



# CoGrammar

## SE PORTFOLIO SESSION 5



**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](http://www.hyperiondev.com/support)
- Report a **safeguarding** incident:  
[www.hyperiondev.com/safeguardreporting](http://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**

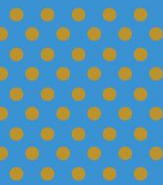
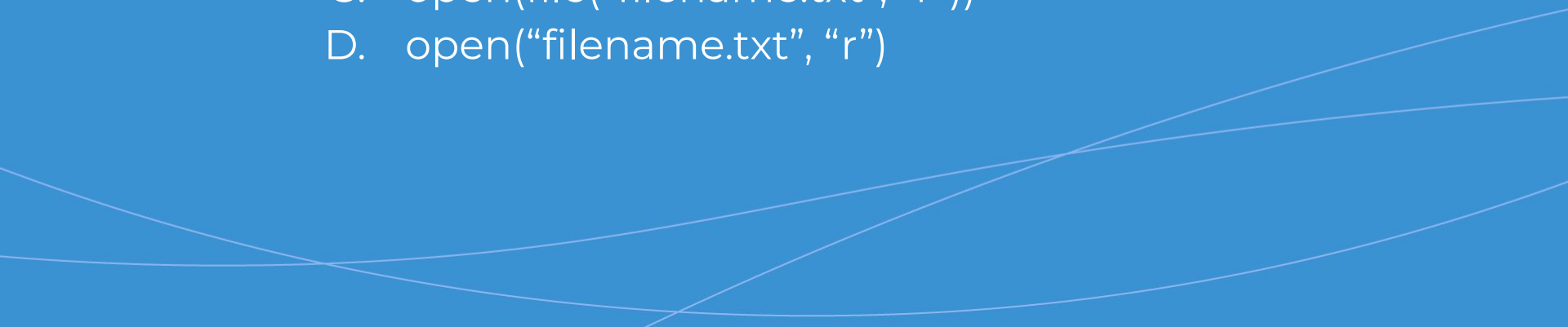
- Complete all mandatory tasks by 24th March 2024.
- Record an Invitation to Interview within 4 weeks of course completion, or by 30th March 2024.
- Achieve 112 GLH by 24th March 2024.

## ✓ **Criterion 4: Employability**

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

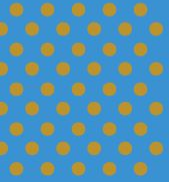



# How do you open a file in python?

- 
- A. `file("filename.txt", "r")`
  - B. `new file("filename.txt")`
  - C. `open(file("filename.txt", "r"))`
  - D. `open("filename.txt", "r")`
- 

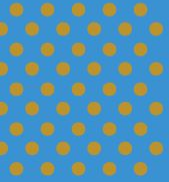
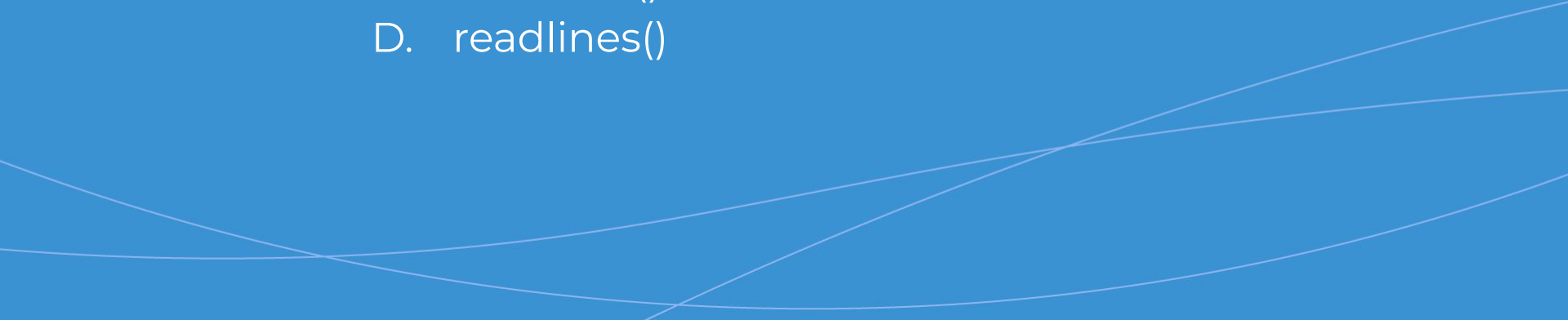


# What data type are the data retrieved from a text file?

- 
- A. string
  - B. int
  - C. complex
  - D. float
- 



# How do you retrieve the data from a text file as a single string value?

- 
- A. `readline()`
  - B. `read()`
  - C. `readable()`
  - D. `readlines()`
- 

# Recap: File Input

## 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.



## Recap: File Input

### 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()
```

# Log Management Tool

- **Background:** You are part of the software development team at Starship Logistics: a logistics company specialising in shipping goods globally. You have decided to create and implement a comprehensive log management system for tracking and analysing the vast amount of shipping-related logs that the company deals with.
- **Challenge:** Create a log management tool which reads shipping logs saved as text files and displays them to a user in a presentable and readable fashion.
- **Objective:**
  - Allow the user to type in the name of a log file they would like to view.
  - Read the data from the selected text file and format the output using string manipulation.
  - Print the log data to the console in a neat and readable format for the user.

# Reading Data From File

## Using String

```
text_data = ""  
for line in file:  
    text_data += line
```

## Using List

```
text_data = []  
for line in file:  
    text_data.append(line)
```

We can use a for loop to read our file line by line. We can then add each line to either a string or a list to store for later use.

# Reading Data From File

```
single_line = file.readline()  
text_data = file.read()  
text_data_list = file.readlines()
```

We can read a single line from a file using `readline()` or we can get all the text from the data and store it as a string using `read()`. `readlines()` will give us a list with each index containing one line of the file.

# Formatting Strings

```
log_string = "11/12/2023 14:30 | Order Placed | 512458 | London | Electronics"
split_string = log_string.split("|")

date_time = split_string[0]
log_type = split_string[1]
tracking_number = split_string[2]
location = split_string[3]
goods_type = split_string[4]

log_output = f"Date: {date_time}\nLOG: {log_type}\nTracking number: {tracking_number}"
log_output = log_output + f"\nLocation: {location}\nGoods: {goods_type}"

print(log_output)
```

Log\_string is an example of a line in our log text file. We can take this line and look for characters that will help us split the line into its respective values. We then split the string and store its value in a variable. We then format our string to make date more presentable and print it to the console.

# Formatting Strings

This is what the output of our string will look like.

```
Date: 11/12/2023 14:30  
LOG:  Order Placed  
Tracking number:  512458  
Location:  London  
Goods:  Electronics
```

# Log Management Tool

Create a log management tool that reads shipping logs saved as text files and displays them to a user in a presentable and readable fashion.

Here are some methods to consider using:

```
file.read()
```

```
file.readline()
```

```
file.readlines()
```

```
str.split()
```

```
str.format()
```

Important features:

1. **Menu:** Provide the user with a menu of what they can do on the program.
2. **Read logs:** User should be able to enter the name of the log text file they want to view.
3. **Log output formatting:** When a log is printed to the console it should be formatted in a user friendly and easy to read format.
4. **Error handling:** Remember to implement error handling within your program.
5. **User experience:** Think about the experience of the user during the flow of your program. Does the program expect the user to repeat unnecessary steps.

Advanced

Challenge:

- Have your program display the names of the log files the user can open and have them select the file from a list rather than entering a name.

# Summary

---

## Opening Files

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

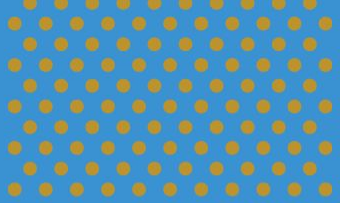
## File Modes

- ★ 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.







# Which mode is used to open a file for both reading and writing in Python?

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



# How do you close a file in Python?

- 
- A. `exit()`
  - B. `close()`
  - C. `end()`
  - D. `terminate()`
- 



# Questions and Answers

Questions around the Case Study

