Welcome to this CoGrammar Lecture:

Text File I/O

The session will start shortly...

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

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** throughout this session, should you wish to ask any follow-up questions.
- 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**

Software Engineering Session 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**

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

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

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

CoGrammar HyperionDev


SKILLS FOR LIFE
SKILLS BOOTCAMPS
Department for Education
CoGrammar
Text File IO

Poll

What will be the output of the following code?

```
def multiply_by(n):
    def multiply(x):
        return x * n
    return multiply


multiply_by_2 = multiply_by(2)
multiply_by_3 = multiply_by(3)
print(multiply_by_2(multiply_by_3(4))
)
```

a. 6

b. 12

c. 24

CoGrammar

Poll

What will be the output of the following code?

```
from functools import reduce

nums = [1, 2, 3, 4, 5, 6]
mapped_list = map(lambda x: x + 2, nums)
result = reduce(lambda x, y: x + y, filter(lambda x: x % 2 == 0, mapped_list))
```

a. 21

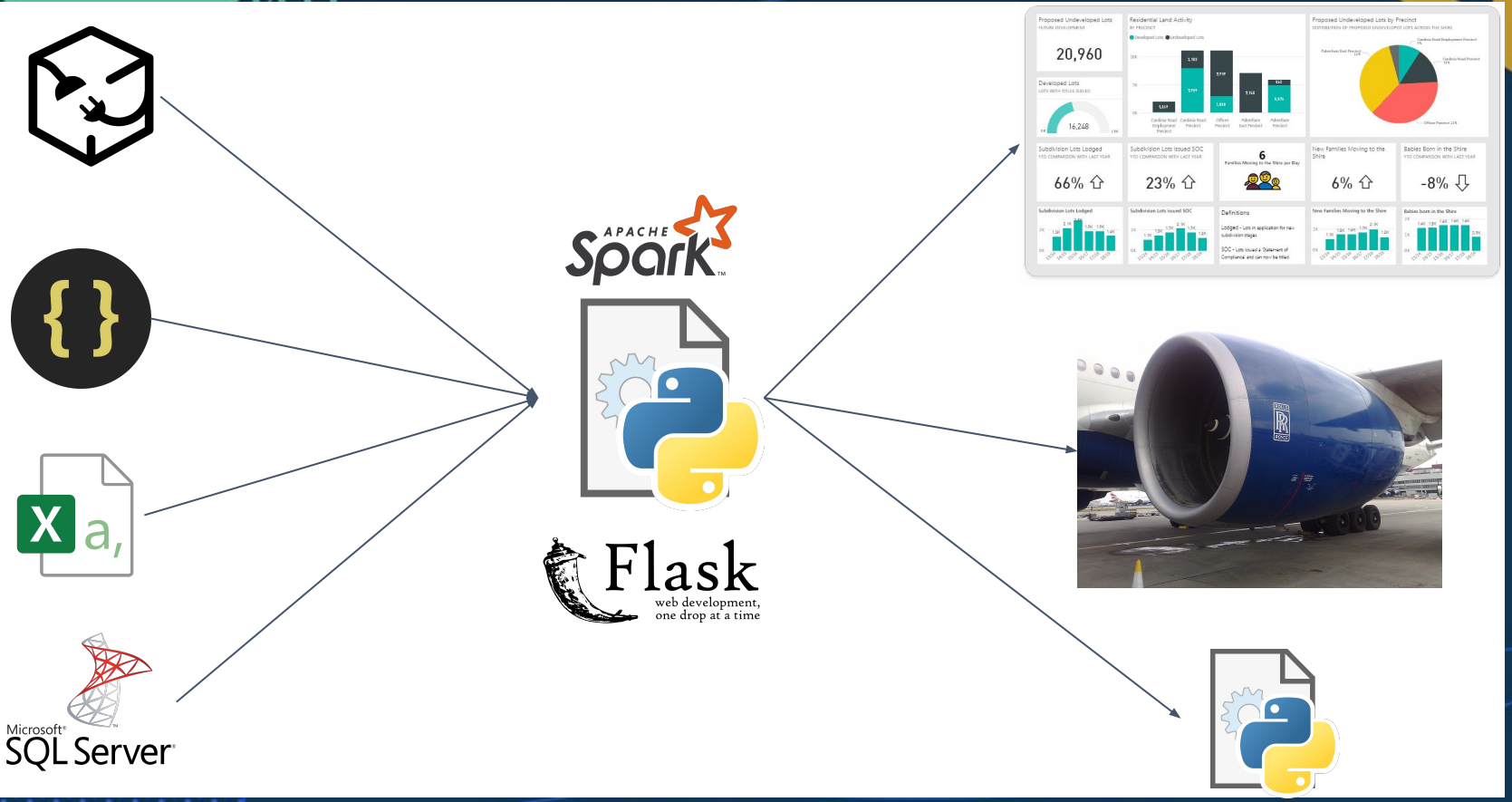
b. 18

c. 15

Learning Objectives & Outcomes

- Demonstrate the basics of file handling in Python, including opening, reading, writing, and closing files.
- Perform common file operations such as reading from and writing to text files.
- Handle file-related exceptions to ensure robust file operations.
- Understand the importance of resource management in programming.
- 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.

Introduction



What is File I/O ?

- **File I/O** stands for **File Input/Output** operations involving files.
- It refers to the process of reading data from files (input) or writing data to files (output) using a computer program.
    - In simpler terms, file I/O is all about your program interacting with files: either taking in information from them or putting information into them. It's like the communication link between your program and the outside world of files

Understanding External Sources

They couldn't play soccer because they couldn't see the ball. They didn't want to go inside and play a game because it was a beautiful springtime night and they liked being ...[1]


```
id,name price
1,Laptop 1200
2,Smartphone 800
3,Headphones 200
4,Mouse 50
5,Monitor  1000
6,Desk 600
7,Speakers 400
```

```
{
    "track_id": "3"
    "name": "Inde"
    "artist": "Bucie"
    "album": "Inde"
    "duration_ms":
373000
    "popularity": 3
}
```

CoGrammar

[1] HIDE AND SEEK | Free Children's book from Monkey Pen

File Manipulations

- **File modes**
  - Specifications used when opening a file.
  - Indicate the intended operation (reading, writing, appending, or a combination).
  - Determine how the file will be accessed.
- **File handling**
  - Process of working with files on computer storage.
  - Allows reading from, writing to, and manipulating files.
  - Python provides built-in functions and methods for file operations.

File Handling

- **File I/O in Python:**
  - Performed using the `open()` function.
  - Utilizes various methods associated with file objects.
- **Opening Files:**
  - The `open()` function opens a file and returns a file object.
  - Requires specifying:
    - File name/path.
    - Mode (e.g., 'r' for reading, 'w' for writing, 'a' for appending).

Working with Files in Python: I/O

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

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

Text file (.txt)

CoGrammar

Resource Management: Implicit Method

- The with statement is used for resource management in Python.
- It ensures that resources are properly cleaned up after use, even if an error occurs.

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

Resource Management: Explicit Method

- **Explicit file handling:**
  - Manual opening and closing of files.
  - Uses `open()` function for opening.
  - Uses `close()` method for closing.

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

File Access Modes (r)

- Reading from Text Files: You can read text from a file using the open() function with the mode 'r'

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

File Access Modes (w)

- Writing to Text Files: You can write text to a file using the open() function with the mode 'w'

