Welcome to this CoGrammar Tutorial:
Text File IO and Exception-Handling

The session will start shortly...

**Questions? Drop them in the chat.
We'll have dedicated moderators
answering questions.**

CoGrammar

CoGrammar

Text File IO

September 2024

Software Engineering Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**

- No question is daft or silly - **ask them!**

- 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 Academic Sessions. You can submit these questions here: **Questions**

CoGrammar

Software Engineering Session Housekeeping cont.

- For all **non-academic questions**, please submit a query:

  www.hyperiondev.com/support

- We would love your **feedback** on lectures: **Feedback on Lectures**

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:
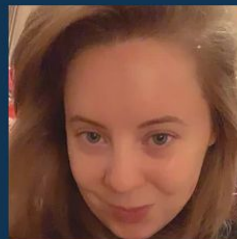

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Rafiq Manan

Charlotte Witcher

Nurhaan Snyman

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar   HyperionDev

Skills Bootcamp Progression Overview

To be eligible for a certificate of completion, students must fulfil three specific criteria. These criteria ensure a high standard of achievement and alignment with the requirements for the successful completion of a Skills Bootcamp.

✅ Criterion 1 - Meeting Initial Requirements

Criterion 1 involves specific achievements within the first two weeks of the program. To meet this criterion, students need to:

- Attend a minimum of 7-8 hours per week of guided learning (lectures, workshops, or mentor calls) within the initial two-week period, for a total minimum of 15 guided learning hours (GLH), by no later than 15 September 2024.

- Successfully complete the Initial Assessment by the end of the first 14 days, by no later than 15 September 2024.

CoGrammar

Skills Bootcamp
Progression Overview

✅ **Criterion 2 - Demonstrating Mid-Course Progress**

Criterion 2 involves demonstrating meaningful progress through the successful completion of tasks within the first half of the bootcamp.
To meet this criterion, students should:

- Complete 42 guided learning hours and the first half of the assigned tasks by the end of week 7, no later than 20 October 2024.

CoGrammar

Skills Bootcamp
Progression Overview

✅ **Criterion 3 - Demonstrating Post-Course Progress**

Criterion 3 involves showcasing students' progress after completing the course. To meet this criterion, students should:

- Complete all mandatory tasks before the bootcamp's end date. This includes any necessary resubmissions, no later than 22 December 2024.

- Achieve at least 84 guided learning hours by the end of the bootcamp, 22 December 2024.

CoGrammar

Poll

Which of the following is the correct way to handle a potential ZeroDivisionError in Python?

A
```
try:
    result = 10 / 0
catch ZeroDivisionError:
    print("Cannot divide by zero")
```

B
```
try:
    result = 10 / 0
except:
    print("An error occurred")
```

C
```
try:
    result = 10 / 0
except ZeroDivisionError:
    print("Cannot divide by zero")
```

D
```
if 10 / 0 == ZeroDivisionError:
    print("Cannot divide by zero")
else:
    result = 10 / 0
```

Which of the following code snippets is the correct one for opening the hello.txt file for **writing**?

a.
```
output_file = openFile("hello.txt", "w")
```

b.
```
output_file = open("hello.txt", "w")
```

c.
```
output_file = File("hello.txt", "w")
```

d.
```
output_file = open("hello.txt", "r")
```

CoGrammar

Learning Objectives & Outcomes

- Perform common file operations such as reading from and writing to text files.
- Use the 'with' statement to manage file resources efficiently, ensuring files are properly closed after operations.
- Apply best practices for managing resources to prevent memory leaks and other resource-related issues.
- Write try and except blocks to catch and handle exceptions.
- Differentiate between various built-in exceptions and handle them appropriately.
- Implement finally blocks to manage cleanup tasks such as closing files or releasing resources.
- Use custom exceptions to provide more informative error messages and improve error handling in their code

CoGrammar

Text File IO

CoGrammar

Introduction
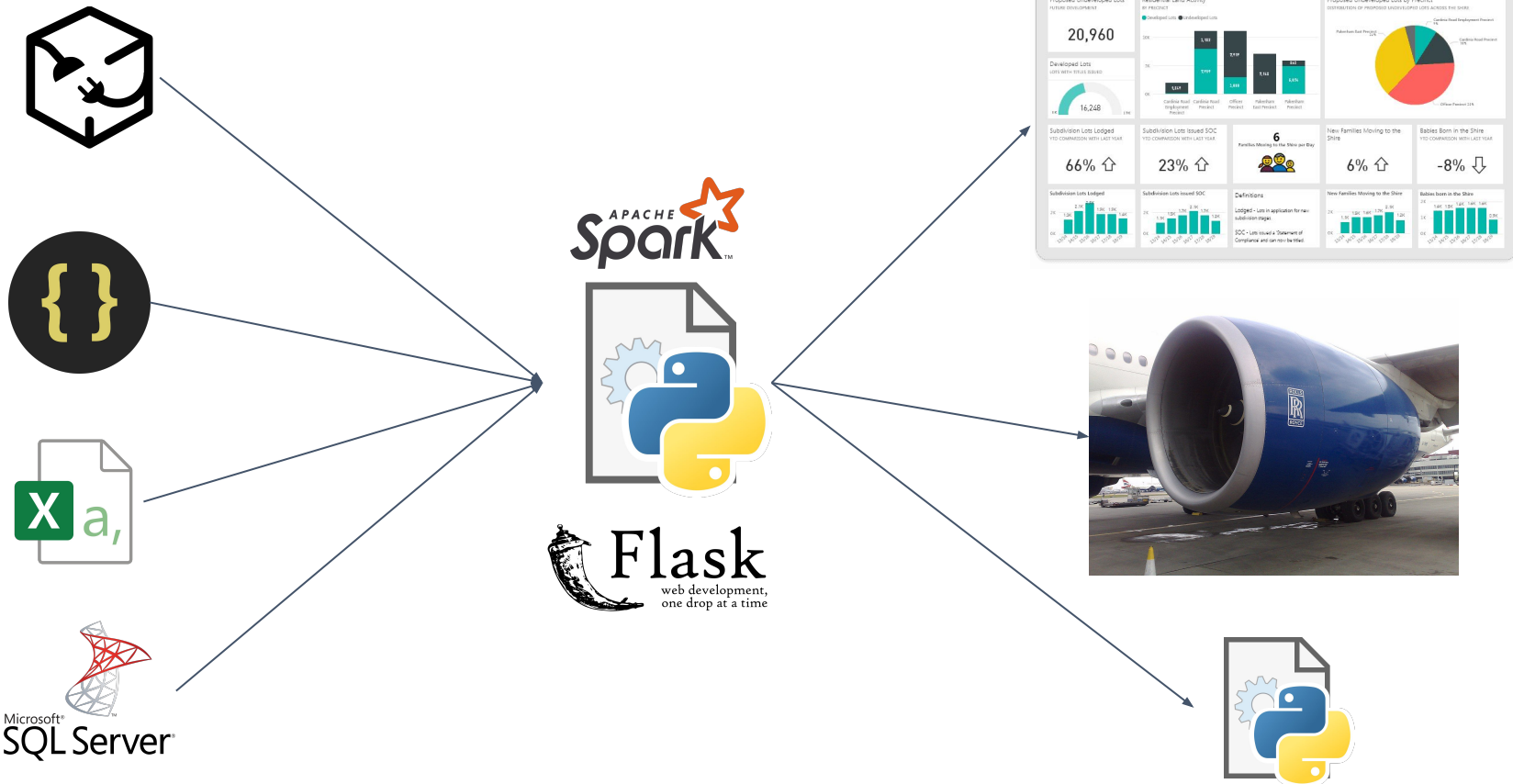
File Access Modes

Table 1: Python File modes

Mode	Description
'r'	Opens a file for reading.
'w'	Open a file for writing. If file does not exist, it creates a new file. If file exists it truncates the file.
'a'	Open a file in append mode. If file does not exist, it creates a new file.
'+'	Open a file for reading and writing (updating)

CoGrammar

Resource Management: Explicit Method

```python
file = open("filename.txt", "access_mode")
content = file.read()
file.close()
```

Text file (.txt)

opening

Read Only	r
Read and Write	r+
Write Only	w
Write and Read	w+
Append Only	a
Append and Read	a+

File Access Modes

closing

Must be closed to avoid issues like memory leaks

CoGrammar

Resource Management: Implicit Method

```
with open("filename.txt", "access_mode") as file:
    content = file.read()
```

opening

closing

Read Only	r
Read and Write	r+
Write Only	w
Write and Read	w+
Append Only	a
Append and Read	a+

Text file (.txt)

File Access Modes

Must be closed to avoid issues like memory leaks

CoGrammar

File Handling (Reading)

Read from a File Python
Methods

read()
Reads the entire
contents of the
file and returns it
as a string.

readline()
Reads a single
line from the file
and returns it as
a string.

readlines()
Reads all lines
from the file and
returns them as
a list of strings.

CoGrammar

File Handling (Writing)

Write to a File Python
Methods

write()
This method is used to write data to the file. It takes a string argument and adds it to the end of the file.

writelines()
This method writes a sequence of strings to the file. It takes a list of strings as an argument and writes each string to the file.

Let's take a short Break

CoGrammar

Custom Exceptions

CoGrammar

Exception Handling

- An **Error/Exception** is an unexpected event that interrupts the normal execution of a computer program, preventing it from achieving its intended outcome.

- **Exception handling** in Python allows you to gracefully manage errors that may occur during program execution, including when working with files.

CoGrammar

File Exception Handling

IsADirectoryError

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

FileNotFoundError

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

PermissionError

```python
with open("filename.txt", 'w') as file:
    content = file.read()
```

CoGrammar

Custom Exceptions in Python Using "raise"

- The **raise** keyword allows you to trigger exceptions in Python.

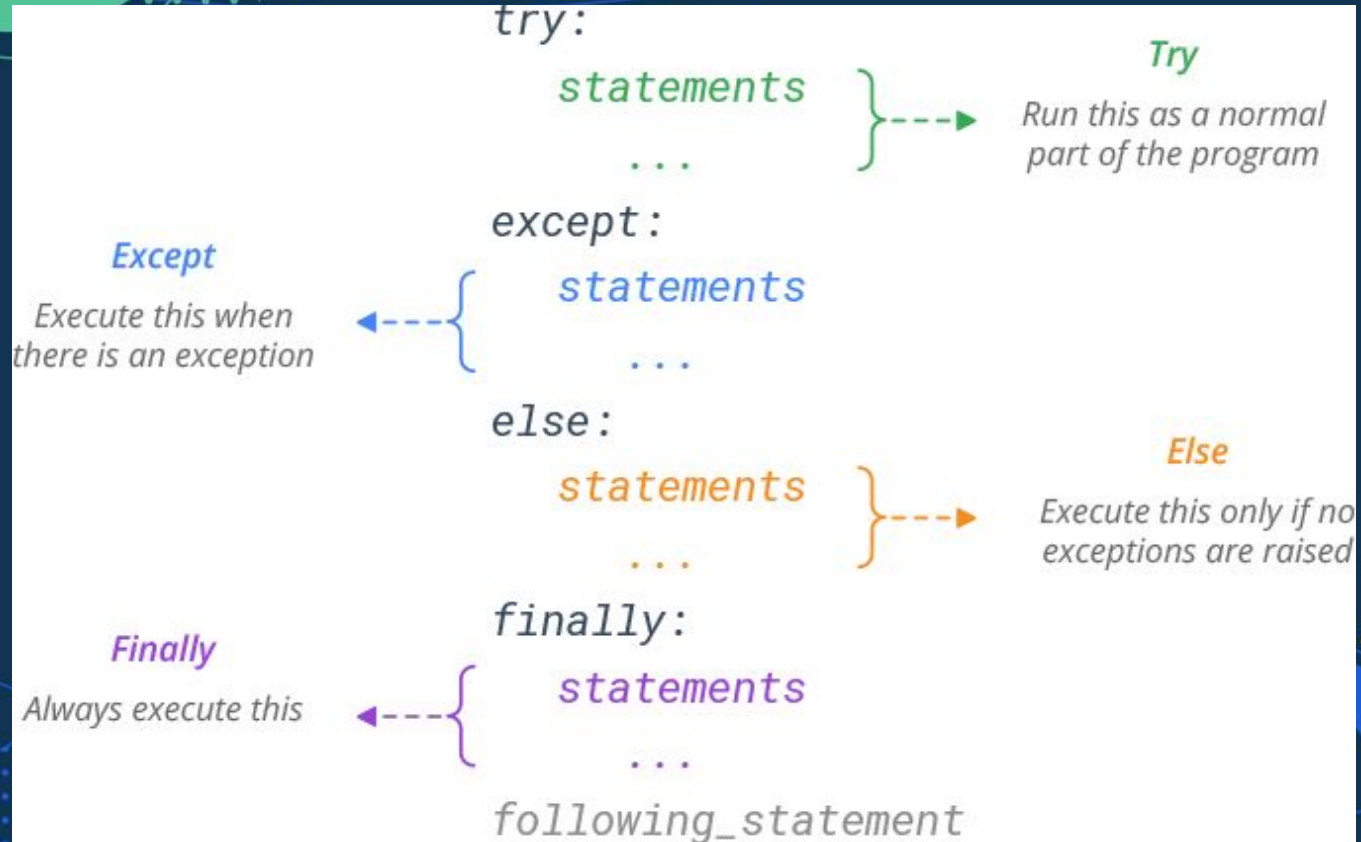- You can raise any built-in exception class to handle errors as needed.

```
raise ExceptionType("Custom error message")
```

- `ExceptionType`: Any valid built-in exception (e.g., `ValueError`, `TypeError`, `ValueError`, `FileNotFoundError`).

- `"Custom error message"`: Descriptive message for the raised exception.

CoGrammar

Custom Exceptions in Python Using "raise"

```python
def divide(a, b):
    if b == 0:
        raise ValueError("Cannot divide by zero")
    return a / b
try:
    result = divide(10, 0)
except ValueError as e:
    print(f"Error occurred: {e}")
```

Try / Except / Finally Structure


```
try:
    statements
    ...
```
Try
Run this as a normal part of the program

```
except:
    statements
    ...
```
Except
Execute this when there is an exception

```
else:
    statements
    ...
```
Else
Execute this only if no exceptions are raised

```
finally:
    statements
    ...
following_statement
```
Finally
Always execute this

CoGrammar

Lesson Conclusion and Recap

- **File Operations in Python**:
  - Opening and closing files using open() and close(), and the advantages of using the with statement for automatic file management.
- **Reading and Writing to Files**:
  - Techniques for reading from (read(), readline(), readlines()) and writing to files (write(), writelines()), and the different file modes (read, write, append).
- **Exception Handling Basics**:
  - The structure of try, except, and finally blocks to catch and manage errors, ensuring programs handle unexpected situations gracefully.
- **Specific Exception Management**:
  - How to catch and handle specific exceptions like FileNotFoundError, and how to raise exceptions using the raise keyword for custom error handling.
- **Best Practices for File I/O and Error Handling**:
  - Importance of resource management (e.g., always closing files), avoiding silent failures, and writing readable, maintainable code when dealing with exceptions.

CoGrammar

Tutorial: ATM Simulator

- **Objectives**: Create a program that simulates an ATM process. The customer can withdraw or deposit money based on their needs.

- **Steps to Implement**:
  - Allow the customer to input a withdrawal or deposit amount.
  - Ensure the withdrawal amount is valid (a positive number and not exceeding the balance)
  - Ensure the deposit amount is valid (a positive number).
  - Update the customer's balance after each transaction (withdrawal or deposit).
  - Log each transaction (successful or failed) into a transaction log file, including details like the transaction type, amount, remaining balance, and timestamp.
  - Provide the option to view the transaction log.

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar