




Welcome to the CoGrammar

Skills Bootcamp: Functions and Error Handling

The session will start shortly...

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



Cyber Security 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](#)

Cyber Security 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](#)
- Find all the lecture **content** in you [Lecture Backpack](#) on GitHub.
- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

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



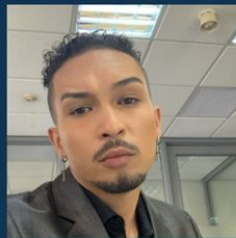
Nurhaan Snyman



Rafiq Manan



Ronald Munodawafa



Tevin Pitts

Scan to report a
safeguarding concern



or email the Designated
Safeguarding Lead:
Ian Wyles

safeguarding@hyperiondev.com

Learning Objectives & Outcomes

- Design Python programs that integrate functions and custom error handling to manage both normal operations and edge cases effectively.
- Write Python functions that include error handling to create robust programs that manage unexpected situations.
- Describe how functions and error handling (try-except blocks) work, including the use of different function parameters and handling common errors.

Stay Safe Series:

Mastering Online Safety One week at a Time

While the digital world can be a wonderful place to make education and learning accessible to all, it is unfortunately also a space where harmful threats like online radicalization, extremist propaganda, phishing scams, online blackmail and hackers can flourish.

As a component of this BootCamp the ***Stay Safe Series*** will guide you through essential measures in order to protect yourself & your community from online dangers, whether they target your privacy, personal information or even attempt to manipulate your beliefs.

Patch it Up:

The Importance of Regular Software Updates

- **Security Enhancements:** Regular updates patch vulnerabilities that hackers can exploit, protecting your devices and personal information from cyber threats.
- **Performance Improvements:** Updates often include bug fixes and optimizations that improve the speed, functionality, and stability of your software.

CoGrammar

Skills Bootcamp: Functions and Error Handling

October 2024

Functions & Error Handling.

What do you think would happen if a program you wrote encountered an unexpected input or error, like trying to divide by zero? How could you design your code to not only perform specific tasks but also handle such errors gracefully?



Polls

Please have a look at the poll notification and select an option.

What is the purpose of a try-except block in python?

- A. To ignore errors in code
- B. To test code functionality
- C. To handle exceptions and prevent program crashes
- D. To log errors to a file

Polls

Please have a look at the poll notification and select an option.

What happens when you don't include a return statement in a python function?

- A. The function returns None by default
- B. The function will cause an error
- C. The function will return the last variable used
- D. The function will not execute

Error Handling Practice: Division Function

- Write a function `safe_divide(a, b)` that takes two numbers and returns their division. If division by zero is attempted, handle the error and return a message like "Division by zero is not allowed."

Palindrome Checker

Write a function `is_palindrome(s)` that checks if a given string is a palindrome (reads the same forwards and backwards). Return `True` if it is, otherwise return `False`.

Let's take a break



Simple Calculator

Write a Python function `calculator(operation, a, b)` that takes three arguments:

- `operation`: a string that can be "add", "subtract", "multiply", or "divide".
- `a`: the first number (float or integer).
- `b`: the second number (float or integer).

The function should return the result of the appropriate arithmetic operation. For division, ensure that you handle division by zero by using error handling.

Temperature Converter

Write a Python function that converts a temperature from Celsius to Fahrenheit and handles input errors.

- Take a single parameter, `celsius`, which represents the temperature in Celsius.
- Use a try-except block to:
 - Catch and handle `TypeError` if the input is not a number, returning "Invalid input. Please enter a number."

Summary

- Combining functions with effective error handling creates robust applications that can manage both normal operations and edge cases efficiently.
- Proper error handling using try-except blocks within functions helps prevent crashes and manage unexpected situations gracefully.
- Explicitly raising exceptions within functions enables developers to signal issues that cannot be resolved locally, providing clearer feedback to users.

Polls

Please have a look at the poll notification and select an option.

When should you use error handling within a function?

- A. Only when you expect input errors
- B. In all functions, regardless of input
- C. When there is a possibility of runtime errors
- D. Error handling is unnecessary

Polls

Please have a look at the poll notification and select an option.

What is the best way to ensure your function can handle invalid input, such as passing a string when an integer is expected?

- A. Write a separate function to check inputs
- B. Use a try-except block to catch type errors within the function
- C. Ignore input validation.
- D. Set default parameter values

Questions and Answers



Thank you for attending



Department
for Education

CoGrammar

