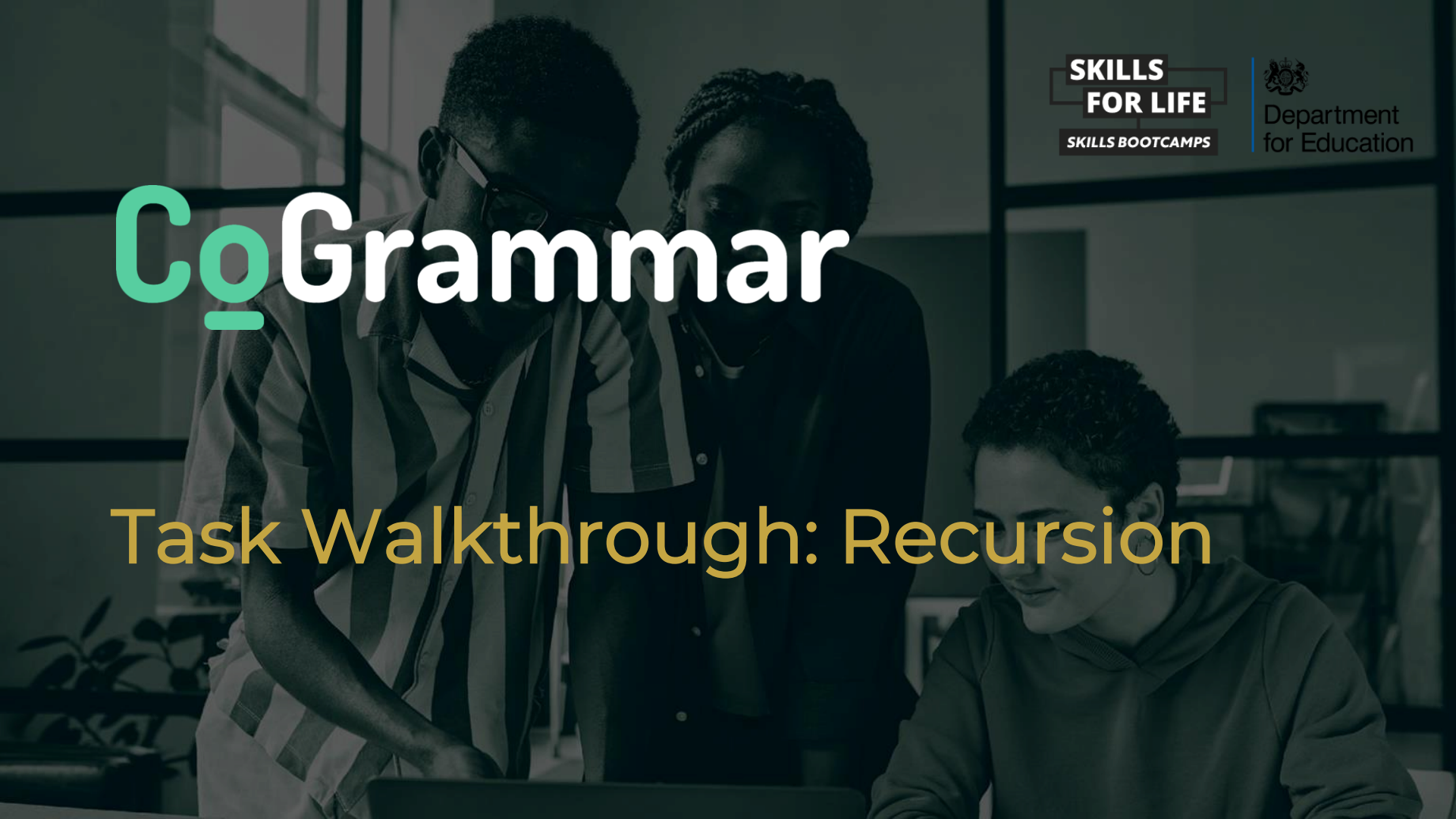
Welcome to this CoGrammar Task Walkthrough

The session will start shortly...

Questions? Drop them in the chat.



SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

Task Walkthrough: Recursion

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

Learning Outcomes

- Break down problems into recursive sub problems and implement base and recursive cases effectively.
- Transfer learnings to complete the Recursion tasks.

Recursion

- **Definition**: A technique where a function calls itself in order to solve a problem.
- **Base Case**: The condition under which the recursion stops; it prevents infinite recursion.
- **Recursive Case**: The part of the function that includes the recursive call; it breaks the problem down into smaller sub problems.
- **Stack Overflow**: Occurs when there are too many recursive calls without hitting a base case, exhausting the call stack.

Recursion

- **Advantages**: Simplifies code for problems that have a natural recursive structure, leading to cleaner and more understandable solutions.
- **Disadvantages**: Can lead to increased memory usage due to function call overhead and risk of stack overflow if not designed carefully.

Recursion Task Walkthrough: Auto-graded Task 1

CoGrammar

Auto-graded task 1

Follow these steps:

1. Create a file named **sum_recursion.py**.
2. Define a function which takes two arguments:
    a. A list of integers.
    b. A single integer that represents an index point.
3. The single integer will represent the index point up to which the function should sum all the numbers in the list.
    a. **Note:** List indices start at 0. The number at the specified index should be included in the calculation.
4. The function is required to sum all the numbers in the list up to and including the given index point.
5. The function should calculate the sum using recursion as opposed to using loops.

Examples of input and output:

```
adding_up_to([1, 4, 5, 3, 12, 16], 4)
=> 25
```
=> adding the numbers all the way up to index 4 (1 + 4 + 5 + 3 + 12)

Recursion Task Walkthrough: Auto-graded Task 2

CoGrammar

Auto-graded task 2

Follow these steps:

1. Create a file named **largest_number.py**.
2. Define a function that takes a single argument:
    a. A list of integers.
3. Within the function, implement logic to find the largest number in the list.
4. The function should return the largest number found in the list.
    a. **Note:** The problem must be solved using recursion without using loops.
    b. **Additional note:** The solution should not use built-in functions such as `max()`.

Examples of input and output:

```
largest_number([1, 4, 5, 3])
=> 5
largest_number([3, 1, 6, 8, 2, 4, 5])
=> 8
```

Be sure to place files for submission inside your **task folder** and click "**Request review**" on your dashboard.

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar