



CoGrammar

SE PORTFOLIO SESSION 3



**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
- Report a **safeguarding** incident:
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**

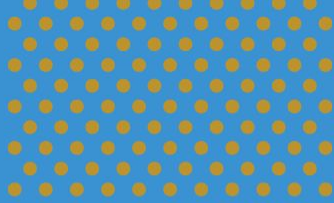
- Software Engineering: Finish 14 tasks by week 8.
- Data Science: Finish 13 tasks by week 8.

✓ **Criterion 3: Post-Course Progress**

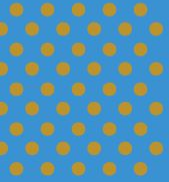
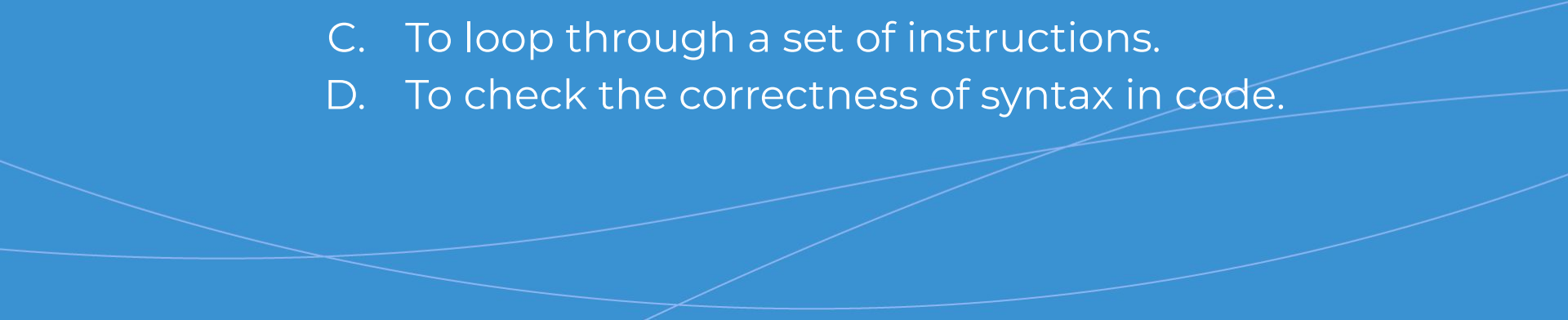
- Complete all mandatory tasks by 24th March 2024.
- Record an Invitation to Interview within 4 weeks of course completion, or by 30th March 2024.
- Achieve 112 GLH by 24th March 2024.

✓ **Criterion 4: Employability**

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



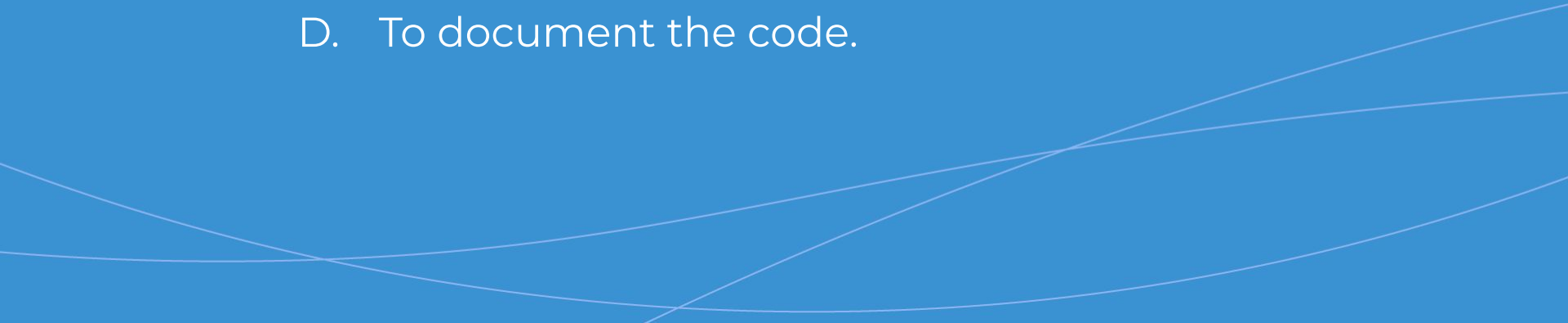
What is the purpose of the try-except block in Python?

