

Assignment-7

2403A52132

Task 1:

```
python

def add(a, b)
    return a + b
```

Code and Output:

The screenshot shows a Jupyter Notebook interface with a file named 'Untitled4.ipynb'. The code cell contains the following Python code:

```
[ ] Start coding or generate with AI.

def add(a,b):
    return a+b
```

The Gemini chat window on the right shows the following interaction:

Gemini ✕

K def add(a,b) return a+b

✦ Okay, I'll add this function definition to a new cell.

I've added the `add` function to your notebook. What would you like to do next? You could try calling the function with some numbers, like `add(2, 3)`, or maybe define another function?

👍 🗨

✦ What can I help you build? 🔄 ▶

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

🔍 Variables 📄 Terminal 9:31 AM Python 3

Explanation:

Gemini ✕

K def add(a,b) return a+b what is the syntax error in the code

✦ The syntax error in your code `def add(a,b) return a+b` is that you are missing a colon `:` at the end of the function definition line and you need an indentation before the `return` statement.

The correct syntax for defining a function in Python is:

```
def function_name(parameters):
    # indented code block
    return result
```

So, your `add` function should look like this:

```
def add(a, b):
    return a + b
```

This code defines a function called `add` that takes two arguments, `a` and `b`, and returns their sum.

👍 🗨

K explanation

✦ The code in cell `a13dfb80` defines a Python function called `add`. This function takes two arguments, `a` and `b`, and returns their sum.

👍 🗨

✦ What can I help you build? 🔄 ▶

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

Task 2:

python

```
def count_down(n):  
    while n >= 0:  
        print(n)  
        n += 1 # Should be n -= 1
```

Identify and fix a logic error in a loop that causes infinite iteration.

Code and Output:

The screenshot shows a Jupyter Notebook interface with a file named 'Untitled4.ipynb'. The notebook has a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu bar is a search bar and a 'Run all' button. The notebook contains a single code cell with the following Python code:

```
def count_down(n):  
    while n>=0:  
        print(n)  
        n-=1 #should be n-=1  
count_down(5)
```

The output of the code cell is displayed below the code, showing the numbers 5, 4, 3, 2, 1, and 0, each on a new line. To the right of the code cell is a Gemini chat interface. The chat history shows a conversation where the user asks to identify and fix a logic error in a loop that causes infinite iteration. The Gemini response explains the error and provides the corrected code. The user then asks for an output, and the Gemini response provides the output of the corrected code.

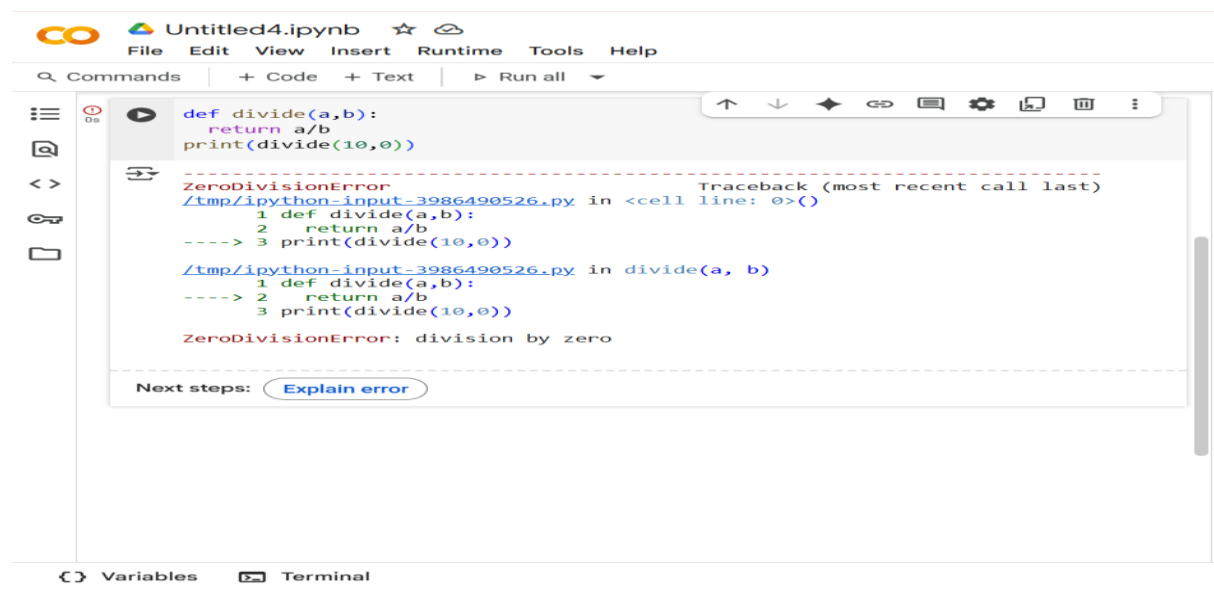
Explanation:

The screenshot shows a Gemini chat interface with a single chat message. The message is from the Gemini model and explains the code in the visible cells. The explanation states that the code contains a Python function called 'count_down' which takes one argument, 'n', and prints the numbers from 'n' down to 0, inclusive. It uses a 'while' loop that continues as long as 'n' is greater than or equal to 0. Inside the loop, it prints the current value of 'n' and then decrements 'n' by 1 ('n -= 1'). The line '#should be n-=1' is a comment indicating the intended logic after a previous correction.

Task 3:

```
# Debug the following code  
def divide(a, b):  
    return a / b  
  
print(divide(10, 0))
```

Code and Output:



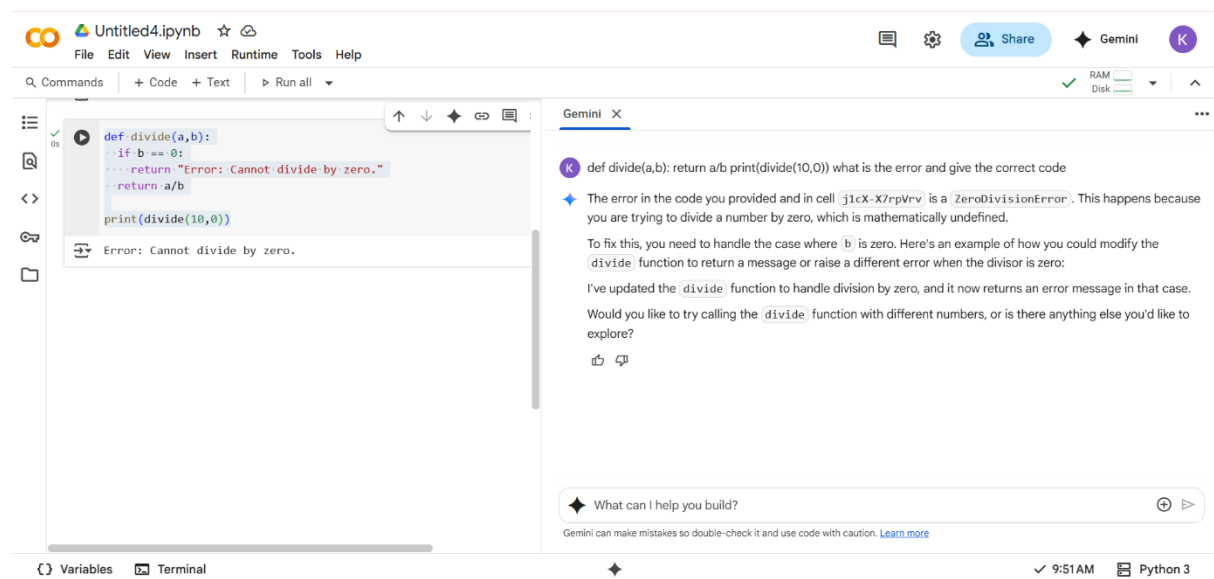
The screenshot shows a Jupyter Notebook titled "Untitled4.ipynb". The code cell contains the following Python code:

```
def divide(a,b):  
    return a/b  
print(divide(10,0))
```

The output of the code cell shows a `ZeroDivisionError` traceback:

```
ZeroDivisionError                                Traceback (most recent call last)  
/tmp/ipython-input-3986490526.py in <cell line: 0>()  
      1 def divide(a,b):  
      2     return a/b  
----> 3 print(divide(10,0))  
  
/tmp/ipython-input-3986490526.py in divide(a, b)  
      1 def divide(a,b):  
----> 2     return a/b  
      3 print(divide(10,0))  
  
ZeroDivisionError: division by zero
```

Below the traceback, there is a "Next steps:" section with a button labeled "Explain error".



The screenshot shows the same Jupyter Notebook interface, but with a Gemini chat window open on the right side. The code cell now contains the following Python code:

```
def divide(a,b):  
    if b == 0:  
        return "Error: Cannot divide by zero."  
    return a/b  
print(divide(10,0))
```

The output of the code cell shows the error message:

```
Error: Cannot divide by zero.
```

The Gemini chat window displays the following conversation:

K def divide(a,b): return a/b print(divide(10,0)) what is the error and give the correct code

◆ The error in the code you provided and in cell j1cX-X7rpVrv is a `ZeroDivisionError`. This happens because you are trying to divide a number by zero, which is mathematically undefined.

To fix this, you need to handle the case where `b` is zero. Here's an example of how you could modify the `divide` function to return a message or raise a different error when the divisor is zero:

I've updated the `divide` function to handle division by zero, and it now returns an error message in that case.

Would you like to try calling the `divide` function with different numbers, or is there anything else you'd like to explore?

◆ What can I help you build?

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

Explanation:

- Cell `j1cX-X7rpVrv`: This cell contains a Python function called `divide`. This function takes two arguments, `a` and `b`. It includes an `if` statement to check if `b` is equal to 0. If it is, the function returns the string "Error: Cannot divide by zero." to prevent a `ZeroDivisionError`. If `b` is not zero, the function returns the result of dividing `a` by `b` (`a / b`). The `print(divide(10,0))` line calls this function with `a=10` and `b=0` and prints the returned value, which will be the error message.



Task 4:

```
python

class Rectangle:
    def __init__(length, width):
        self.length = length
        self.width = width
```

Code and output:

The screenshot shows a Jupyter Notebook interface with a file named "Untitled4.ipynb". The notebook contains a Python class definition for a `Rectangle` class and its usage in a code cell.

Code Cell:

```
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

[23] # Create a Rectangle object with length 10 and width 5
my_rectangle = Rectangle(10, 5)

# Access and print the attributes
print(f"Length of the rectangle: {my_rectangle.length}")
print(f"Width of the rectangle: {my_rectangle.width}")
```

Output:

```
Length of the rectangle: 10
Width of the rectangle: 5
```

Gemini Chat Window:

lass Rectangle: def __init__(length, width): self.length = length self.width = width what is the error and give the correct code with an example

It seems you're still having trouble with the `Rectangle` class definition. The error here is the same as before: the `__init__` method is missing the required `self` parameter, and there's a typo in the method name (it should be `__init__` with double underscores). The `self` parameter is a reference to the instance of the class being created.

Here's the corrected code for the `Rectangle` class:

```
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width
```

Great, the `Rectangle` class definition is now corrected.

Here is an example of how to create an instance of the `Rectangle` class and access its attributes:

```
my_rectangle = Rectangle(10, 5)
print(my_rectangle.length)
print(my_rectangle.width)
```

I've shown you how to define and use the `Rectangle` class.

What would you like to do next? You could try creating other `Rectangle` objects with different dimensions, add more methods to the `Rectangle` class (like calculating the area or perimeter), or work on something else?

What can I help you build?

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

Explanation:

- Cell `q1ddCP8ZqyrS`: This cell defines a Python class called `Rectangle`. Classes are blueprints for creating objects. The `__init__` method is a special method called a constructor; it's automatically called when you create a new `Rectangle` object. The `self` parameter refers to the instance of the class being created. This `__init__` method takes `length` and `width` as arguments and stores them as attributes of the object using `self.length = length` and `self.width = width`.
- Cell `a498d427`: This cell demonstrates how to use the `Rectangle` class defined in the previous cell.
 - `my_rectangle = Rectangle(10, 5)` creates a new `Rectangle` object with a length of 10 and a width of 5. This calls the `__init__` method of the `Rectangle` class.
 - The `print` statements access the `length` and `width` attributes of the `my_rectangle` object using dot notation (`my_rectangle.length` and `my_rectangle.width`) and print their values.

Task 5:

python

```
numbers = [1, 2, 3]
print(numbers[5])
```

Code and Output:

The screenshot shows a Jupyter Notebook interface. The top part displays a code cell with the following Python code:

```
numbers=[1,2,3]
print(numbers[5])
```

Below the code cell, the output shows a `Traceback (most recent call last)` and an `IndexError: list index out of range` message. The error message indicates that the list `numbers` has only three elements (indices 0, 1, and 2), and attempting to access index 5 is out of range.

Below the error message, there is a button labeled "Next steps: Explain error".

The bottom part of the screenshot shows the Gemini chat interface. The chat window displays the following message:

numbers=[1,2,3] print(numbers[5]) what is the error and the correct code with an example

The Gemini response explains the error and provides the corrected code:

The error in the code you provided and in cell `jHrUvEhgu_gr` is an `IndexError: list index out of range`. This error occurs when you try to access an element in a list using an index that does not exist. In your code, the list `numbers` has only three elements, with indices 0, 1, and 2. You are trying to access the element at index 5, which is beyond the end of the list.

To fix this, you need to use a valid index, which should be between 0 and 2 (inclusive) for this list.

Here's the corrected code accessing a valid index, along with an example:

I've corrected the code to access a valid index in the `numbers` list, and you can see the output of accessing the first element (at index 0) in the cell output.

What would you like to do next? You could try accessing other valid indices in the list, learn

What can I help you build?

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

Explanation:

- `numbers = [1, 2, 3]` creates a list named `numbers` containing the integers 1, 2, and 3.
- `print(numbers[0])` accesses the element at index 0 in the `numbers` list and prints its value. In Python, list indices start from 0, so index 0 corresponds to the first element. The comment `# Accessing the first element (index 0)` clarifies this.

The screenshot shows the Gemini chat interface with a follow-up question and response. The question is:

What can I help you build?

The response is:

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)