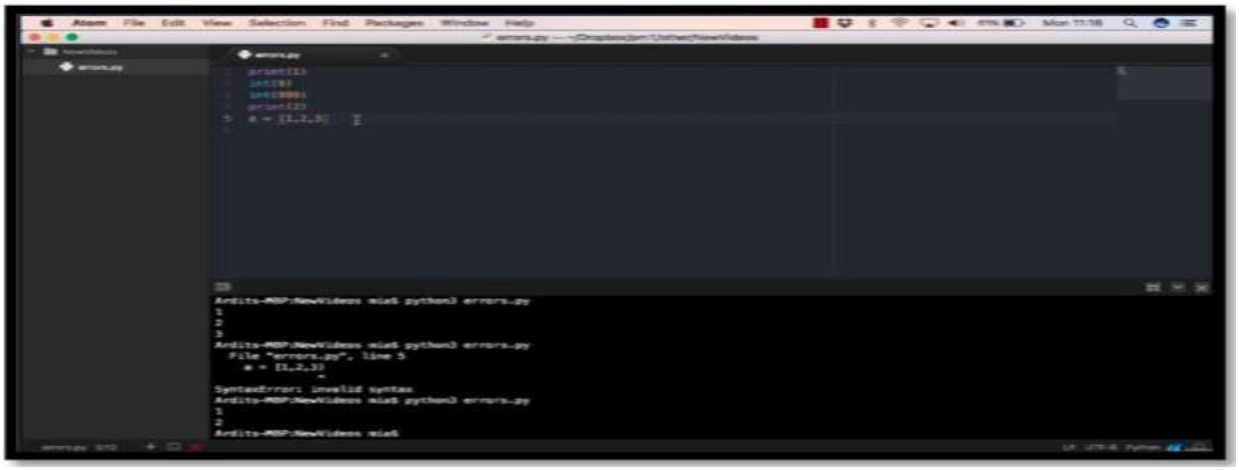
DAILY ASSESSMENT

Date:	**25/05/2020**	**Name:**	**Apeksha S Shetty**
Course:	Python	**USN:**	**4AL16EC006**
Topic:	Fixing Programming Errors Application 3: Build a Website B Locker	**Semester & Section:**	**8TH sem A**
Github Repository	**Apeksha-97**		

Image of the session



REPORT

Fixing Programming Errors :

Errors or mistakes in a program are often referred to as bugs. They are almost always the fault of the programmer. The process of finding and eliminating errors is called debugging. Errors can be classified into three major groups:

- Syntax errors
- Runtime errors
- Logical errors

Syntax errors:

- Python will find these kinds of errors when it tries to parse your program, and exit with an error message without running anything.
- Syntax errors are mistakes in the use of the Python language, and are analogous to spelling or grammar mistakes in a language like English:

- Common Python syntax errors include:
  - leaving out a keyword
  - putting a keyword in the wrong place
  - leaving out a symbol, such as a colon, comma or brackets
  - misspelling a keyword
  - incorrect indentation
  - empty block

Runtime errors:

- If a program is syntactically correct –that is, free of syntax errors
- it will be run by the Python interpreter. However, the program may exit unexpectedly during execution if it encounters a runtime error
- A problem which was not detected when the program was parsed, but is only revealed when a particular line is executed
- When a program comes to a halt because of a runtime error, we say that it has crashed.

Logical errors:

- Logical errors are the most difficult to fix.
- They occur when the program runs without crashing, but produces an incorrect result.
- The error is caused by a mistake in the program's logic. You won't get an error message, because no syntax or runtime error has occurred.

In Python, exceptions can be handled using a try statement. The critical operation which can raise an exception is placed inside the try clause. The code that handles the exceptions is written in the except clause.

Code :

```
import time
from datetime import datetime as dt

hosts_temp=r"D:\Dropbox\pp\block_websites\Demo\hosts"
hosts_path="/etc/hosts"
redirect="127.0.0.1"
while True:
if dt(dt.now().year,dt.now().month,dt.now().day,8) < dt.now() <
dt(dt.now().year,dt.now().month,dt.now().day,16):

print("Working hours...")
with open(hosts_path,'r+') as file:
content=file.read()
for website
in website_list:
if website in content:
pass
else:
file.write(redirect+" "+ website+"\n")
else:
with open(hosts_path,'r+') as file:
content=file.readlines()
file.seek(0)
for line in content:
if not any(website in line for website in website_list):
file.write(line)
file.truncate()
print ("Fun hours...")
time.sleep(5)
```

The python script that can run at system startup to block the access to the particular Web sites. Open PyCharm to edit the code or we can use any IDE you want

We start iterating over the items of website List using a for loop. In the first iteration we would have: any (Website in "trees Are good" for website in website list)

Inside the parenthesis of any()there's another loop that iterates over website list:
1.("face" In "trees are good")
2.("clock" in" trees are good")
3.("trend" In "trees are good")

If anyof the above is True you get the expression evaluated to True. In this case none of them is True, so you get False

If we want to return True (if all of them are True), use all ( ) instead of any( )

So, the part any(website in Line for website in website list) will either be equal to True Or False