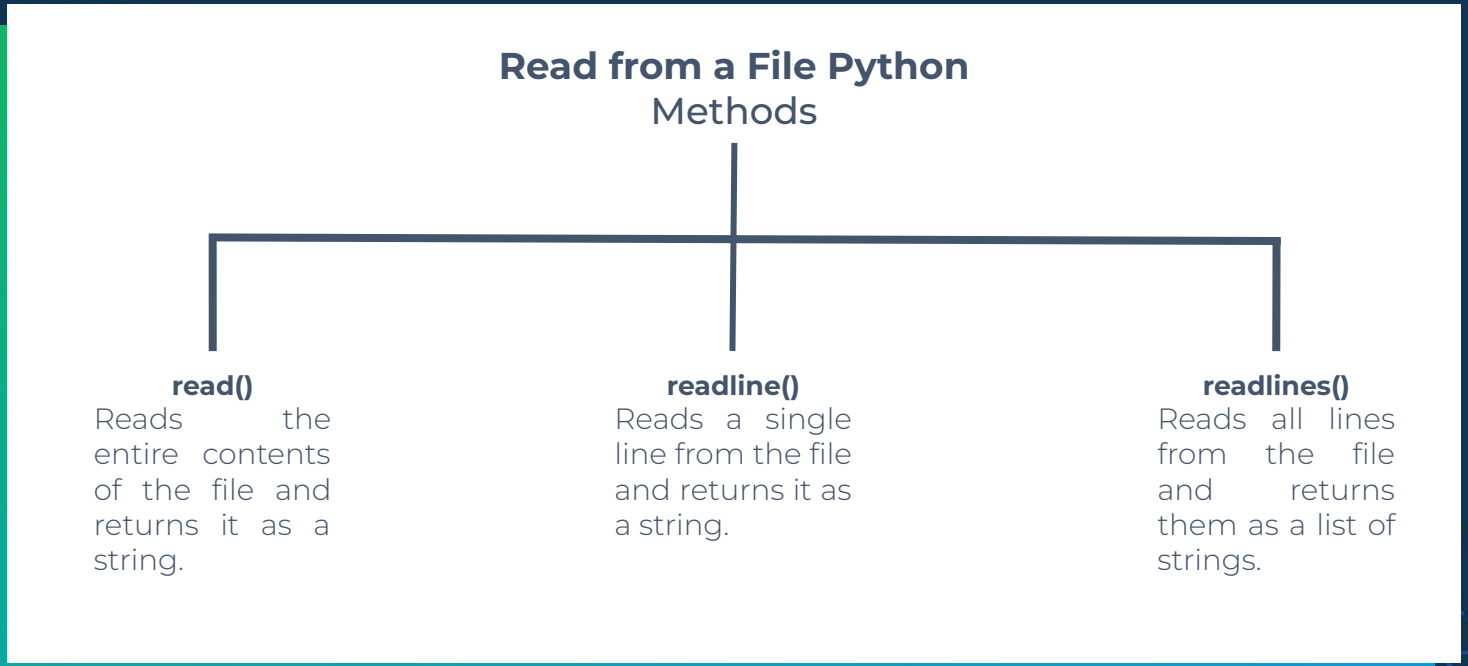
```
with open('filename.txt', 'w') as file:
    file.write("Hello, world!")
```

File Access Modes (a)

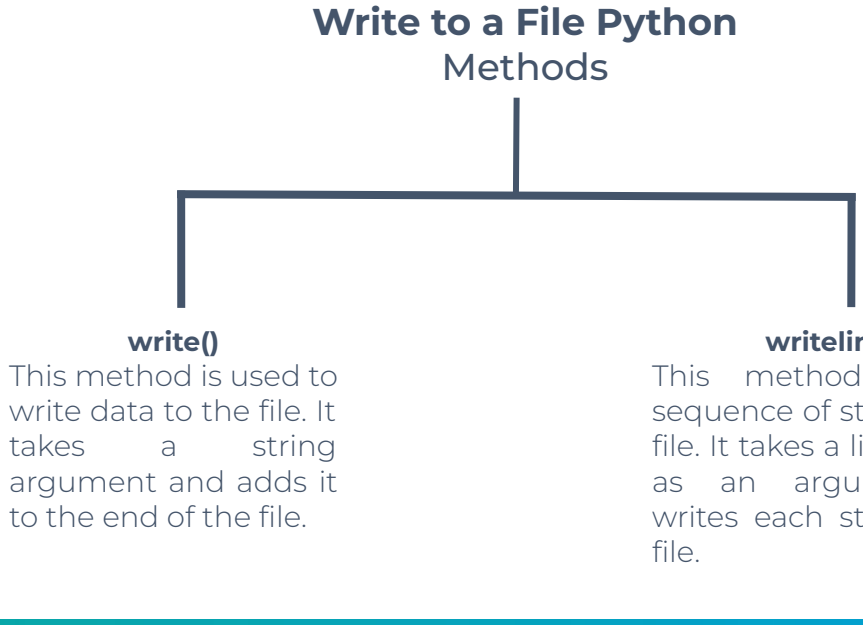
- Appending to Text Files: You can append text to an existing file using the open() function with the mode 'a'

```
with open('filename.txt', 'a') as file:
    file.write("\nThis is a new line.")
```

File Handling (Reading)



File Handling (Writing)



Error Handling

CoGrammar

Exception Handling

- **Error/Exception**
  - Unexpected events that interrupt normal program execution.
  - Prevent the program from achieving its intended outcome.
- **Exception handling**
  - Allows graceful management of errors during program execution.
  - Applies to various situations, including file operations.

File Exception Handling

IsADirectoryError

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

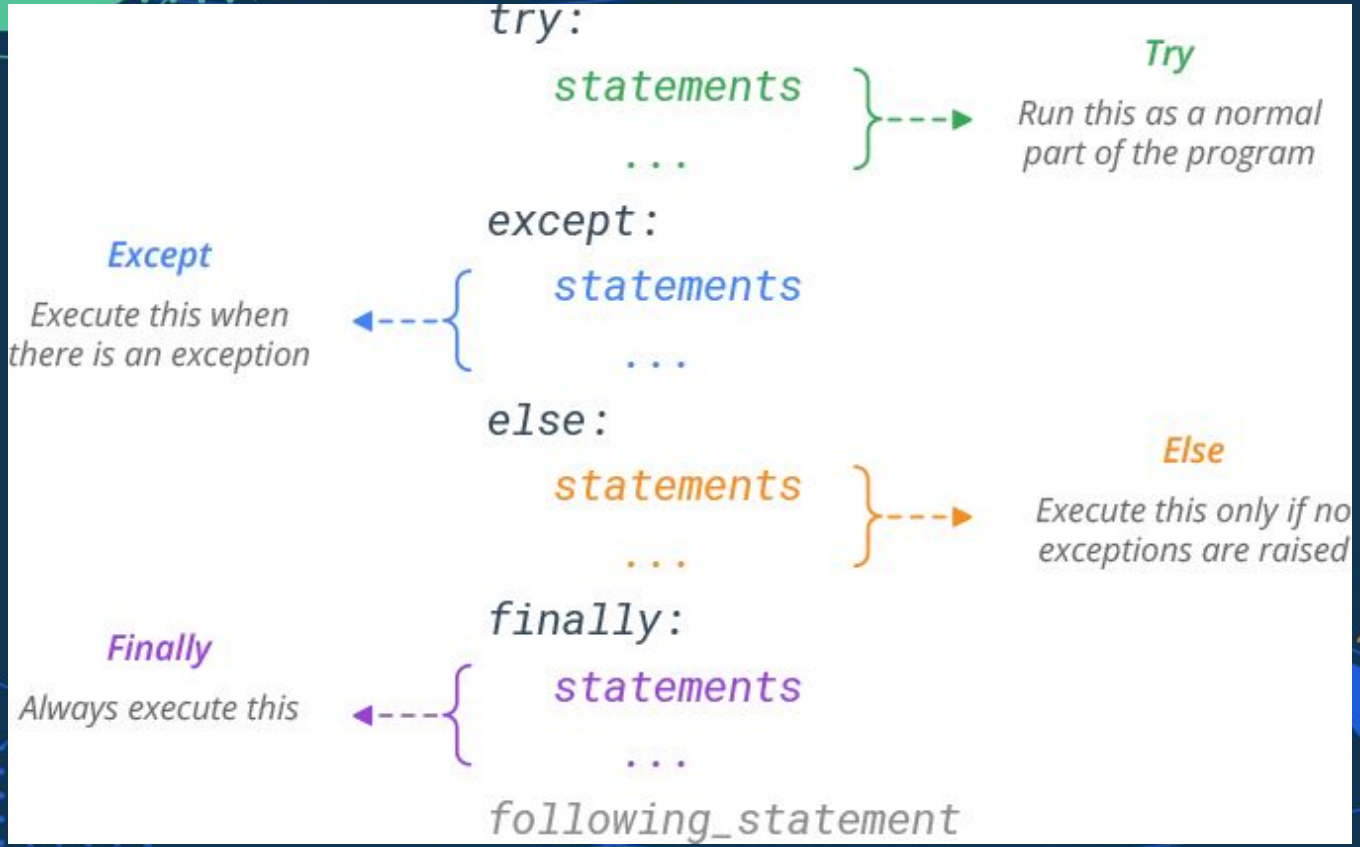
FileNotFoundError

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

PermissionError

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

Try / Except / Finally Structure



Try / Except Example

- You can wrap file I/O operations inside a try block and catch specific exceptions using except blocks.

```
try:
    with open('file.txt', 'r') as file:
        content = file.read()
    print(content)
except FileNotFoundError:
    print("File not found!")
except PermissionError:
    print("Permission denied to open the
file!")
except IOError as e:
    print(f"An I/O error occurred: {e}")
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

Try / Except / Finally

- We can also use a finally block to ensure that certain cleanup actions are always performed, regardless of whether an exception occurred or not.

```
try:
    file = open('file.txt', 'r')
    content = file.read()
    print(content)
except FileNotFoundError:
    print("File not found!")
finally:
    file.close()  # Ensure file is closed even if an exception occurs
```

Poll

What will happen when this code is executed?

```
try:
    with open('data.txt', 'r') as file:
        content = file.read()
except FileNotFoundError as e:
    print("File missing")
finally:
    print("Operation complete")
    file.close()
```

1. A NameError will be raised in the finally block because 'file' is out of scope
2. "File missing" and "Operation complete" will be printed
3. Only "Operation complete" will be printed

Poll

Given a file 'data.txt' with content "1,2,3\n4,5,6\n", what's the output?

```
with open('data.txt', 'r') as f:
    lines = f.readlines()
    result = [line.strip().split(',') for line in
lines]
print(result[0][1])
```

a. 1
b. 2
c. '2'

CoGrammar

Lesson Conclusion and Recap

- **File Operations**
  - Open/close, read/write files (text, CSV, JSON)
  - File modes: read, write, append
  - open(), read(), write(), with statement
- **Working with External Data Formats**
  - Handling text
- **Error Handling with Exceptions**
  - Importance of error handling
  - Common exceptions: FileNotFoundError, PermissionError, IsADirectory
  - try, except, else, finally blocks

Practical: Building a Word Counter Application

- **Problem:** Build a Python application to analyze text files, including:
  - Word frequency counting.
  - Stop word exclusion (e.g., "the," "a," "is").
  - Punctuation and case handling.
- Solution: We'll build a Python program that:
  - Read a text file.
  - Remove stop words and punctuation.
  - Count word frequencies.
  - Use file I/O for input/output.
  - Use dictionaries for efficient storage.
  - Handle file/permission errors.

Follow-up Activity

3. Write the Results to a New File:
   - Write a function `write_results(filename, average, highest, lowest)` that writes the average, highest, and lowest grades to a new text file.
4. Main Program:
   - Combine the functions in a main block:
     - Use `read_grades` to read the grades from a file named `grades.txt`.
     - Use `process_grades` to calculate the necessary statistics.
     - Use `write_results` to save the results in a file named `results.txt`.
   - Example Files:
     - Input File (**`grades.txt`**):
       `85`
     - `90`
     - `78`
     - `92`
     - `88`
   - Output File (**`results.txt`**):
     - `Average Grade: 86.6`
     - `Highest Grade: 92`
     - `Lowest Grade: 78`

Follow-up Activity

1. **Submission**: Just make sure that you have the output provided above. This is not tied to any of your tasks.
2. Use any method available to you. As long as you understand the process.

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar