Welcome to this

CoGrammar

Task Walkthrough: Task 3 & Task 4

The session will start shortly...

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

CoGrammar

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

Enhancing Accessibility: Activate Browser Captions

Why Enable Browser Captions?

- Captions provide **real-time text for spoken content**, ensuring inclusivity.
- Ideal for individuals in noisy or quiet environments or for those with **hearing impairments**.

How to Activate Captions:

1. YouTube or Video Players:
   - Look for the CC (Closed Captions) icon and click to enable.
2. Browser Settings:
   - Google Chrome: Go to *Settings > Accessibility > Live Captions* and toggle ON.
   - Edge: Enable captions in *Settings > Accessibility*.

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

- Apply your knowledge of string manipulation.
- Implement basic arithmetic operations and user interaction.
- Use your skills by focusing on operators, arithmetic operations, and conditional logic.
- Apply knowledge of while loops, for loops and conditional logic.

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar

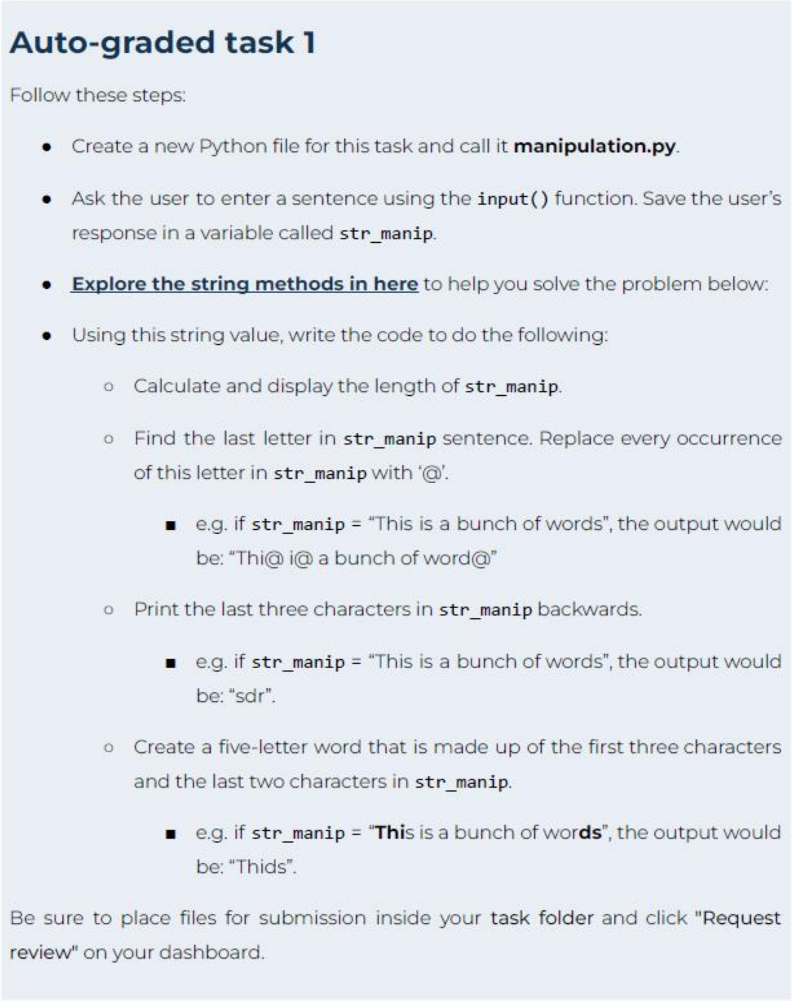
Task Walkthrough: Data Types & Conditional Statements

Part 1 Walkthrough

Variables and String Manipulation

- Variables are containers that hold information.
- A string is simply a way to represent text in programming and is identified with the presence of quotation marks (“ ”)
- Strings can be joined, cut up, and measured.
- Built-in methods to manipulate strings

CoGrammar



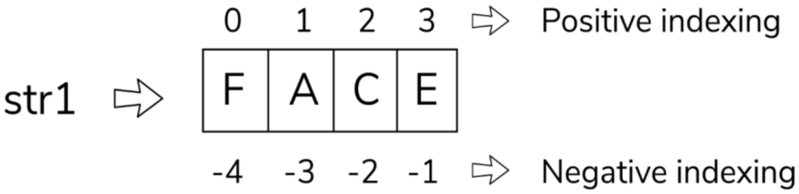
Auto-graded task 1

Follow these steps:

- Create a new Python file for this task and call it **manipulation.py**.
- Ask the user to enter a sentence using the `input()` function. Save the user's response in a variable called `str_manip`.
- **Explore the string methods in here** to help you solve the problem below:
- Using this string value, write the code to do the following:
  - Calculate and display the length of `str_manip`.
  - Find the last letter in `str_manip` sentence. Replace every occurrence of this letter in `str_manip` with '@'.
    - e.g. if `str_manip` = "This is a bunch of words", the output would be: "Thi@ i@ a bunch of word@"
  - Print the last three characters in `str_manip` backwards.
    - e.g. if `str_manip` = "This is a bunch of words", the output would be: "sdr".
  - Create a five-letter word that is made up of the first three characters and the last two characters in `str_manip`.
    - e.g. if `str_manip` = "**Thi**s is a bunch of wor**ds**", the output would be: "Thids".

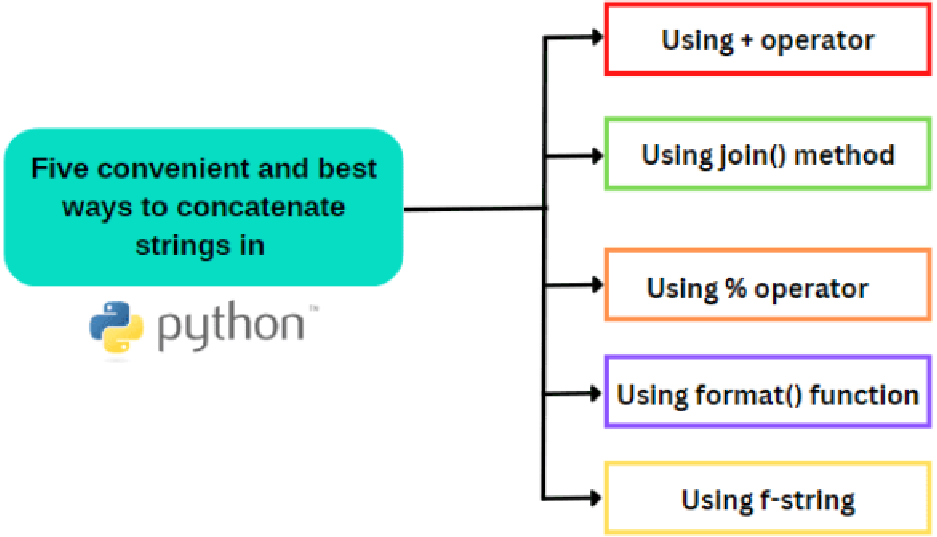
Be sure to place files for submission inside your task folder and click "Request review" on your dashboard.

String Slicing



str1[1:3] = AC

str1[-3:-1] = AC



Five convenient and best ways to concatenate strings in

Using + operator

Using join() method

Using % operator

Using format() function

Using f-string

manipulation.py

Task objective

- The objective of this task is to demonstrate your knowledge of string manipulation by working with user input.
- You'll use different string methods to transform and manipulate the user input in specific ways, such as
  - calculating the string's length,
  - replacing characters,
  - slicing, and reversing portions of the string.
- This task is designed to enhance your problem-solving skills and deepen your understanding of how to work with strings.

