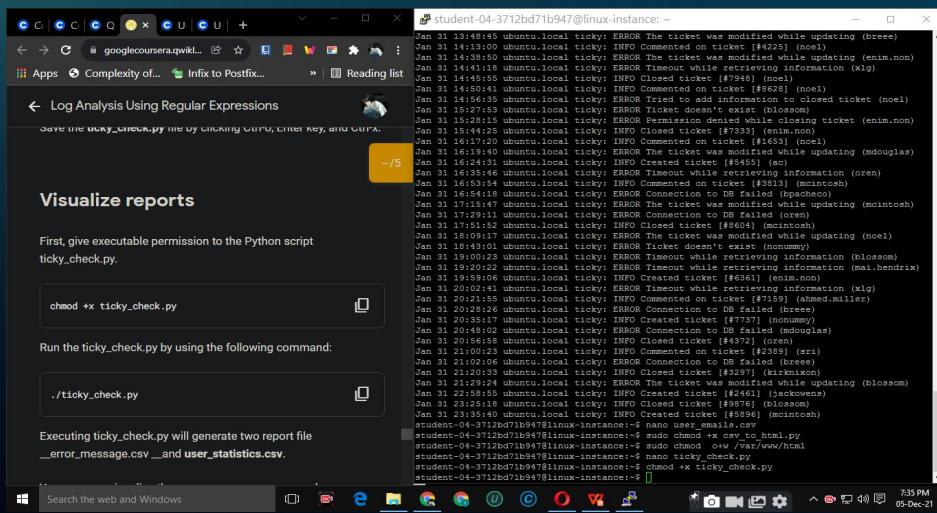


The screenshot shows a Coursera course page. The top navigation bar includes the Coursera logo, a search bar, and a 'Search' button. The course title is 'Using Python to Interact with the Web' and the module is 'Week 5: Manual Testing and Automated Testing'. The module page title is 'Manual Testing and Automated Testing'. It features a video thumbnail of a man, course navigation links for 'Simple Tests', 'Unit Tests', 'Other Test Concepts', 'Errors and Exceptions', and 'Module Review', and download buttons for 'Save note' and 'Download'.



Bash Scripting

In this module, we be exposed to what the Linux OS has to offer and we learn about Bash scripting. We'll go over basic Linux commands and explore the many processes Linux has to offer, including a key concept called redirection. We'll then deep dive into creating bash scripts using variables and globs. Finally, we'll learn about advanced bash concepts and develop an understanding of when to use bash versus Python.



