

## Exploring a simple Python shell

Review the blogs at Praka (2018) and Szabo (n.d.) and then create a CLI/ shell that implements the following:

- When you enter the command LIST it lists the contents of the current directory
- The ADD command will add the following two numbers together and provide the result
- The HELP command provides a list of commands available
- The EXIT command exits the shell

Add suitable comments to your code. Run the shell you have created, try a few commands, and then answer the questions below.

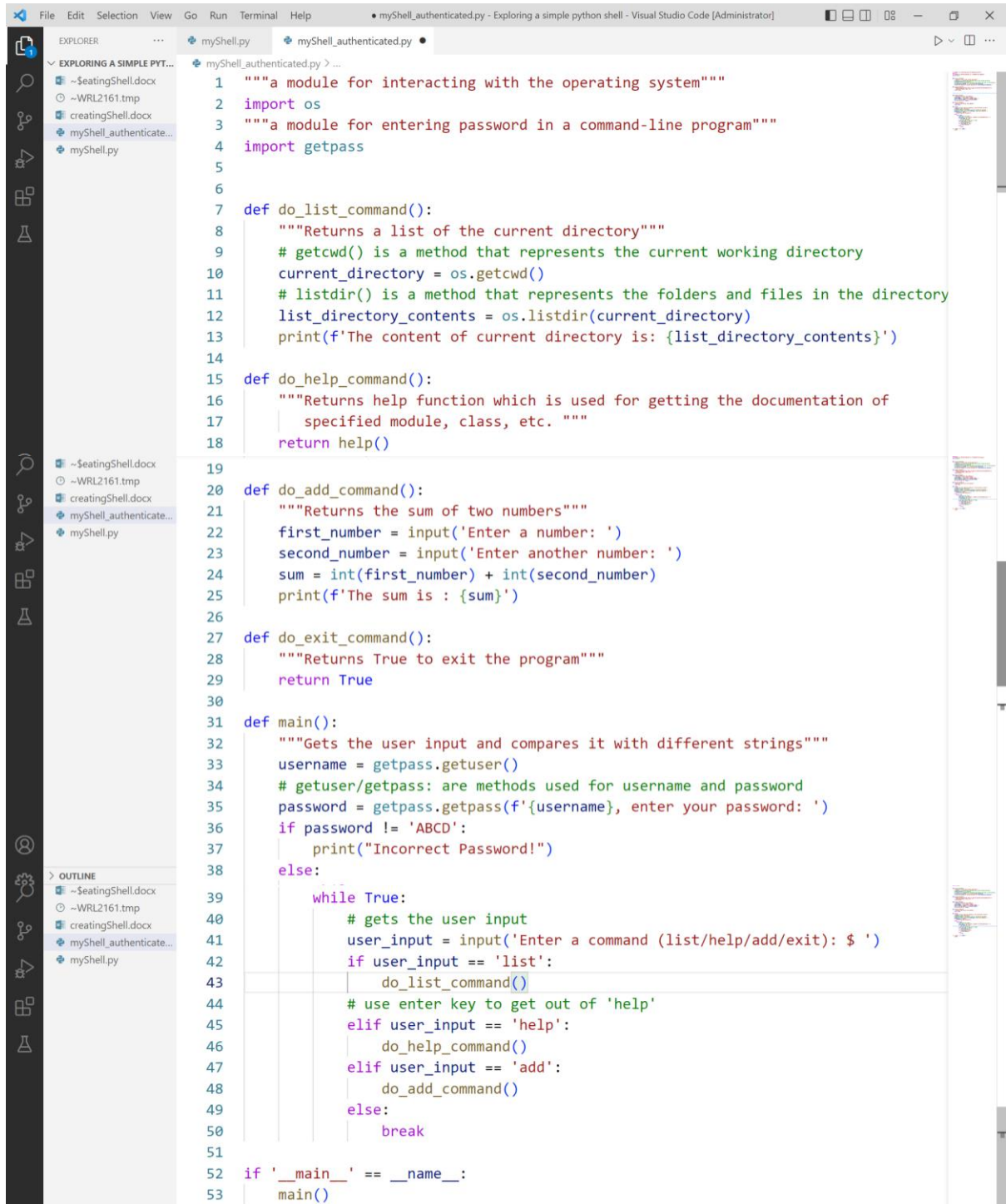
- What are the two main security vulnerabilities with your shell?

‘Authentication attacks’ are one of bash vulnerabilities which should be avoided. In my bash, the security vulnerability can happen in accessing the directory.

- What is one recommendation you would make to increase the security of the shell?

Using ‘getpass’ module and its methods (getuser/getpass), bash can identify unauthorized access and only allows authorized individuals access the directory.

- Add a section to your e-portfolio that provides a (pseudo)code example of changes you would make to the shell to improve its security.



```
1 """a module for interacting with the operating system"""
2 import os
3 """a module for entering password in a command-line program"""
4 import getpass
5
6
7 def do_list_command():
8     """Returns a list of the current directory"""
9     # getcwd() is a method that represents the current working directory
10    current_directory = os.getcwd()
11    # listdir() is a method that represents the folders and files in the directory
12    list_directory_contents = os.listdir(current_directory)
13    print(f'The content of current directory is: {list_directory_contents}')
14
15 def do_help_command():
16     """Returns help function which is used for getting the documentation of
17     specified module, class, etc. """
18     return help()
19
20 def do_add_command():
21     """Returns the sum of two numbers"""
22     first_number = input('Enter a number: ')
23     second_number = input('Enter another number: ')
24     sum = int(first_number) + int(second_number)
25     print(f'The sum is : {sum}')
26
27 def do_exit_command():
28     """Returns True to exit the program"""
29     return True
30
31 def main():
32     """Gets the user input and compares it with different strings"""
33     username = getpass.getuser()
34     # getuser/getpass: are methods used for username and password
35     password = getpass.getpass(f'{username}, enter your password: ')
36     if password != 'ABCD':
37         print("Incorrect Password!")
38     else:
39         while True:
40             # gets the user input
41             user_input = input('Enter a command (list/help/add/exit): $ ')
42             if user_input == 'list':
43                 do_list_command()
44             # use enter key to get out of 'help'
45             elif user_input == 'help':
46                 do_help_command()
47             elif user_input == 'add':
48                 do_add_command()
49             else:
50                 break
51
52 if '__main__' == __name__:
53     main()
```

```
File Edit Selection View Go Run Terminal Help myShell_authenticated.py - Exploring a simple python shell - Visual Studio Code [Administrator]

EXPLORER
EXPLORING A SIMPLE PYTHON SHELL
  ~$eatingShell.docx
  creatingShell.docx
  myShell_authenticated.py
  myShell.py

myShell_authenticated.py
1 """a module for interacting with the operating system"""

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
cmd + v + + + + + X

Microsoft Windows [Version 10.0.19043.1889]
(c) Microsoft Corporation. All rights reserved.

F:\EssexUni\EssesModules\SecureSoftwareDevelopment-June2022\units\unit_12\Portfolio\Portfolio-secureSoftwareDevelopment\Codes\Codio\unit 7\Exploring a simple python shell>python myShell_authenticated.py
emsar, enter your password:
Enter a command (list/help/add/exit): $ list
The content of current directory is: ['creatingShell.docx', 'myShell.py', 'myShell_authenticated.py', '~$eatingShell.docx']
Enter a command (list/help/add/exit): $ add
Enter a number: 12
Enter another number: 13
The sum is : 25
Enter a command (list/help/add/exit): $ exit

F:\EssexUni\EssesModules\SecureSoftwareDevelopment-June2022\units\unit_12\Portfolio\Portfolio-secureSoftwareDevelopment\Codes\Codio\unit 7\Exploring a simple python shell>
```

## Developing an API for a Distributed Environment

Create a RESTful API which can be used to create and delete user records.

### Question 1

Run the API.py code. Take a screenshot of the terminal output. What command did you use to compile and run the code? python api.py

```
F:\EssexUni\EssesModules\SecureSoftwareDevelopment-June2022\units\unit_7\Codio\Developing an api for a distributed environment>python api.py
* Serving Flask app 'api'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 352-708-480
```

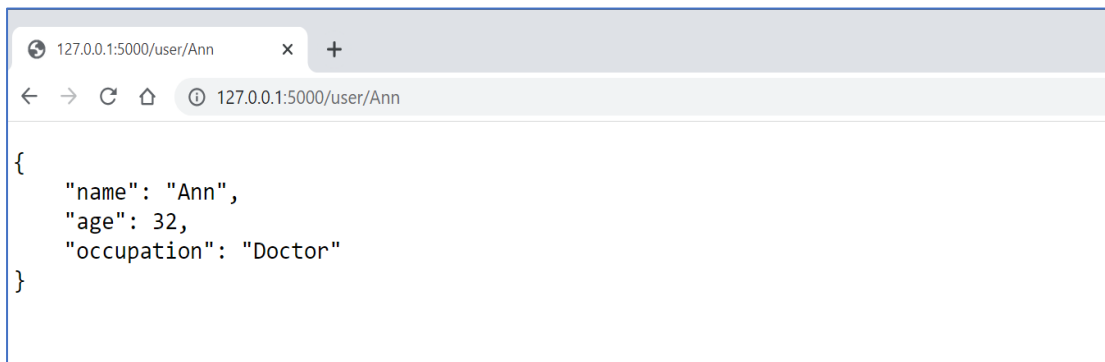
## Question 2

Run the following command at the terminal prompt:

w3m <http://127.0.0.1:5000/user/Ann>

What happens when this command is run, and why?

When the command 'http://127.0.0.1:5000/user/Ann' is run, the function 'get' checks the name and looks for it in the users variable. When it finds 'Ann' in the list, it returns HTTP status code 200 which means the request was successful. So, it returns the all the information related to 'Ann'.



```
{
  "name": "Ann",
  "age": 32,
  "occupation": "Doctor"
}
```

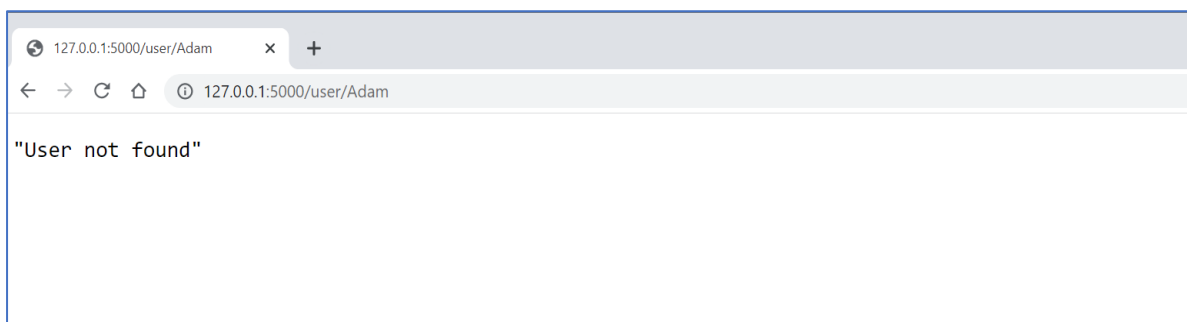
## Question 3

Run the following command at the terminal prompt:

w3m <http://127.0.0.1:5000/user/Adam>

What happens when this command is run, and why?

When the command 'http://127.0.0.1:5000/user/Adam' is run, the function 'get' checks the name and looks for it in the users variable. When it cannot find 'Adam' in the list, it returns HTTP status code 404 which means the requested resources could not be found.



```
"User not found"
```

#### **Question 4**

##### **What capability is achieved by the flask library?**

- Flask allows building a web application.
- Flask is suitable for small application that could get big quickly
- Flask is a microframework that can interact with other tools easily
- Flask is easy to deploy and test
- Flask supports modular programming
- The key feature of the Flask is its flexibility. It is so simple that can be easily rearranged