**ASSIGNMENT-7.3 ( 2403A52398 )**

## Task-1: Syntax Error

## Wrong code:

A computer screen shot of a program code

AI-generated content may be incorrect.

## Explaination:

The error message SyntaxError: expected ':' indicates that there's a missing colon at the end of the def line in your Python code. Function definitions in Python require a colon to mark the beginning of the function body. I've fixed the code by adding the colon.

## Correct code:

A black background with white text

AI-generated content may be incorrect.

## Task-2: Logic error in a loop that causes infinite iteration

## Wrong code:

A screenshot of a computer program

AI-generated content may be incorrect.

## Explaination:

* **def count\_down(n):**: This line defines a function named count\_down that takes one argument, n.
* **while n>=0:**: This is a while loop. It means the code inside the loop will keep running as long as the condition n >= 0 is true (as long as the value of n is greater than or equal to 0).
* **print(n)**: Inside the loop, this line prints the current value of n to the console.
* **n -= 1 # Decrement n to count down**: This line subtracts 1 from the value of n. This is the crucial step that makes the number count down.
* **count\_down(5)**: This line calls the count\_down function and passes the value 5 as the argument for n.

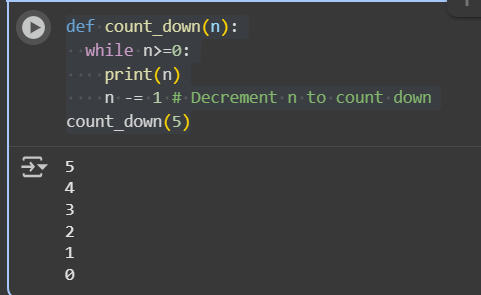
So, when you run this code, it starts with n = 5, prints 5, then changes n to 4. It checks if 4 is greater than or equal to 0 (which it is), prints 4, changes n to 3, and so on, until n becomes -1. At that point, the while n>=0 condition is no longer true, and the loop stops.

The output you see is the numbers 5, 4, 3, 2, 1, and 0 printed on separate lines.

0 / 2000

Gemini can make mistakes so double-check it and use code with caution. [Learn more](http://g.co/legal/generative-code" \t "_blank)

## Correct code:



## Task-3: Division by zero Error Handling

## Wrong code:

## Explaination:

You've defined a divide function that returns the result of a divided by b. When you call print(divide(10, 0)), you are trying to divide by zero, which is mathematically undefined and causes a ZeroDivisionError in Python.

To handle this, you can add a check to see if b is zero before performing the division. I'll modify the cell to include this error handling.

Great! The divide function now handles division by zero gracefully.

What would you like to do next? You could:

* Test the divide function with different numbers.
* Explore other types of error handling in Python.
* Move on to a different coding task.

## Correct code: A screenshot of a computer AI-generated content may be incorrect.

## Task-4: Class definition (missing self in parameters)

## Wrong code:

A computer screen shot of a computer code

AI-generated content may be incorrect.

## Explaination:

The error message TypeError: Rectangle.\_\_init\_\_() takes 2 positional arguments but 3 were given tells us that when you tried to create a Rectangle object with Rectangle(5, 3), the \_\_init\_\_ method was expecting two arguments but received three.

In Python class methods, the first argument is conventionally named self, and it refers to the instance of the class itself. When you create an object like Rectangle(5, 3), the values 5 and 3 are passed as arguments, but Python also implicitly passes the newly created object as the first argument to \_\_init\_\_.

Your original \_\_init\_\_ method was defined as def \_\_init\_\_(length, width):, which only expects length and width. However, because of the implicit self argument, it was receiving three arguments in total (self, 5, and 3), leading to the error.

To fix this, you need to include self as the first parameter in the \_\_init\_\_ method definition. I will modify the cell to correct this.

## Correct code:

A screen shot of a computer code

AI-generated content may be incorrect.

## Task-5: Index Error

## Wrong code:

A screenshot of a computer code

AI-generated content may be incorrect.

## Explaination:

You've provided a code snippet that will result in an IndexError. This happens because you are trying to access an element at index 5 in the numbers list, but the list only has elements at indices 0, 1, and 2.

Python lists are zero-indexed, meaning the first element is at index 0, the second at index 1, and so on. Since your list numbers has three elements, the valid indices are 0, 1, and 2.

To fix this, you need to use an index that is within the valid range of the list. I will modify the cell to access a valid index (e.g., index 0).

Great! The code now successfully accesses a valid index in the list.

What would you like to do next? We could:

* Explore other list operations (like adding, removing, or slicing elements).
* Learn about different ways to handle errors in Python.
* Work on a different coding problem.

## Correct code:

A screen shot of a computer

AI-generated content may be incorrect.