The image shows a Windows desktop environment. In the center is a terminal window with the following text:

```
student-04-3712bd71b947@linux-instance: ~
Jan 31 13:04:45 ubuntu.local ticky: ERROR The ticket was modified while updating (breeze)
Jan 31 13:04:45 ubuntu.local ticky: ERROR The ticket was modified while updating (enim.non)
Jan 31 14:41:18 ubuntu.local ticky: ERROR Timeout while retrieving information (xig)
Jan 31 14:41:18 ubuntu.local ticky: INFO Closed ticket (#7958) (noel)
Jan 31 14:56:35 ubuntu.local ticky: ERROR Tried to add information to closed ticket (noel)
Jan 31 15:27:53 ubuntu.local ticky: ERROR Ticket doesn't exist (blossom)
Jan 31 15:28:05 ubuntu.local ticky: ERROR Permission denied while closing ticket (enim.non)
Jan 31 15:42:42 ubuntu.local ticky: INFO Commented on ticket (#14531) (noel)
Jan 31 16:17:20 ubuntu.local ticky: INFO Commented on ticket (#14531) (noel)
Jan 31 16:19:40 ubuntu.local ticky: ERROR The ticket was modified while updating (mcdouglas)
Jan 31 16:28:38 ubuntu.local ticky: INFO Created ticket (#5055) (ac)
Jan 31 16:53:46 ubuntu.local ticky: INFO Commented on ticket (#39511) (mcintosh)
Jan 31 16:53:54 ubuntu.local ticky: ERROR Connection to DB failed (bphacheo)
Jan 31 16:54:18 ubuntu.local ticky: ERROR The ticket was modified while updating (mcintosh)
Jan 31 17:13:47 ubuntu.local ticky: ERROR Permission denied while updating (mcintosh)
Jan 31 17:13:47 ubuntu.local ticky: INFO Commented on ticket (#7151) (ahmed.mallek)
Jan 31 17:58:152 ubuntu.local ticky: INFO Closed ticket (#8604) (mcintosh)
Jan 31 18:09:17 ubuntu.local ticky: ERROR The ticket was modified while updating (noel)
Jan 31 18:49:01 ubuntu.local ticky: ERROR Ticket doesn't exist (nonummy)
Jan 31 19:02:12 ubuntu.local ticky: INFO Commented on ticket (#14531) (blossom)
Jan 31 19:20:22 ubuntu.local ticky: ERROR Timeout while retrieving information (mai.hendrix)
Jan 31 19:59:06 ubuntu.local ticky: INFO Created ticket (#6361) (enim.non)
Jan 31 20:14:45 ubuntu.local ticky: ERROR Timeout while retrieving information (xig)
Jan 31 20:14:45 ubuntu.local ticky: INFO Commented on ticket (#7151) (ahmed.mallek)
Jan 31 20:28:26 ubuntu.local ticky: ERROR Connection to DB failed (breeze)
Jan 31 20:35:17 ubuntu.local ticky: INFO Created ticket (#7737) (nonummy)
Jan 31 20:48:02 ubuntu.local ticky: INFO Commented on ticket (#7737) (nonummy)
Jan 31 20:48:02 ubuntu.local ticky: INFO Closed ticket (#4577) (mcdouglas)
Jan 31 21:00:123 ubuntu.local ticky: INFO Commented on ticket (#2389) (eriz)
Jan 31 21:02:07 ubuntu.local ticky: ERROR Connection to DB failed (breeze)
Jan 31 21:29:24 ubuntu.local ticky: ERROR The ticket was modified while updating (blossom)
Jan 31 22:58:55 ubuntu.local ticky: INFO Created ticket (#2461) (jekowens)
Jan 31 23:25:13 ubuntu.local ticky: INFO Closed ticket (#9876) (blossom)
Jan 31 23:38:35 ubuntu.local ticky: INFO Commented on ticket (#8563) (mcintosh)
student-04-3712bd71b947@linux-instance: ~$ sudo chmod +x csv_to_html.py
student-04-3712bd71b947@linux-instance: ~$ sudo chmod +x ./var/www/html/ticky
student-04-3712bd71b947@linux-instance: ~$ ./ticky
student-04-3712bd71b947@linux-instance: ~$ ./ticky_check.py
student-04-3712bd71b947@linux-instance: ~$
```

