

2303A51802
Billa Spurthi
Batch 29

Task1:

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Lab7.5 > Task1.py > ...

1 ## Task1 - 7.5

2 # Analyze the following Python function and identify any logical issue related to default arguments.

3 def add_item(item, items=None):

4 if items is None:

5 items = []

6 items.append(item)

7 return items

8

9 print(add_item(1))

10 print(add_item(2))

11

12 # Explanation

13 # The function 'add_item' uses a default argument 'items' which is set to 'None'.

14 # This is a common practice to avoid mutable default arguments like lists.

15 # If we were to use 'items=[]' directly as a default argument,

16 # it would be shared across all calls to the function, leading to unexpected

17 # behavior where items from previous calls would accumulate in the list.

18 # By checking if 'items' is 'None' and then initializing it to an empty list,

19 # we ensure that each call to the function gets its own separate list,

20 # thus avoiding the logical issue of shared mutable default arguments.

21

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding'; & 'on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '59rs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task1.py'

[1]

[2]

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> |

Task2:

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Lab7.5 > Task2.py > ...

1 #Task2 7.5

2 # Analyze the following Python function. It compares floating-point values and produces an unexpected result.

3 def check_sum():

4 return abs((0.1 + 0.2) - 0.3) < 1e-9

5

6 print(check_sum())

7

8 #Explanation: The issue arises due to the way floating-point arithmetic works in Python

9 # (and in general in computing). The expression 0.1 + 0.2 does not exactly equal 0.3 due

10 # to precision limitations of floating-point representation.

11 # To fix this, we can use a small tolerance value (like 1e-9)

12 # to check if the two values are close enough, rather than checking for exact equality.

13

14

15

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding'; & 'on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '59rs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task2.py'

True

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> |

Task3:

The screenshot shows the VS Code interface with three main panels:

- EXPLORER:** A file tree on the left with 'Lab7.5' selected under the 'AI ASSISTANT CODING' folder.
- AI ASSISTANT CODING:** The central panel showing a Python script for a recursive countdown function. The code is as follows:


```

1 # Task3 7.5
2 # Analyze the following Python function. It uses recursion but causes an infinite recursion error.
3 def countdown(n):
4     print(n)
5     if n == 0:
6         return
7     return countdown(n-1)
8
9 countdown(5)
10 #Explanation:
11 # The function 'countdown' is designed to print numbers from 'n' down to 0.
12 # However, if the input 'n' is negative, the function will keep calling itself
13 # with decreasing values of 'n', leading to an infinite recursion error.
14 # To fix this, we can add a base case to handle negative values of 'n'
15 # and prevent the function from calling itself indefinitely.
16
17
18
```
- TERMINAL:** The bottom panel showing a command prompt. The command entered is:


```
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5' & python Task3.py
```

 The output shows a recursive error:


```

5
4
3
2
1
0
RecursionError: maximum recursion depth exceeded

```

Task4:

The screenshot shows the Visual Studio Code editor with a file named `Task4.py` open. The code is as follows:

```

1 #Task4 7.5
2 # Analyze the following Python function. It raises an error due to accessing a non-existing dictionary key.
3 def get_value():
4     data = {"a": 1, "b": 2}
5     return data.get("c", "key not found")
6
7 print(get_value())
8 #Explanation:
9 # The function 'get_value' attempts to access the key "c" in the dictionary 'data', which does not exist.
10 # This would normally raise a KeyError if we tried to access it directly (e.g., 'data["c"]').
11 # However, by using the 'get' method of the dictionary,
12 # we can provide a default value ("key not found") that will be
13 # returned if the key is not found in the dictionary.
14
15
16
17

```

The output in the console is:

```

key not found

```

Task5

The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'AI ASSISTANT CODING' with a folder 'Lab7.5' containing files 'Task1.py' through 'Task15.py'. The file 'Task5.py' is selected. The code in 'Task5.py' is as follows:

```
1
2 # Analyze the following Python function. The loop inside the function runs infinitely.
3 # Bug: Infinite loop
4 def loop_example():
5     i = 0
6     while i < 5:
7         print(i)
8         i += 1
9
10 loop_example()
11
12
13 #Explanation:
14 # The function 'loop_example' is designed to print numbers from 0 to 4.
15 # However, if the variable 'i' is not incremented properly,
16 # it will lead to an infinite loop. In this case, the line 'i += 1'
17 # is correctly incrementing 'i', so the loop will terminate after printing 0 to 4.
18 # If we were to remove or comment out the line 'i += 1', the loop would run indefinitely, printing 0 repeatedly.
19
```

The terminal shows the command to run the script:

```
PS C:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding> c:\; cd 'c:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding\' & on.exe 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' "rs\SPURTHI\OneDrive\Desktop\AI Assistant coding\Lab7.5\Task5.py"
```

The output of the terminal is:

```
0
1
2
3
4
```

Task6:

The screenshot displays the VS Code interface with three main panes. The Explorer pane on the left shows a file tree for 'AI ASSISTANT CODING' with files Task1.py through Task15.py. The AI Assistant pane in the middle shows a list of tasks, with 'Task6.py' selected. The Editor pane on the right shows the code for Task6.py, which includes a Python script with a tuple unpacking error and a comment explaining the issue. The terminal at the bottom shows the command prompt and the execution of the script.

Task7:

EXPLORER

...

Task10.pyTask9.pyTask8.pyTask7.pyTask6.py Lab7.5Task5.py Lab7.5Task4

AI ASSISTANT CODING

> Lab 6

> Lab3.1

> Lab5

Lab7.5

Task1.pyTask2.pyTask3.pyTask4.pyTask5.pyTask6.pyTask7.pyTask8.pyTask9.pyTask10.pyTask11.pyTask12.pyTask13.pyTask14.pyTask15.pylog.txtOddEven.pyPrime.pyString.pyuser_activity.log

Lab7.5 > Task7.py > ...

1# Analyze the following Python code. It raises an indentation error due to inconsistent spacing.
2# Identify the issue and fix the indentation to follow Python standards
3
4def func():
5 x = 5
6 y = 10
7 return x + y
8
9
10print(func())
11
12#explanation: The issue arises because the code uses inconsistent indentation
13# (mixing tabs and spaces or using different numbers of spaces).
14# To fix this, ensure that all lines of code within the function are
15# indented with the same number of spaces (commonly 4 spaces) and avoid mixing tabs and spaces.

PROBLEMSOUTPUTDEBUG CONSOLETERMINALPORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c::; cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task7.py'
15
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding>

Task8:

EXPLORER

...

Task10.pyTask9.pyTask8.pyTask7.pyTask6.py Lab7.5Task5.py Lab7.5

AI ASSISTANT CODING

> Lab 6

> Lab3.1

> Lab5

Lab7.5

Task1.pyTask2.pyTask3.pyTask4.pyTask5.pyTask6.pyTask7.pyTask8.pyTask9.pyTask10.pyTask11.pyTask12.pyTask13.pyTask14.pyTask15.pylog.txtOddEven.pyPrime.pyString.pyuser_activity.log

Lab7.5 > Task8.py > ...

1# The following code raises an import error. Identify the issue and correct the module name.
2import math
3print(math.sqrt(16))
4#explanation: The import statement is correct,
5# and the code should work without any issues.
6# If you are encountering an import error,
7# it may be due to a problem with your Python environment or
8# the math module itself. Ensure that you have Python installed
9# correctly and that there are no issues with your environment.
10
11

PROBLEMSOUTPUTDEBUG CONSOLETERMINALPORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c::; cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task8.py'
4.0
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding>

Task10:

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Task10.py X Task9.py Task8.py Task7.py Task6.py Lab7.5 Task5.py Lab7.5 Task4.py Lab7.5

Lab7.5 > Task10.py > ...

1 # Identify the missing variable definitions and fix the function using parameters. Add assert test cases.

2 def calculate_area(length, width):

3 | return length * width

4

5 # Assertions

6 assert calculate_area(5, 4) == 20

7 assert calculate_area(3, 3) == 9

8 assert calculate_area(10, 2) == 20

9

10 print("All tests passed.")

11

12 #explanation: The original function likely had missing variable definitions

13 # for 'length' and 'width', which would cause an error when trying to calculate

14 # the area. By adding these as parameters to the function, we can pass the necessary

15 # values when calling the function. The assert statements are used to test that the

16 # function returns the expected results for given inputs.

17

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding'; & on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '54152' '...' rs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task10.py'

All tests passed.

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> |

Task11

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Task11.py X Task10.py Task9.py Task8.py Task7.py Task6.py Lab7.5 Task5.py Lab7.5 Task4.py Lab7.5

Lab7.5 > Task11.py > ...

1 # Explain why adding int and string fails and fix using type conversion.

2

3 # Adding an integer and a string fails because Python does not allow implicit type conversion between these types.

4 # To fix this, we must explicitly convert one of the values to the same type as the other.

5

6 # Example of failure:

7 # result = 5 + "10" # This raises a TypeError

8

9 # Corrected version:

10 result = 5 + int("10") # Convert string "10" to integer 10

11 print(result) # Output: 15

12 #explanation: In the corrected version, we use the int()

13 # function to convert the string "10" into an integer before

14 # performing the addition. This allows us to successfully add

15 # the two values together without any errors.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding'; & 'c:\Python on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '54152' '...' rs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task11.py'

15

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> |

Task12:

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Task12.py X Task11.py Task10.py Task9.py Task8.py Task7.py Task6.py Lab7.5

Lab7.5 > Task12.py > ...

1 # Fix the type mismatch between string and list.

2 def combine():

3 | return "Numbers: " + str([1, 2, 3])

4

5 assert combine() == "Numbers: [1, 2, 3]"

6 assert "Numbers: " + str([4, 5]) == "Numbers: [4, 5]"

7 assert isinstance(combine(), str)

8

9 print("All tests passed.")

10

11 #explanation: The issue arises because we are trying to

12 # concatenate a string with a list, which is not allowed in

13 # Python. To fix this, we can convert the list to a string using

14 # the str() function before concatenating it with the other string.

15 # This way, we can successfully combine the two without any type mismatch errors.

16

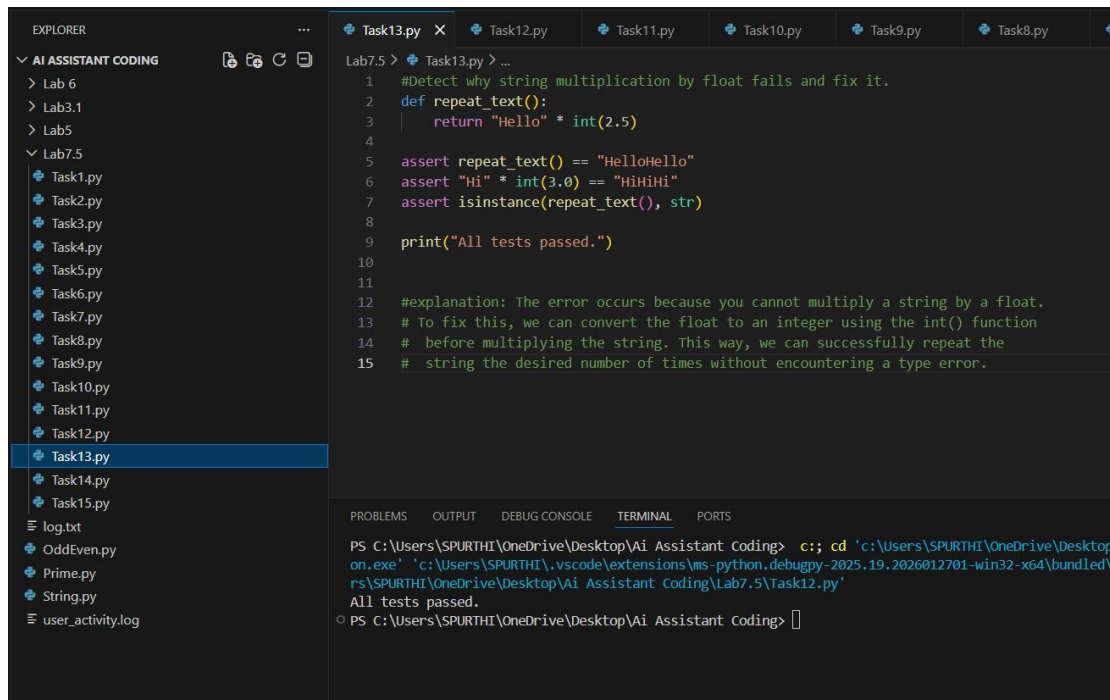
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding'; & on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '54152' '...' rs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task12.py'

All tests passed.

PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> |

Task13

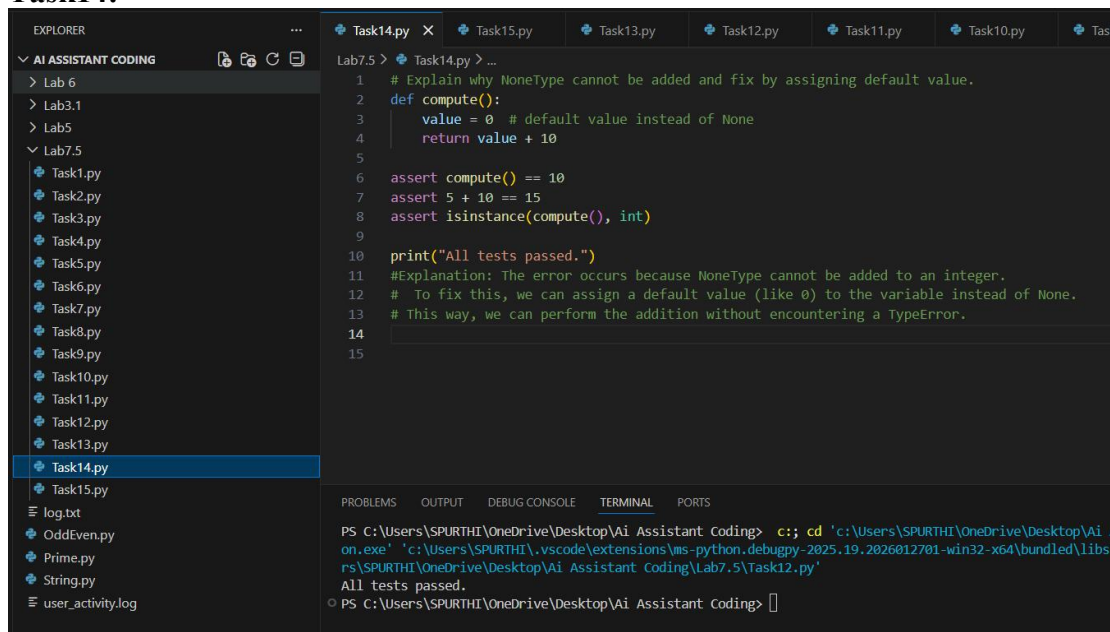


The screenshot shows the VS Code interface with the Explorer panel on the left displaying a file tree under 'AI ASSISTANT CODING'. The file 'Task13.py' is selected. The main editor shows the code for Task13.py, which includes a function `repeat_text()` that returns a string multiplied by an integer. The code includes assertions to verify the function's behavior and a print statement. The bottom panel shows the terminal output, indicating that all tests passed.

```
Lab7.5 > Task13.py > ...
1 #Detect why string multiplication by float fails and fix it.
2 def repeat_text():
3     return "Hello" * int(2.5)
4
5 assert repeat_text() == "HelloHello"
6 assert "Hi" * int(3.0) == "HiHiHi"
7 assert isinstance(repeat_text(), str)
8
9 print("All tests passed.")
10
11
12 #explanation: The error occurs because you cannot multiply a string by a float.
13 # To fix this, we can convert the float to an integer using the int() function
14 # before multiplying the string. This way, we can successfully repeat the
15 # string the desired number of times without encountering a type error.
```

```
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task12.py'
on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task12.py'
All tests passed.
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding>
```

Task14:



The screenshot shows the VS Code interface with the Explorer panel on the left displaying a file tree under 'AI ASSISTANT CODING'. The file 'Task14.py' is selected. The main editor shows the code for Task14.py, which includes a function `compute()` that returns a value plus 10. The code includes assertions to verify the function's behavior and a print statement. The bottom panel shows the terminal output, indicating that all tests passed.

```
Lab7.5 > Task14.py > ...
1 # Explain why NoneType cannot be added and fix by assigning default value.
2 def compute():
3     value = 0 # default value instead of None
4     return value + 10
5
6 assert compute() == 10
7 assert 5 + 10 == 15
8 assert isinstance(compute(), int)
9
10 print("All tests passed.")
11 #Explanation: The error occurs because NoneType cannot be added to an integer.
12 # To fix this, we can assign a default value (like 0) to the variable instead of None.
13 # This way, we can perform the addition without encountering a TypeError.
14
15
```

```
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding> c:: cd 'c:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task12.py'
on.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\SPURTHI\OneDrive\Desktop\Ai Assistant Coding\Lab7.5\Task12.py'
All tests passed.
PS C:\Users\SPURTHI\OneDrive\Desktop\Ai Assistant Coding>
```

Task15

EXPLORER

AI ASSISTANT CODING

Lab 6

Lab3.1

Lab5

Lab7.5

Task1.py

Task2.py

Task3.py

Task4.py

Task5.py

Task6.py

Task7.py

Task8.py

Task9.py

Task10.py

Task11.py

Task12.py

Task13.py

Task14.py

Task15.py

log.txt

OddEven.py

Prime.py

String.py

user_activity.log

Task14.py

Task15.py

Task13.py

Task12.py

Task11.py

Task10.py

Task9.py

Task8.py

Task7.py

Lab7.5 > Task15.py > ...

1 #Explain why input() returns string and fix using type conversion.

2 # The input() function in Python always returns a string because it is designed to read user input as text. If you want to use

3

4 def sum_two_numbers(a, b):

5 return int(a) + int(b)

6

7 assert sum_two_numbers("5", "3") == 8

8 assert sum_two_numbers("10", "2") == 12

9 assert sum_two_numbers("0", "7") == 7

10

11 print("All tests passed.")

12

13 #explanation: In the function 'sum_two_numbers', we use the int()

14 # function to convert the string inputs 'a' and 'b' into integers

15 # before performing the addition. This allows us to correctly sum

16 # the two numbers even though they were originally provided as strings.

17

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

PS c:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding> c; cd 'c:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding'; & 'c:\Python314\python.exe' 'c:\Users\SPURTHI\.vscode\extensions\ms-python.debugpy-2025.19.2026012701-win32-x64\bundled\libs\debugpy\launcher' '54357' '...' 'c:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding\Lab7.5\Task12.py'

All tests passed.

PS c:\Users\SPURTHI\OneDrive\Desktop\AI Assistant Coding>