Questions and Answers

CoGrammar

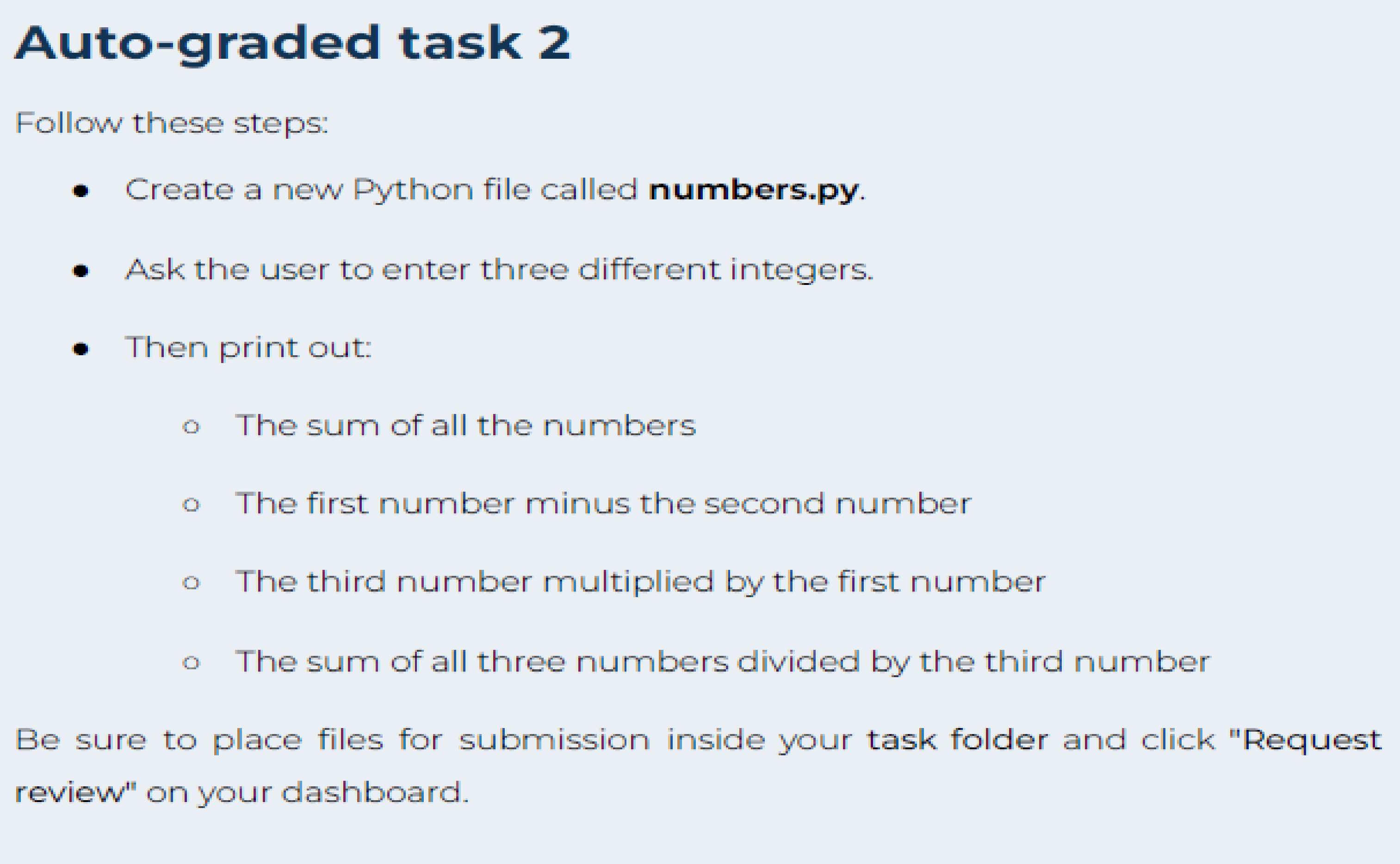
Part 2
Walkthrough

CoGrammar

Data Types Recap

- Data types in programming define the type of data a variable can hold and how that data can be used.
- Data types: Integers, Floats, Strings, and Booleans.
- Data types can be converted from one type to another - Within reason!
- Arithmetic operations in Python: +, -, /, *, %, **
- Arithmetic built-in functions

CoGrammar



Auto-graded task 2

Follow these steps:

- Create a new Python file called **numbers.py**.
- Ask the user to enter three different integers.
- Then print out:
    - The sum of all the numbers
    - The first number minus the second number
    - The third number multiplied by the first number
    - The sum of all three numbers divided by the third number

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

numbers.py

Task objective

- The objective of this task is to build confidence and understanding basic arithmetic operations and user interaction.
- You'll perform calculations:
  - add up all the numbers,
  - subtract the second number from the first,
  - multiply the third number by the first,
  - and finally, divide the sum of all three numbers by the third.
- This exercise is designed to help you practice working with numbers and user inputs, giving you an understanding of how to handle basic operations.