Below the terminal is a Python script interface with the following text:

```
Visualize reports
First, give executable permission to the Python script
ticky_check.py.

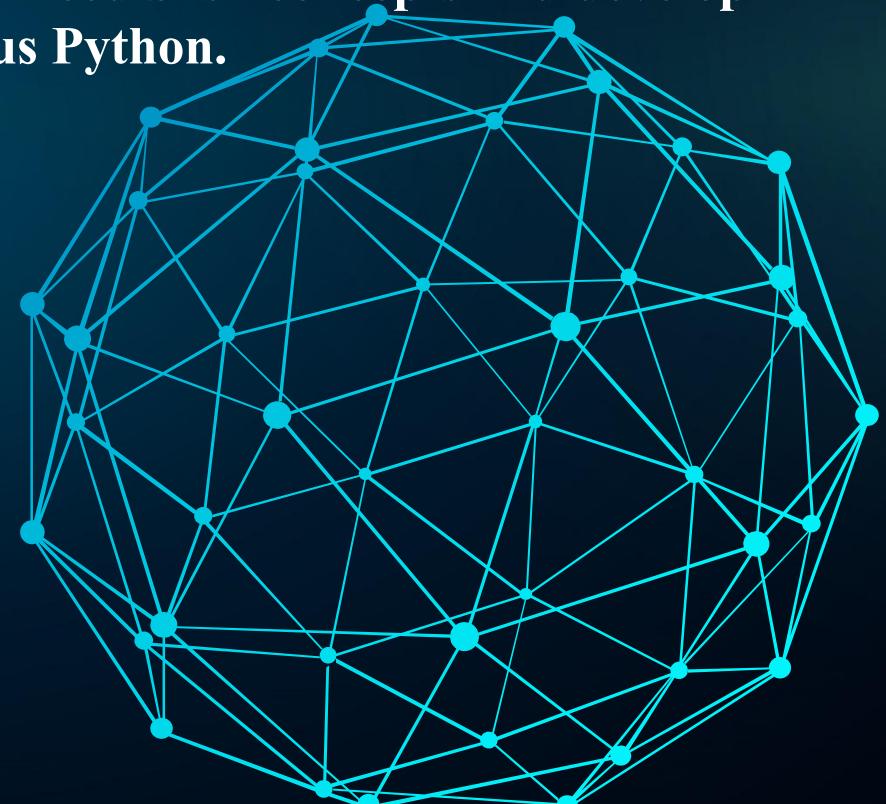
chmod +x ticky_check.py

Run the ticky_check.py by using the following command:

./ticky_check.py

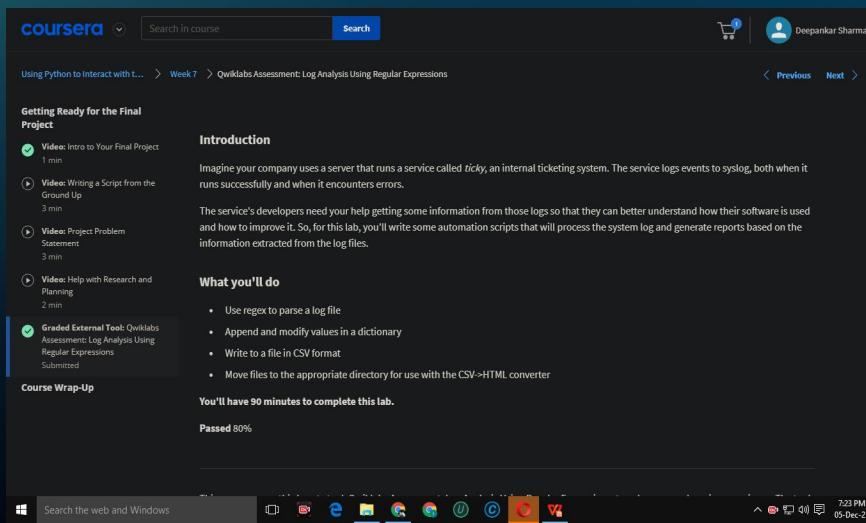
Executing ticky_check.py will generate two report file
_error_message.csv _and user_statistics.csv.
```

At the bottom is a Windows taskbar with various icons.



Final Project

In this module, we put everything we've learned so far into action! we apply your scripting knowledge to tackle a challenging final project: writing a script that scans for a specific error in the log files. we are given a problem statement to understand the challenge, conduct some research to see what options are available, then begin planning how we intend to solve the problem. Lastly, we write the code to implement our solution!



Search in course **Search**

Using Python to Interact with... > Week 7 > Qwiklabs Assessment: Log Analysis Using Regular Expressions

Getting Ready for the Final Project

- Video: Intro to Your Final Project 1 min
- Video: Writing a Script from the Ground Up 3 min
- Video: Project Problem Statement 3 min
- Video: Help with Research and Planning 2 min

Graded External Tool: Qwiklabs Assessment: Log Analysis Using Regular Expressions Submitted

Course Wrap-Up

You'll have 90 minutes to complete this lab.

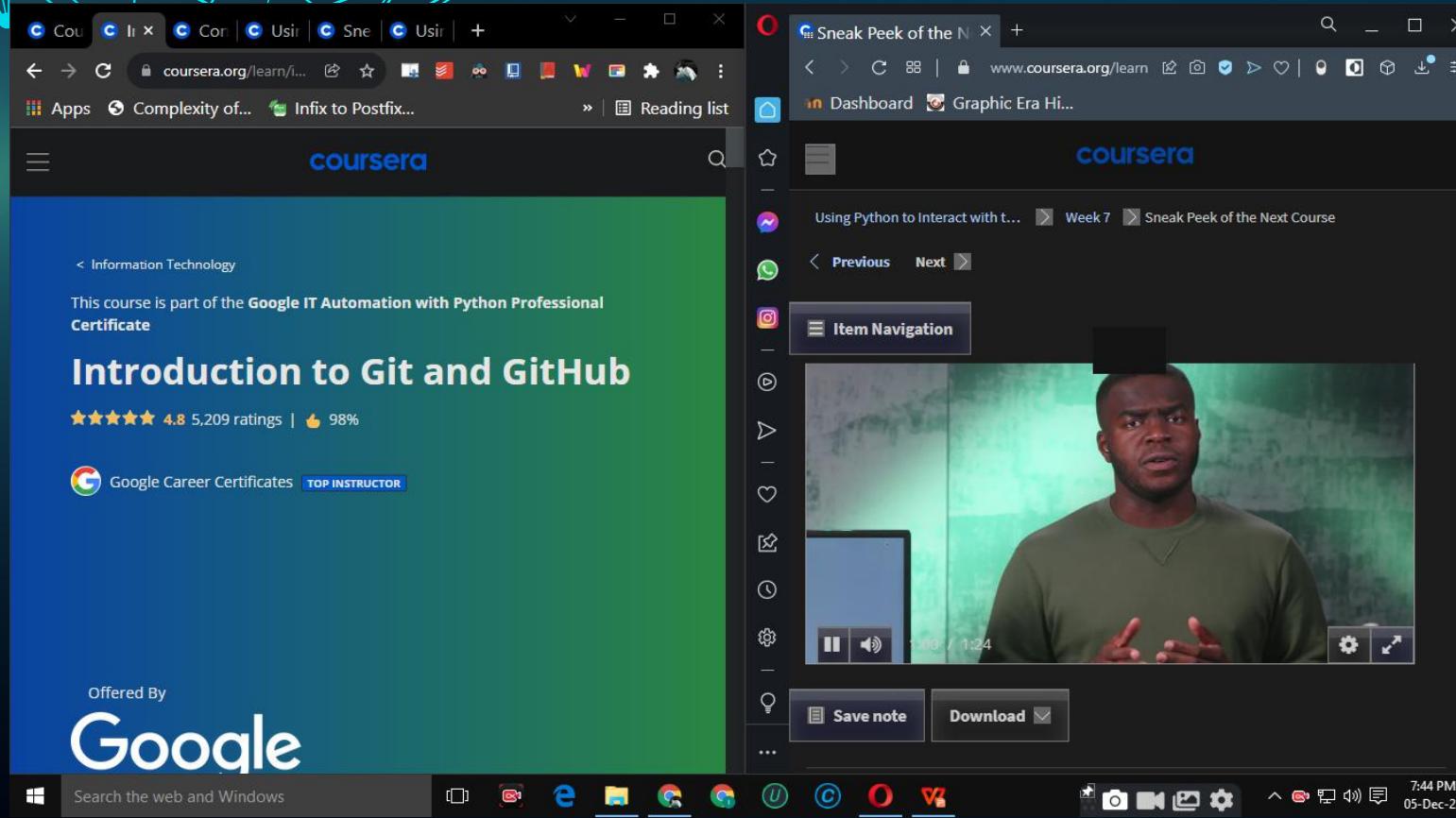
Passed 80%

Search the web and Windows



Sneak Peek of the Next Course

Introduction to Git and Github !!!



The screenshot shows a web browser window with the Coursera website. The course title is 'Introduction to Git and GitHub'. It is part of the 'Google IT Automation with Python Professional Certificate'. The course has 4.8 ratings and 5,209 reviews. It is offered by Google. The video player shows a man speaking, and the navigation bar includes 'Item Navigation' and 'Save note' buttons.



Course Completion and Conclusion

By now, we're much more familiar with the operating system and the many different tasks that we can accomplish using Python. We know how to create, change, and delete files and directories. We know how to process text with regular expressions both with Python and with system tools. We've learned how to interact with system processes by starting them and sending signals to them. We've got some experience with automatically testing your code to make sure that it does what it should. We've dipped our toes into some Bash scripting and we've topped all of this off by solving a complex real-world problem with the coding road.