- 
- A. To improve the speed of code execution.
 - B. To handle exceptions and prevent program crashes.
 - C. To loop through a set of instructions.
 - D. To check the correctness of syntax in code.
- 



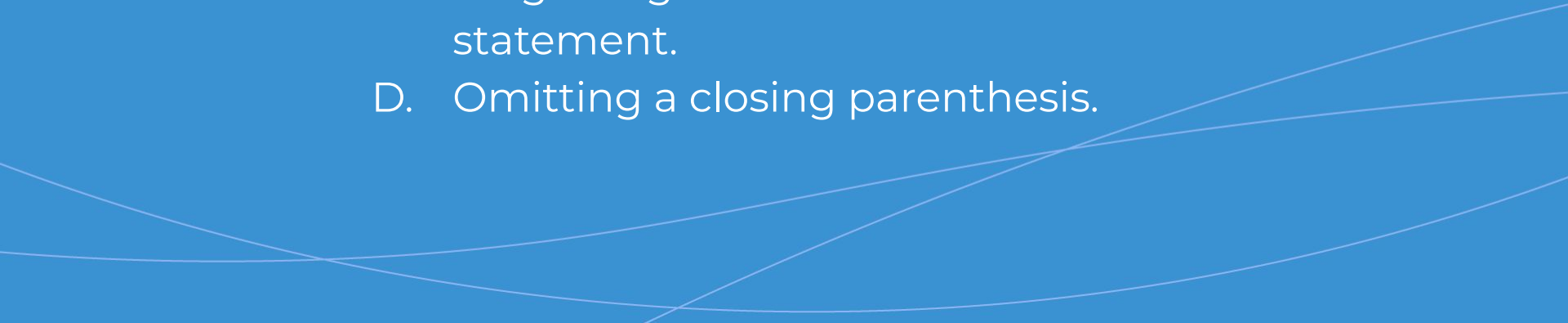
Why is input validation important in software development?



- A. To enhance the visual interface of applications.
 - B. To prevent invalid or malicious data input.
 - C. To increase the execution speed of the program.
 - D. To document the code.
- 



Which of the following is an example of a runtime error in Python?

- A. Misspelling a keyword.
 - B. Trying to divide by zero.
 - C. Forgetting a colon at the end of an if statement.
 - D. Omitting a closing parenthesis.
- 

Recap of Week 3: Defensive Programming

Conditional Statements

- Use conditional statements for input validation and error handling

Try-Except

- Use try-except blocks to handle anticipated exceptions and protect you application from runtime errors

Error Types

- Recognise the types of errors that can occur. Syntax, runtime, and logical.

CareerCrafter

- **Background:** You have been asked to build an program that validates the personal data of a user's application.
- **Challenge:** Allow a user to enter all their personal information for an application and validates the input according to a specific criteria.
- **Objective:** Use defensive programming to ensure the user input is valid:
 - Use conditional statements for input validation.
 - Use try-except blocks to gracefully manage unexpected exceptions
 - Recognise and address various error types.

Validation of User Email

```
user_email = input("Please enter your location: ")
if "@" in user_email and "." in user_email:
    print("The applicant has a valid email address.")
else:
    print("The applicant has an invalid email address.")
```

Here we implement some basic validation for an email address of an applicant. If the email contains "@" and "." we consider the email to be valid.

Demo: Using Conditional Statements to Validate Applicant Cell Phone Number

- Here we use a combination of the len() function and the isdigit() string method to validate the cell phone number of an applicant.

```
user_cell_number = input("Please enter your cellphone number: ")
if len(user_cell_number) == 10 and user_cell_number.isdigit():
    print("The applicant has a valid cell phone number.")
else:
    print("The applicant has an invalid cell phone number.")
```

Demo: Using Try-Except Blocks to Validate Applicant Age

- We enter all the code we would like to evaluate within the scope of the try block. We then catch the exception we expect to occur, which in this case is the ValueError and provide the user with an error message.

```
try:
    user_age = input("Please enter your age: ")
    user_age = int(user_age)
    print("The applicant has a valid age.")
except ValueError:
    print("The applicant's age contains invalid characters.")
```

Types of Errors

- **Syntax Error:** Occurs when the interpreter is unable to parse the code because it violates Python language rules, such as improper indentation, incorrect keyword usage, or incorrect operator use.
- **Runtime Error:** Runtime errors are difficult to debug because they occur in real time and are difficult to reproduce.
- **Logical Error:** In Python, a logical error occurs when code runs without any syntax or runtime errors but produces incorrect results due to flawed logic.

Valentina's Shopping Cart

Your challenge is to create a program that will validate the input a user provides for an application according to the application's criteria.

Here are some examples of input that would need to be validated:

Name
Surname
Cell phone number
Location
Skills
Programming languages

Step-by-Step Tasks:

1. **User-Input:** Conceptualise how your program will receive user input, while prioritising user friendliness/experience.
2. **Input-Validation:** Consider the different ways in which users can possibly provide input and clean this data before processing.
3. **Exception Handling:** Handle predictable/common user errors by implementing the try-except blocks for the respective exceptions associated with these errors.
4. **Edge Cases:** Consider edge cases such as the possibility of a mathematical operation where division by zero can occur, and account for these.

Advanced

Challenge:

- Allow users to set up their own application conditions that the applicant's data should adhere to.

Summary

Input Validation

- ★ Input validation is the process of testing input received by the application for compliance against a standard defined within the application.

Exception Handling

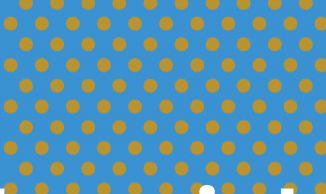
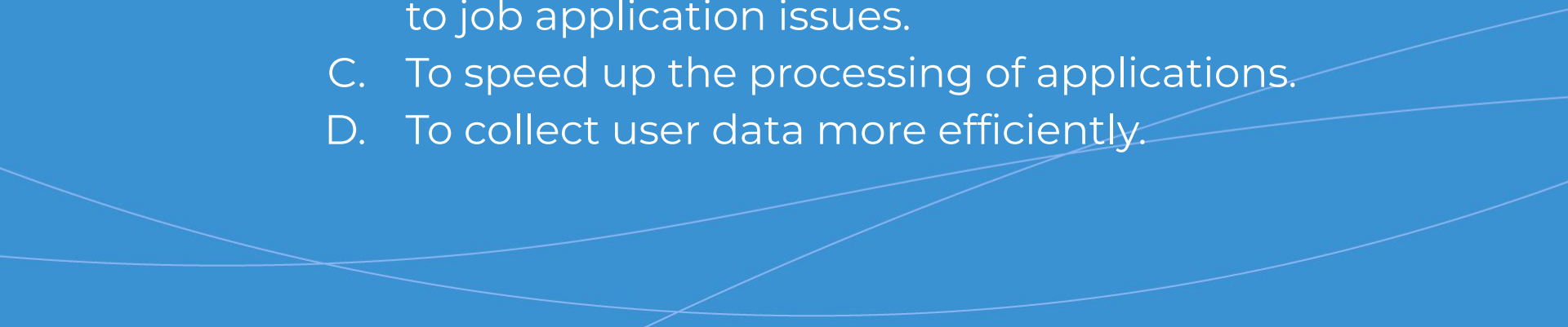
- ★ Exception handling is the process of responding to unwanted or unexpected events when a computer program runs.

Error Types

- ★ The most common types of errors that can be found while programming in Python are categorised as syntax errors, runtime errors, or logical errors


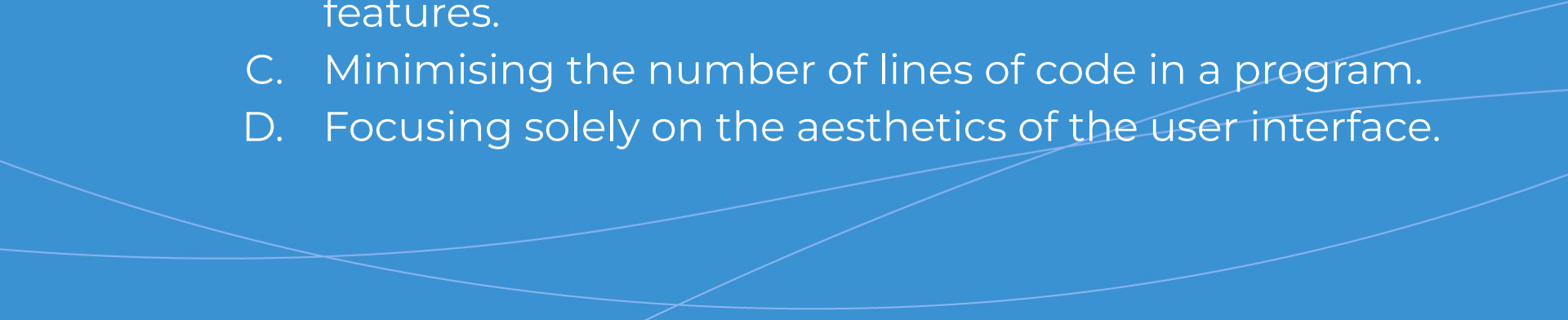


In the CareerCrafter system, why might you define a custom exception like **InvalidApplicationError**?

- A. To automatically correct user input errors.
 - B. To provide a specific error message related to job application issues.
 - C. To speed up the processing of applications.
 - D. To collect user data more efficiently.
- 
- 



What is a key strategy of defensive programming?

- 
- A. Writing code that handles potential errors and unexpected user inputs.
 - B. Using the most advanced programming language features.
 - C. Minimising the number of lines of code in a program.
 - D. Focusing solely on the aesthetics of the user interface.
- 



Questions and Answers

Questions around the Case Study