Questions and Answers

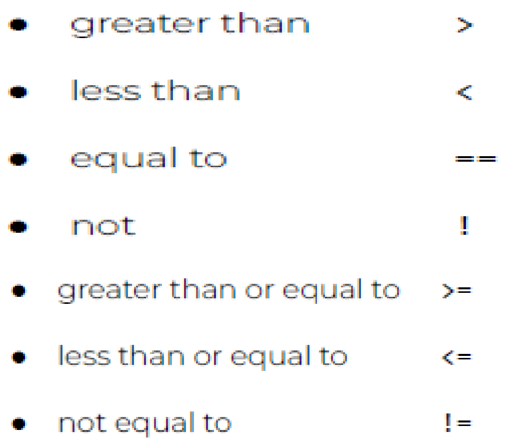
CoGrammar

Part 3
Walkthrough

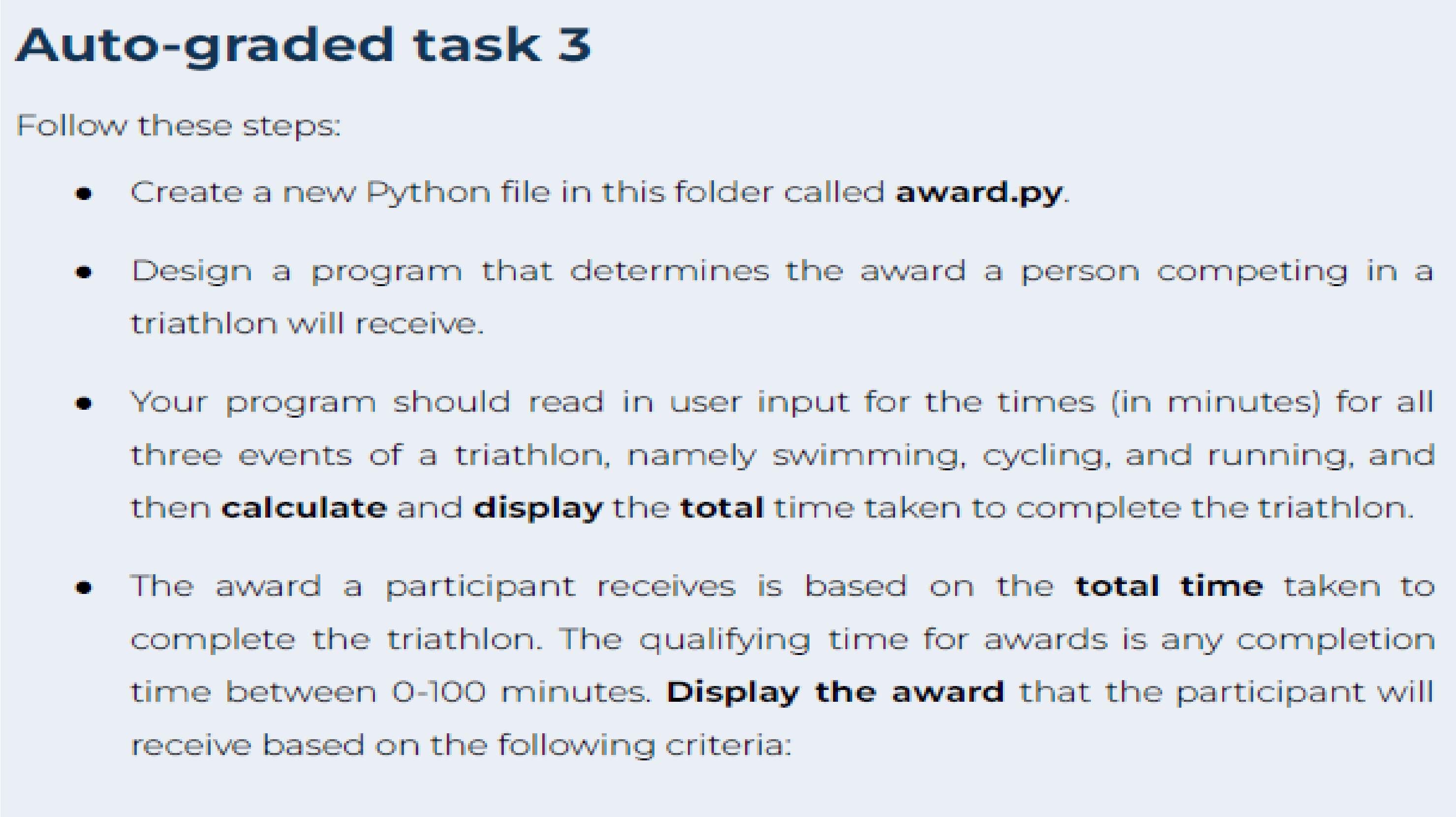
CoGrammar

Conditional Statements Recap

- Conditional statements are like the decision-makers in programming.
- They allow your code to choose different paths based on specific conditions.
- Conditional statements: if, elif, and else
- Comparison operators
- Logical operators: and, or & not



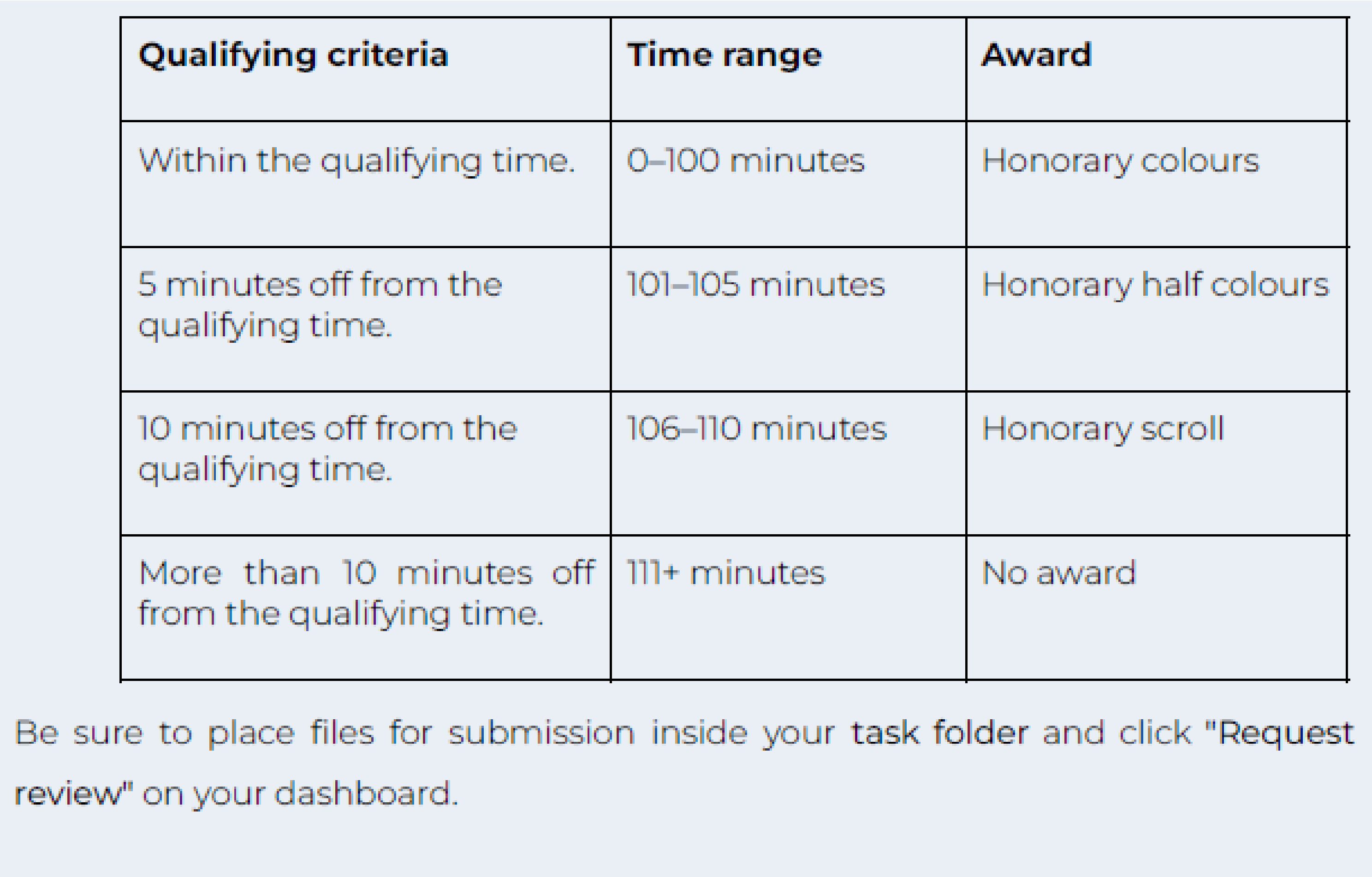
greater than	>
less than	<
equal to	==
not	!
greater than or equal to	>=
less than or equal to	<=
not equal to	!=



Auto-graded task 3

Follow these steps:

- Create a new Python file in this folder called **award.py**.
- Design a program that determines the award a person competing in a triathlon will receive.
- Your program should read in user input for the times (in minutes) for all three events of a triathlon, namely swimming, cycling, and running, and then **calculate** and **display** the **total** time taken to complete the triathlon.
- The award a participant receives is based on the **total time** taken to complete the triathlon. The qualifying time for awards is any completion time between 0-100 minutes. **Display the award** that the participant will receive based on the following criteria:



Qualifying criteria	Time range	Award
Within the qualifying time.	0–100 minutes	Honorary colours
5 minutes off from the qualifying time.	101–105 minutes	Honorary half colours
10 minutes off from the qualifying time.	106–110 minutes	Honorary scroll
More than 10 minutes off from the qualifying time.	111+ minutes	No award

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

award.py

Task objective

- The objective for this task is to enhance your skills by focusing on operators, arithmetic operations, and conditional logic.
- In this task, you will create a program to
  - calculate a triathlon participant's total time from their swimming, cycling, and running events.
  - The program should then determine and display the appropriate award based on the total time.

CoGrammar

Questions and Answers

CoGrammar

SKILLS FOR LIFE

SKILLS BOOTCAMPS

Department for Education

CoGrammar

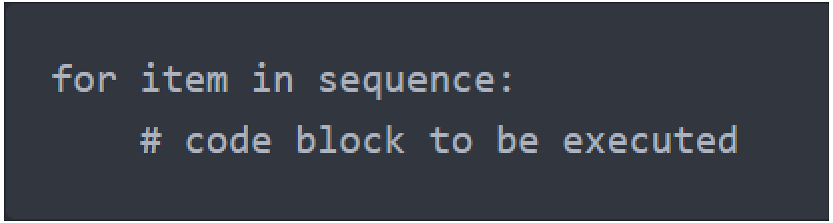
Task Walkthrough: Iteration

Iteration

- Iteration refers to the process of executing a set of instructions repeatedly.
- For loops and while loops are commonly used to handle repetitive tasks in Python.
- Condition-based iteration allows the loop to continue or stop based on a condition (e.g., user input or reaching a specific value)

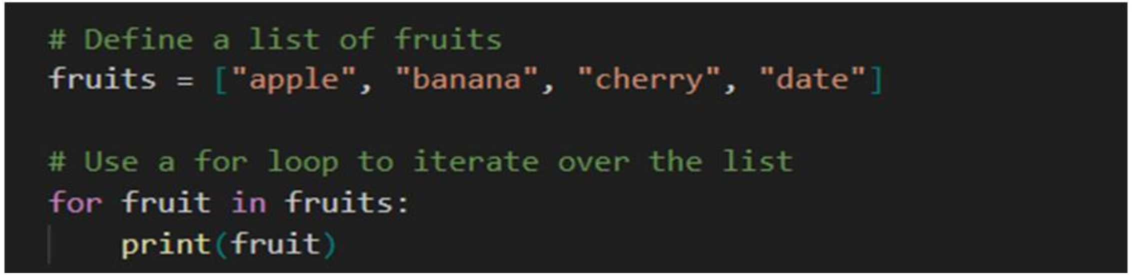
For Loops

- For loops are control flow structures used to iterate over a sequence (such as a list, tuple, string, etc.) and execute a block of code for each element in the sequence.
- For loops are used when you know the number of times you want to execute a block of code.



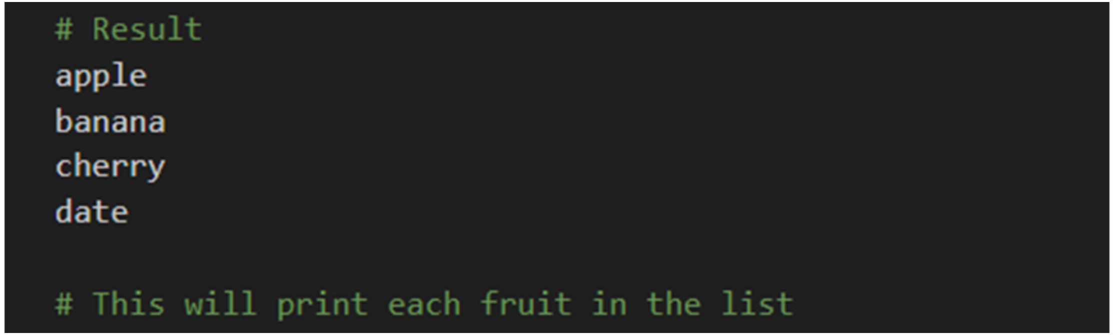
```
for item in sequence:
    # code block to be executed
```

For Loop Example



```python
# Define a list of fruits
fruits = ["apple", "banana", "cherry", "date"]

# Use a for loop to iterate over the list
for fruit in fruits:
    print(fruit)
```

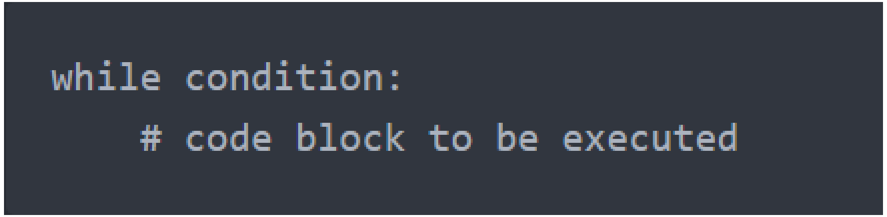


```
# Result
apple
banana
cherry
date

# This will print each fruit in the list
```

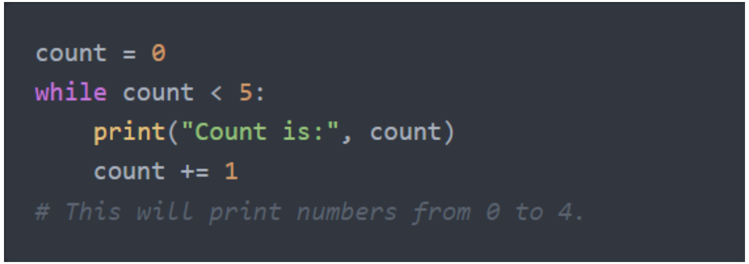
While Loops

- While loops are control flow structures that repeatedly execute a block of code as long as a specified condition is true.
- These are used when you want to execute a block of code repeatedly as long as a specified condition is true. They continue iterating until the condition becomes false.



```
while condition:
    # code block to be executed
```

While Loop Example

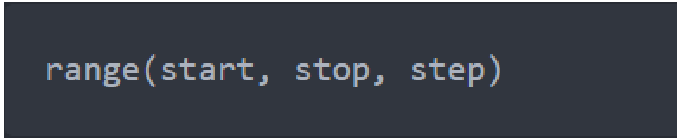


```python
count = 0
while count < 5:
    print("Count is:", count)
    count += 1
# This will print numbers from 0 to 4.
```

- while count < 5: Start of a while loop. It checks if the value of count is less than 5. If this condition is true, the code block inside the loop will execute. If the condition is false, the loop will terminate.
- print("Count is:", count): This line prints the current value of count along with the text "Count is:". Since count is initially 0, it will print "Count is: 0".
- count += 1: This line increments the value of count by 1 in each iteration of the loop. So, after the first iteration, count becomes 1, then 2, etc.

For Loops – Range Function

- Range is a built-in Python function used to generate a sequence of numbers. It is commonly used with for loops.
- Ranges in for loops are a way to specify a sequence of numbers that you want to iterate over. The range() function generates this sequence of numbers based on the arguments you provide.

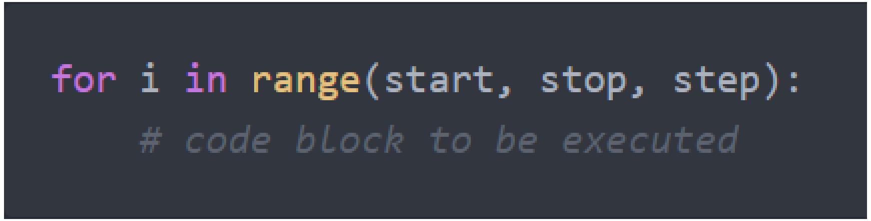


```
range(start, stop, step)
```

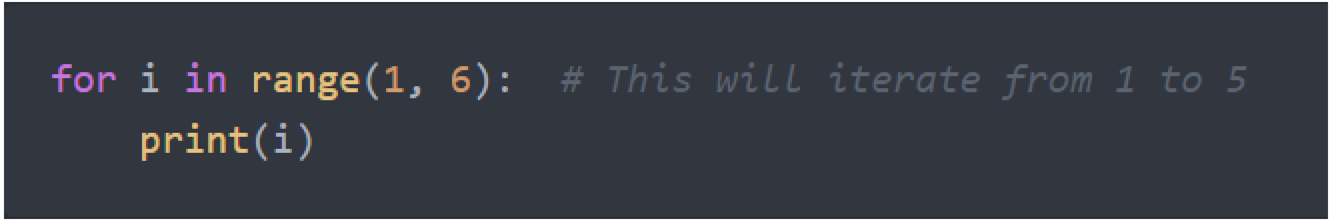
For Loops – Range Function

- Range() takes three arguments: start, stop, and step.
- start: The starting value of the sequence (inclusive). If not provided, it defaults to 0.
- stop: The ending value of the sequence (exclusive). This is a required argument.
- step: The increment between each value in the sequence. If not provided, it defaults to 1.

Range Function Example



```python
for i in range(start, stop, step):
    # code block to be executed
```

