



CoGrammar

SE PORTFOLIO SESSION 7



**SKILLS
FOR LIFE**

SKILLS BOOTCAMPS



Department
for Education

Software Engineering Lecture Housekeeping

- The use of disrespectful language is prohibited if asking a question. This is a supportive, learning environment for all – please engage accordingly! **(FBV: Mutual Respect.)**
- No question is 'silly' – **ask away!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Open Classes. You can submit these questions here: [Open Class Questions](#)

Software Engineering Lecture Housekeeping cont.

- For all **non-academic questions**, please submit a query:
www.hyperiondev.com/support
- Report a **safeguarding** incident:
www.hyperiondev.com/safeguardreporting
- We would love your **feedback** on lectures: [Feedback on Lectures](#)

Progression Criteria

✓ **Criterion 1: Initial Requirements**

- Complete 15 hours of Guided Learning Hours and the first four tasks within two weeks.

✓ **Criterion 2: Mid-Course Progress**

- Software Engineering: Finish 14 tasks by week 8.
- Data Science: Finish 13 tasks by week 8.

✓ **Criterion 3: Post-Course Progress**



- Complete all mandatory tasks by 24th March 2024.
- Record an Invitation to Interview within 4 weeks of course completion, or by 30th March 2024.
- Achieve 112 GLH by 24th March 2024.

✓ **Criterion 4: Employability**

- Record a Final Job Outcome within 12 weeks of graduation, or by 23rd September 2024.




How can you write a single line of text to a file?

- 
- A. `file.write(text)`
 - B. `file.addline(text)`
 - C. `file.writeline(text)`
 - D. `writeline(file, text)`
- 



Which mode will allow us to write to a file without overwriting the current data?

- A. 'r'
 - B. 'a'
 - C. 'w+'
 - D. 'w'
- 

Recap: File Output

Opening Files

- To open a file in Python, use the `open()` function specifying the file path and the desired mode.

File Modes

- File modes determine how a file is opened and what operations can be performed on it. We use 'w' to write to a file and 'a' to append to a file.

File Methods

- We can use different file methods to output to files such as `write()`, `writeline()` and `writelines()`.

Recap of Week 8: File Output

Open Files

```
file = open("my_file.txt", "r")  
content = file.read()  
file.close()
```

Open Files Using “with”

```
with open("my_file.txt", "r") as file:  
    content = file.read()
```


Gaming System

- **Background:** You've been tasked with designing and creating a gaming system called PlayMaster, which will be used in primary schools across the country as a teaching aid during computer lessons.
- **Challenge:** Create a gaming system which simulates a game of your choice.
- **Objective:**
 - The game should allow users to create new games, pause games, continue paused games and view past games.
 - Allow users to create profiles for the game that they have to use to log in and play.
 - Implement a scoring and ranking system that users can view to see their ranking.

Register New Users

We start by getting the user's username, password and password confirmation. We then check to see if the two password are the same. If they are, we add the new user to our user_password dictionary and write their details to the user text file.

```
new_username = input("New Username: ")
new_password = input("New Password: ")
confirm_password = input("Confirm Password: ")

if new_password == confirm_password:
    print("New user added")
    username_password[new_username] = new_password

    with open("user.txt", "w") as out_file:
        user_data = []
        for k in username_password:
            user_data.append(f"{k};{username_password[k]}")
        out_file.write("\n".join(user_data))
else:
    print("Passwords do not match")
```

Logging User In

We ask the user to enter their username and password. We first check if the user exists. If they do, we compare the password the user entered to the one in the user file. If both these conditions are met, we can login out the user.

```
username_password = {}

logged_in = False
while not logged_in:

    print("LOGIN")
    curr_user = input("Username: ")
    curr_pass = input("Password: ")
    if curr_user not in username_password.keys():
        print("User does not exist")
        continue
    elif username_password[curr_user] != curr_pass:
        print("Wrong password")
        continue
    else:
        print("Login Successful!")
        logged_in = True
```

Example Game

Here we have an example of a game to add to your program. A user can guess a number between 1 and 100 until they get the correct number.

```
import random

target_number = random.randint(1, 100)
guess = 0
attempts = 0

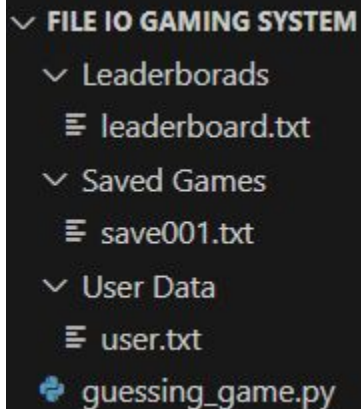
while guess != target_number:
    guess = int(input("Guess the number (between 1 and 100): "))
    attempts += 1
    if guess < target_number:
        print("Too low! Try again.")
    elif guess > target_number:
        print("Too high! Try again.")
    else:
        print(f"Congratulations! You guessed the number in {attempts} attempts.")
```

Saving Game State

- When trying to save a game like our Guessing game we have to determine what the game's state depends on.
- For the guessing game, we can see the state of the game depends on the number the user has to guess and the number of attempts they have taken to guess the number.
- To save a game we just have to store these to variables. When we resume, we can retrieve these two values and rebuild the previous game state.
- We can store the values of these variables in a text file to allow users to resume their games even after the program has been stopped.

Project File Structure

- You will be working with a lot of input and output files in this project.
- It would be a good idea to plan out your file's structure and layout.
- Create subdirectories for you text files such as user_data, saved_games, leaderboards.



```
▼ FILE IO GAMING SYSTEM
  ▼ Leaderboards
    ≡ leaderboard.txt
  ▼ Saved Games
    ≡ save001.txt
  ▼ User Data
    ≡ user.txt
  📄 guessing_game.py
```

A screenshot of a file explorer window with a dark background. The root directory is 'FILE IO GAMING SYSTEM', which is expanded to show three subdirectories: 'Leaderboards', 'Saved Games', and 'User Data'. Each subdirectory is expanded to show a single text file: 'leaderboard.txt' under Leaderboards, 'save001.txt' under Saved Games, and 'user.txt' under User Data. Below these subdirectories is a Python script file named 'guessing_game.py'.

Gaming System

Create a gaming system which simulates a game of your choice.

Here are some games to consider using for your program.

Guess the Number
Hangman
Rock, Paper, Scissors
Tic-tac-toe
Simple Quiz

Important features:

1. **Login:** Users have to log into an account before they can access and play the game. If a user does not have an account they should be able to register one.
2. **Menu:** Provide the user with a menu listing all the available option such as starting a new game, resuming a game, viewing the leaderboards, etc.
3. **Game:** Implement a game of your choice for the user to play. Remember to provide them with some sort of score for the leaderboards.
4. **Pause Game:** When the user wants to pause the game, the current state of the game should be saved for the user to access later when they choose to resume playing.
5. **History:** Allow the user to view a history of the games they have played and the scores they received.

Advanced

Challenge:

- Implement multiple games into your program for the user to play each game with it's own leaderboards and scores.

Summary

Opening Files

- ★ To open a file in Python, use the `open()` function specifying the file path and the desired mode.

File Modes

- ★ File modes determine how a file is opened and what operations can be performed on it.

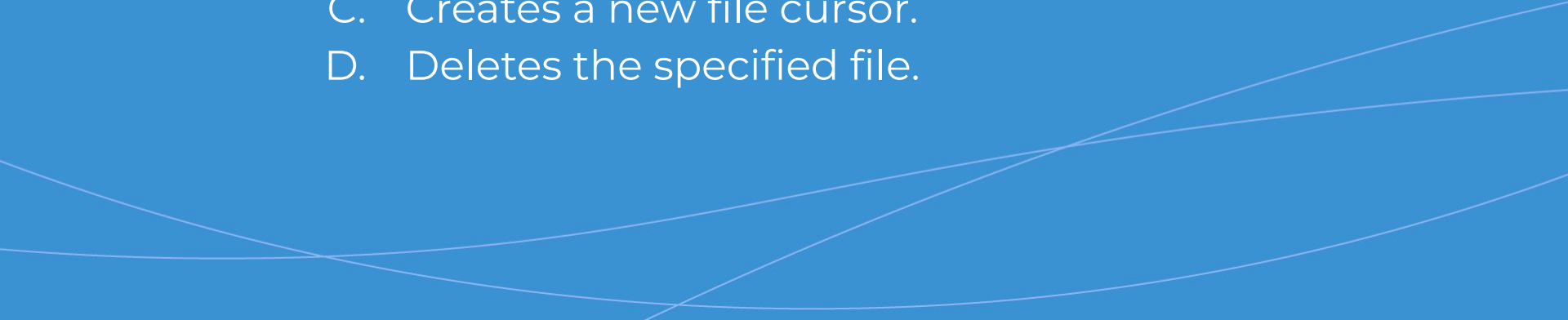
File Methods

- ★ Operations that allow us to manipulate and interact with files.



What does the seek() method in Python file handling do?



- A. Moves the file cursor to the end of the file.
 - B. Sets the file cursor to a specified position.
 - C. Creates a new file cursor.
 - D. Deletes the specified file.
- 



Which data type should your data be when calling `writelines()`?

- 
- A. String
 - B. Integer
 - C. Dictionary
 - D. List
- 



Questions and Answers

Questions around the Case Study





CoGrammar

Thank you for joining us

1. Take regular breaks
2. Stay hydrated
3. Avoid prolonged screen time
4. Practice good posture
5. Get regular exercise

“With great power comes great responsibility”
