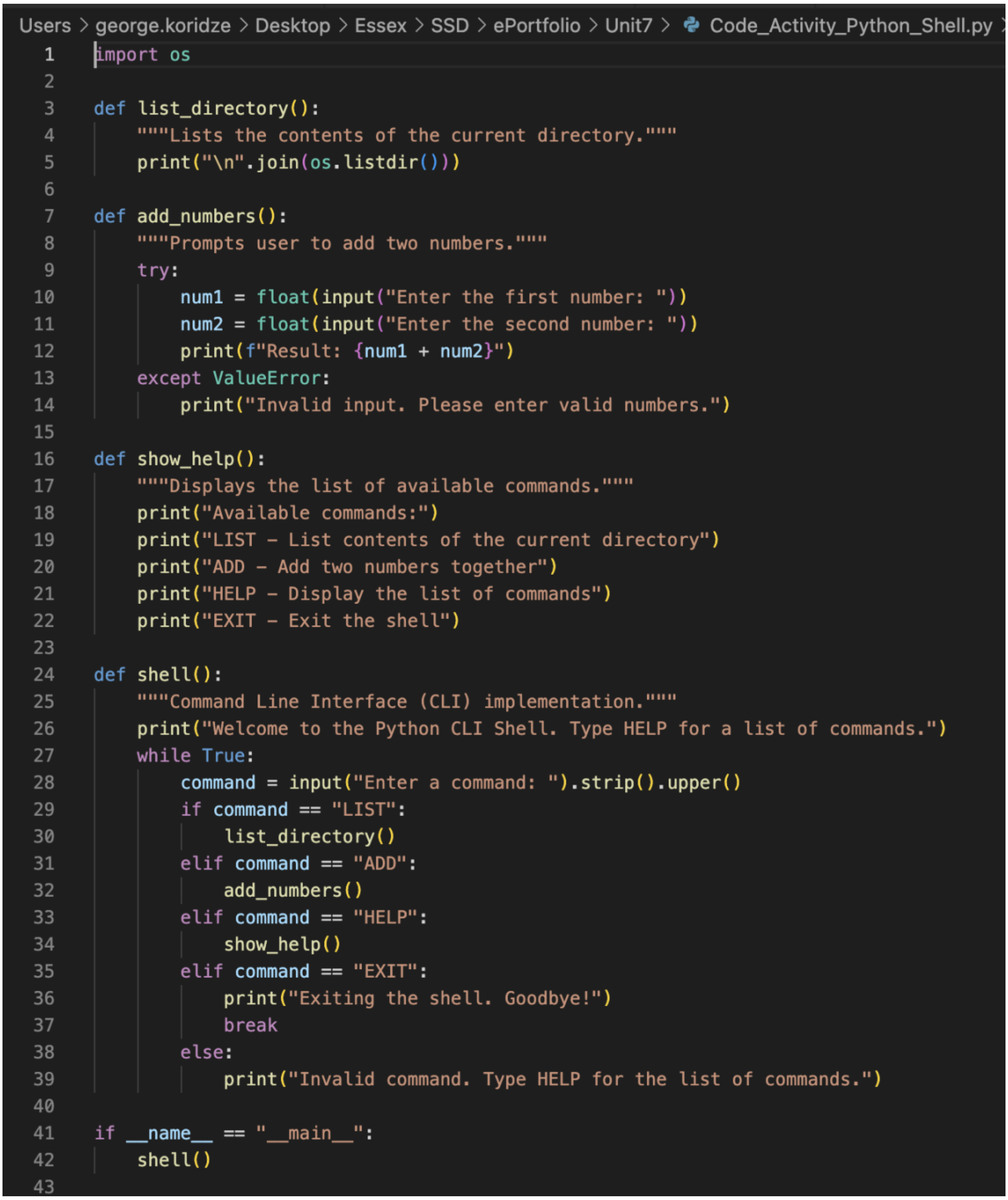
Exploring a simple Python shell

Code:



Users > george.koridze > Desktop > Essex > SSD > ePortfolio > Unit7 > Code_Activity_Python_Shell.py

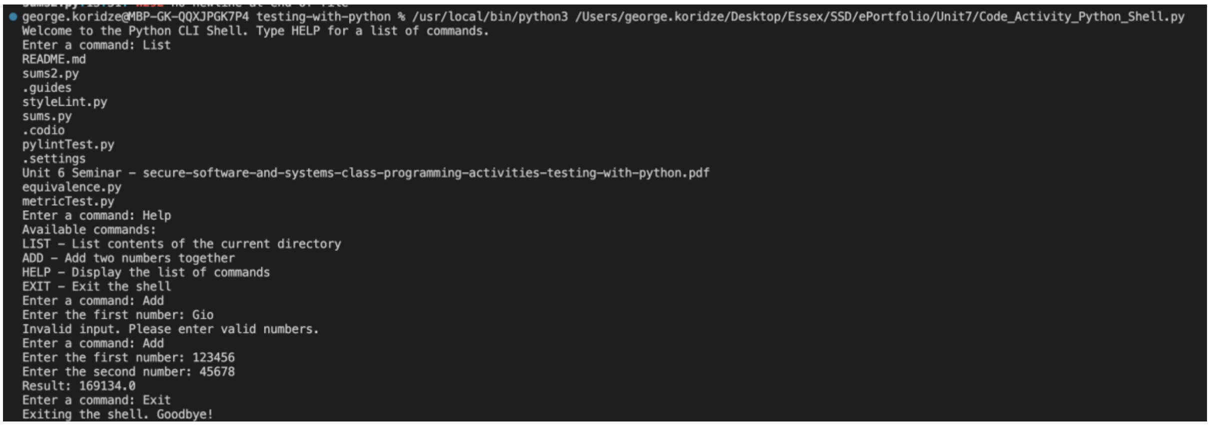
```python
import os

def list_directory():
    """Lists the contents of the current directory."""
    print("\n".join(os.listdir()))

def add_numbers():
    """Prompts user to add two numbers."""
    try:
        num1 = float(input("Enter the first number: "))
        num2 = float(input("Enter the second number: "))
        print(f"Result: {num1 + num2}")
    except ValueError:
        print("Invalid input. Please enter valid numbers.")

def show_help():
    """Displays the list of available commands."""
    print("Available commands:")
    print("LIST - List contents of the current directory")
    print("ADD - Add two numbers together")
    print("HELP - Display the list of commands")
    print("EXIT - Exit the shell")

def shell():
    """Command Line Interface (CLI) implementation."""
    print("Welcome to the Python CLI Shell. Type HELP for a list of commands.")
    while True:
        command = input("Enter a command: ").strip().upper()
        if command == "LIST":
            list_directory()
        elif command == "ADD":
            add_numbers()
        elif command == "HELP":
            show_help()
        elif command == "EXIT":
            print("Exiting the shell. Goodbye!")
            break
        else:
            print("Invalid command. Type HELP for the list of commands.")

if __name__ == "__main__":
    shell()

```

Run the shell using any Python interpreter. The shell will:

1. Respond to LIST by showing the contents of the current directory.

2. Use ADD to prompt for two numbers, calculate their sum, and display the result.
3. Display available commands with HELP.
4. Exit the shell when the EXIT command is entered.

Output:



```
sums2.py:13:31: W292 no newline at end of file
● george.koridze@MBP-GK-QQXJPGK7P4 testing-with-python % /usr/local/bin/python3 /Users/george.koridze/Desktop/Essex/SSD/ePortfolio/Unit7/Code_Activity_Python_Shell.py
  Welcome to the Python CLI Shell. Type HELP for a list of commands.
  Enter a command: List
  README.md
  sums2.py
  .guides
  styleLint.py
  sums.py
  .codio
  pylintTest.py
  .settings
  Unit 6 Seminar - secure-software-and-systems-class-programming-activities-testing-with-python.pdf
  equivalence.py
  metricTest.py
  Enter a command: Help
  Available commands:
  LIST - List contents of the current directory
  ADD - Add two numbers together
  HELP - Display the list of commands
  EXIT - Exit the shell
  Enter a command: Add
  Enter the first number: Gio
  Invalid input. Please enter valid numbers.
  Enter a command: Add
  Enter the first number: 123456
  Enter the second number: 45678
  Result: 169134.0
  Enter a command: Exit
  Exiting the shell. Goodbye!
```

Questions and Answers

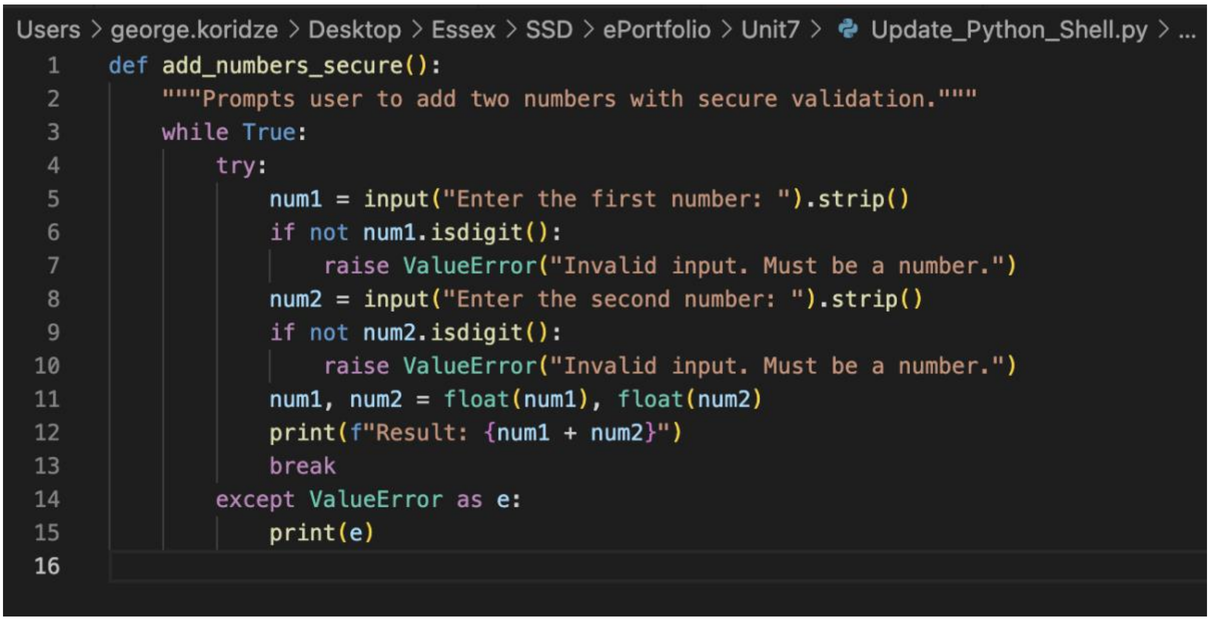
1. What are the two main security vulnerabilities with your shell?

- **Command Injection**: While the shell itself does not explicitly execute external commands, if additional commands (e.g., system calls) were added, user input could potentially exploit those commands.
- **Input Validation**: The ADD command does not sufficiently validate inputs. Malicious inputs could exploit the program or cause it to crash.

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

- Implement stricter input validation. For instance, limit the input to specific allowed commands and sanitize user inputs, ensuring they do not contain potentially harmful characters or unintended commands.

3. Pseudo-code example of changes to improve security: This is an improved version of the ADD function with enhanced validation:


Users > george.koridze > Desktop > Essex > SSD > ePortfolio > Unit7 > Update_Python_Shell.py > ...

```python
def add_numbers_secure():
    """Prompts user to add two numbers with secure validation."""
    while True:
        try:
            num1 = input("Enter the first number: ").strip()
            if not num1.isdigit():
                raise ValueError("Invalid input. Must be a number.")
            num2 = input("Enter the second number: ").strip()
            if not num2.isdigit():
                raise ValueError("Invalid input. Must be a number.")
            num1, num2 = float(num1), float(num2)
            print(f"Result: {num1 + num2}")
            break
        except ValueError as e:
            print(e)
```