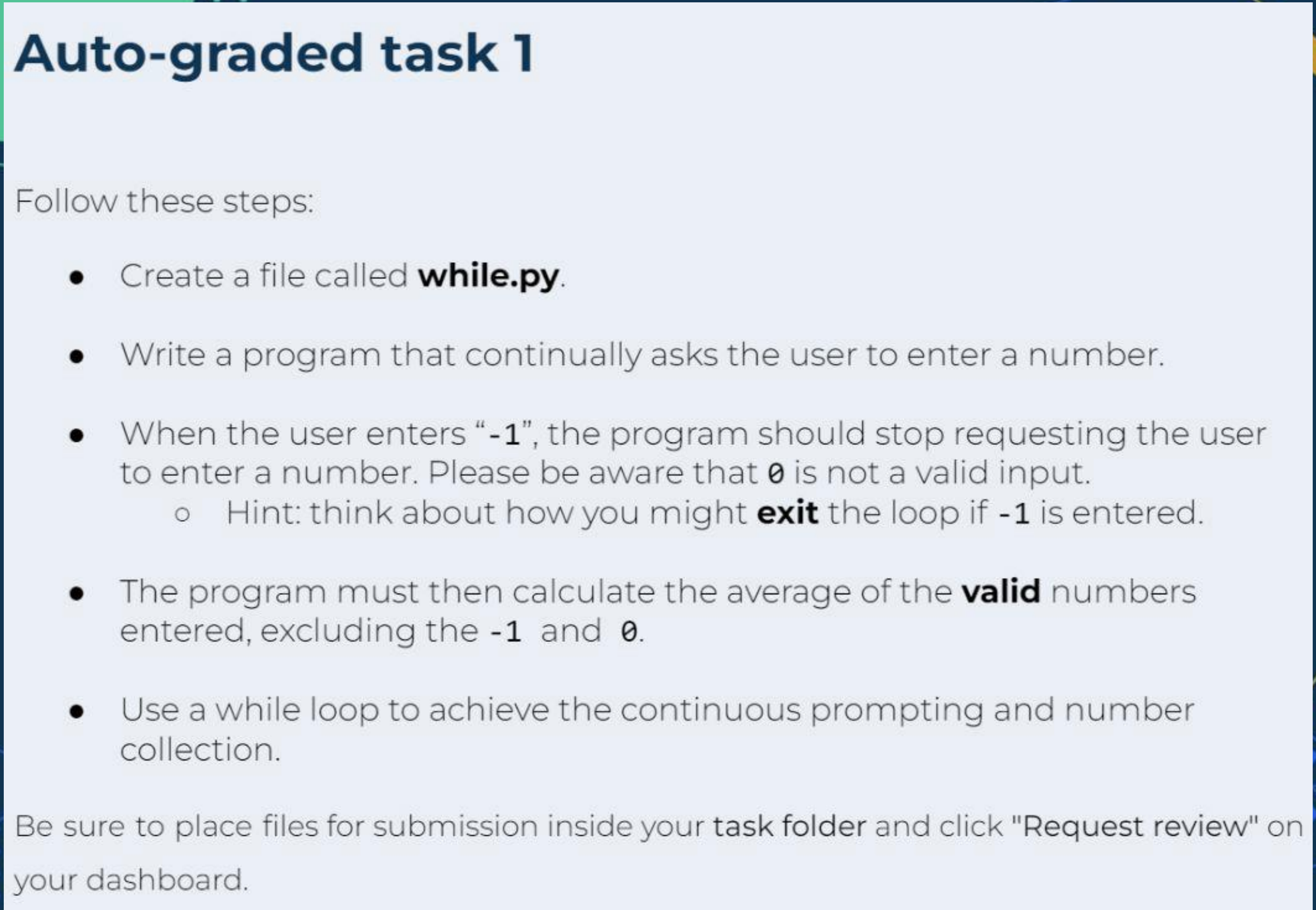


```python
for i in range(1, 6):  # This will iterate from 1 to 5
    print(i)
```

- This loop will print numbers 1 through 5. Remember, the stop value is exclusive, so the loop stops before reaching 6.

Part 1 Walkthrough

CoGrammar



Auto-graded task 1

Follow these steps:

- Create a file called **while.py**.
- Write a program that continually asks the user to enter a number.
- When the user enters "`-1`", the program should stop requesting the user to enter a number. Please be aware that `0` is not a valid input.
    - Hint: think about how you might **exit** the loop if `-1` is entered.
- The program must then calculate the average of the **valid** numbers entered, excluding the `-1` and `0`.
- Use a while loop to achieve the continuous prompting and number collection.

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

while.py

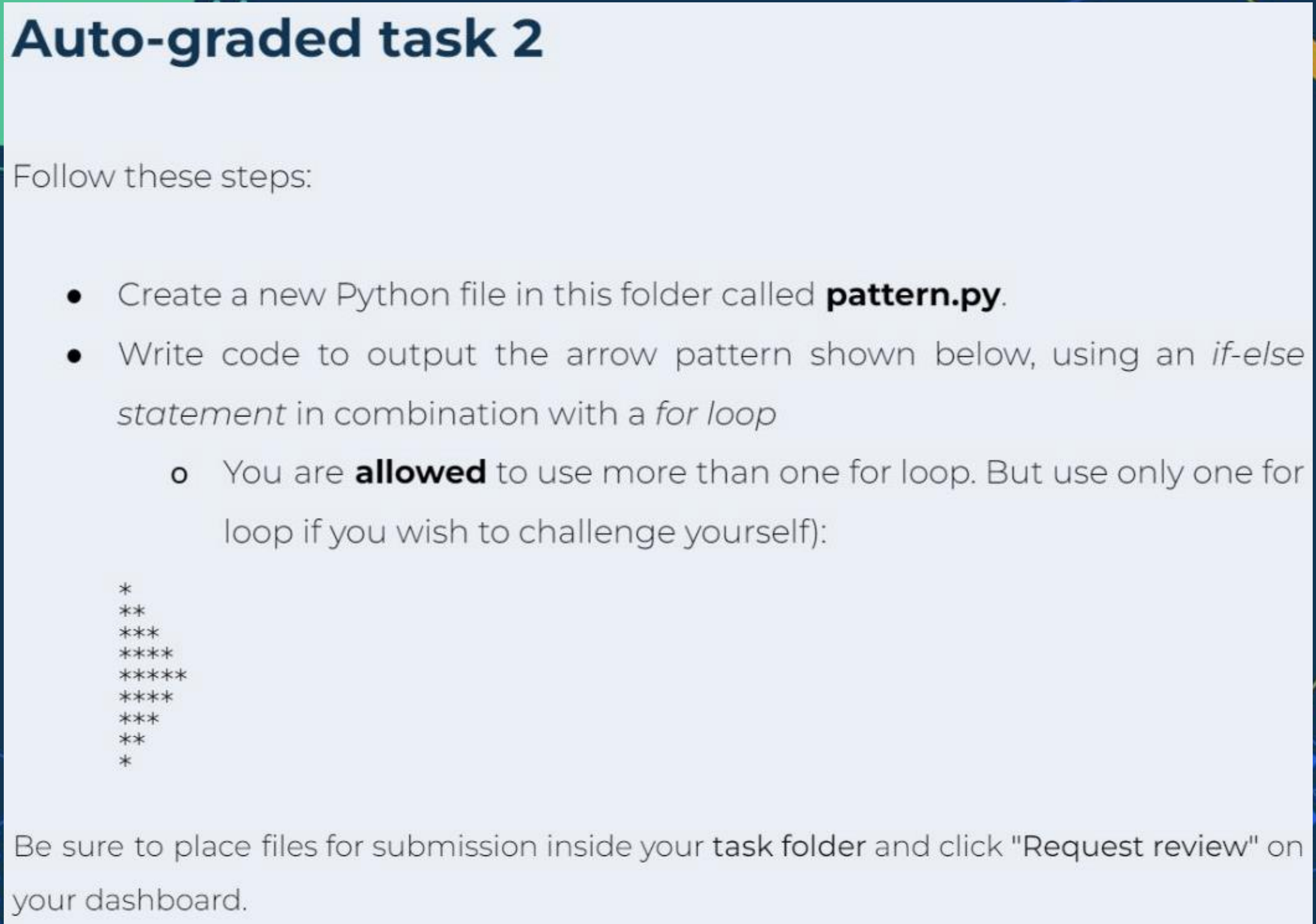
Task Objective

- The objective of this task is to demonstrate your understanding of while loops and their application in repetitive tasks by working with user input.
- You'll use the while loop repetition structure to repeatedly perform actions based on user input, such as:
  - Continuously prompting the user to enter numbers.
  - Stopping the loop when a specific condition is met (input of -1).
  - Calculating the average of the numbers entered, excluding the termination value (-1).
- This task is designed to enhance your problem-solving skills and deepen your understanding of loops and conditional logic.

CoGrammar

Part 2
Walkthrough

CoGrammar



Auto-graded task 2

Follow these steps:

- Create a new Python file in this folder called **pattern.py**.
- Write code to output the arrow pattern shown below, using an *if-else statement* in combination with a *for loop*
  - You are **allowed** to use more than one for loop. But use only one for loop if you wish to challenge yourself):

```
*
**
***
****
*****
****
***
**
*
```

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

pattern.py

Task objective

- The objective of this task is to demonstrate your knowledge of for loops and conditional logic by generating a specific pattern output.
- You'll use an if-else statement in combination with a single for loop to:
    - Dynamically create and output lines of varying patterns (e.g., **, *, *******, etc.).
    - Apply logic within the loop to control the pattern based on conditions.
- This task is designed to enhance your problem-solving skills and deepen your understanding of loop structures and decision-making in Python.

Questions and Answers

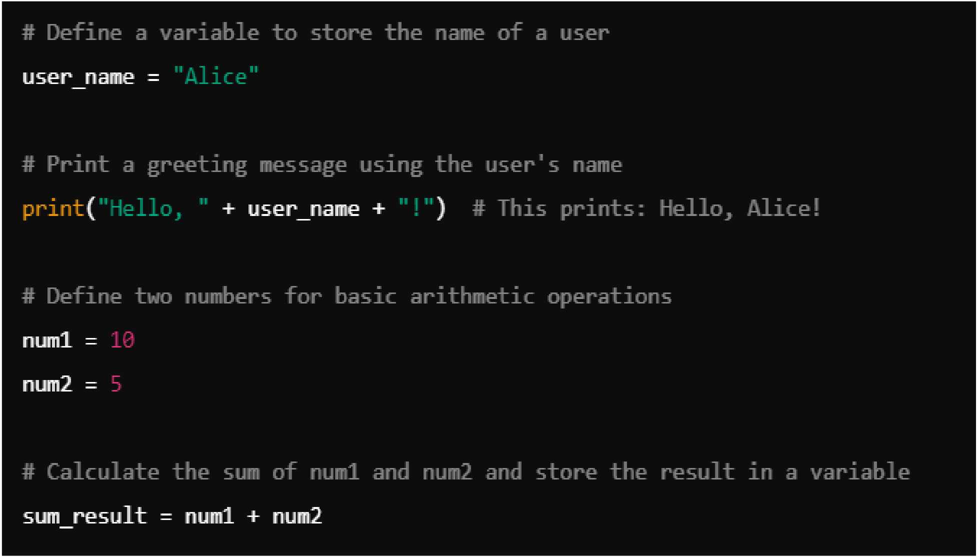
CoGrammar

Documentation and Style

- Add comments to your code. Explain your approach, and/or how your code works.
- Consult the Python PEP8 guidelines: https://peps.python.org/pep-0008/

Pay close attention to:

- Variable names
- Spacing around operators
- Separating logical sections
- Indentation



```
# Define a variable to store the name of a user
user_name = "Alice"

# Print a greeting message using the user's name
print("Hello, " + user_name + "!")  # This prints: Hello, Alice!

# Define two numbers for basic arithmetic operations
num1 = 10
num2 = 5

# Calculate the sum of num1 and num2 and store the result in a variable
sum_result = num1 + num2
```

CoGrammar

Learner Challenge

- **For those who are looking for an additional challenge:**
  - Adjust the code for the pattern.py task example, by asking the user for a max-width diamond size and print a full diamond pattern.
  - Note: Each line of the pattern will increase by 2 characters until the middle is printed and the total characters will need to be centred and with the appropriate spaces in the front.

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar