

ASSIGNMENT-3.4

2303A51546

Batch-10

TASK-1:

Prompt:

Write a Python function that prints the first N Fibonacci numbers.

CODE:

```
def fibonacci(n):
```

```
    a, b = 0, 1
```

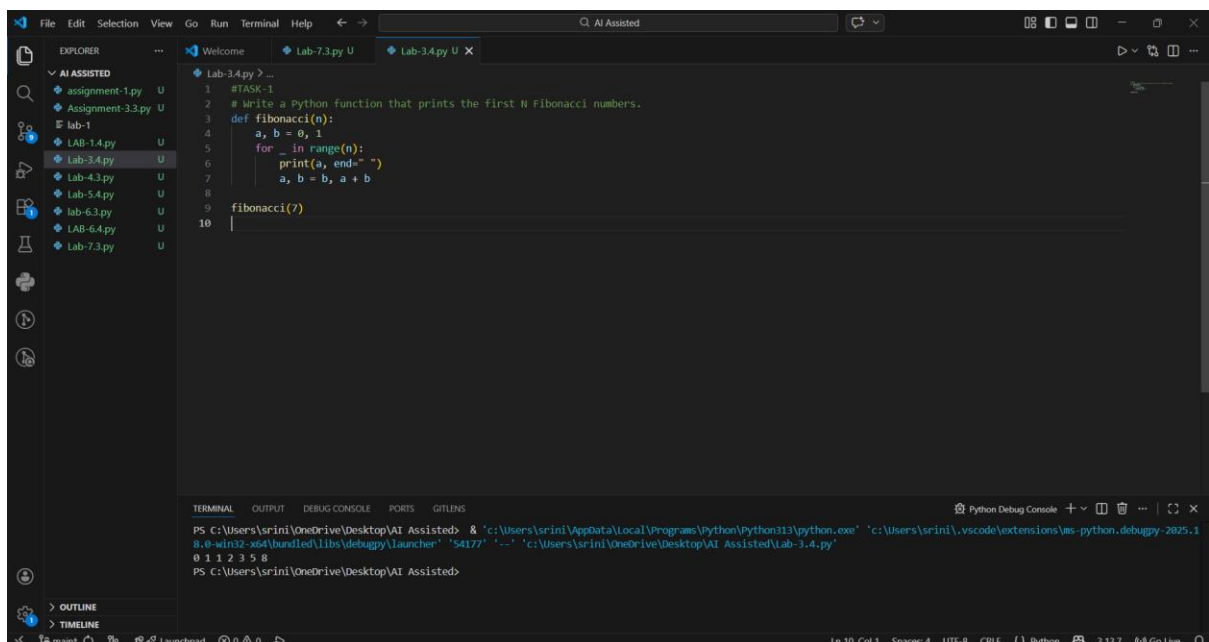
```
    for _ in range(n):
```

```
        print(a, end=" ")
```

```
        a, b = b, a + b
```

```
fibonacci(7)
```

Output:



The screenshot shows a Visual Studio Code editor with a Python file named 'Lab-3.4.py'. The code defines a function 'fibonacci(n)' that prints the first 'n' Fibonacci numbers. The function uses a loop to calculate the sequence, starting with 'a=0' and 'b=1', and updates them using tuple swapping. The function is called with 'fibonacci(7)'. The terminal at the bottom shows the command being executed and the output: '0 1 1 2 3 5 8'.

```
1 #TASK-1
2 # Write a Python function that prints the first N Fibonacci numbers.
3 def fibonacci(n):
4     a, b = 0, 1
5     for _ in range(n):
6         print(a, end=" ")
7         a, b = b, a + b
8
9 fibonacci(7)
10
```

```
PS C:\Users\srin\OneDrive\Desktop\AI Assisted> & 'c:\Users\srin\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\srin\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '54177' '--' 'c:\Users\srin\OneDrive\Desktop\AI Assisted\Lab-3.4.py'
0 1 1 2 3 5 8
PS C:\Users\srin\OneDrive\Desktop\AI Assisted>
```

Analysis:

The function `fibonacci(n)` prints the first N Fibonacci numbers using a loop. It starts with 0 and 1, then keeps updating values using tuple swapping.

TASK-2:

Prompt:

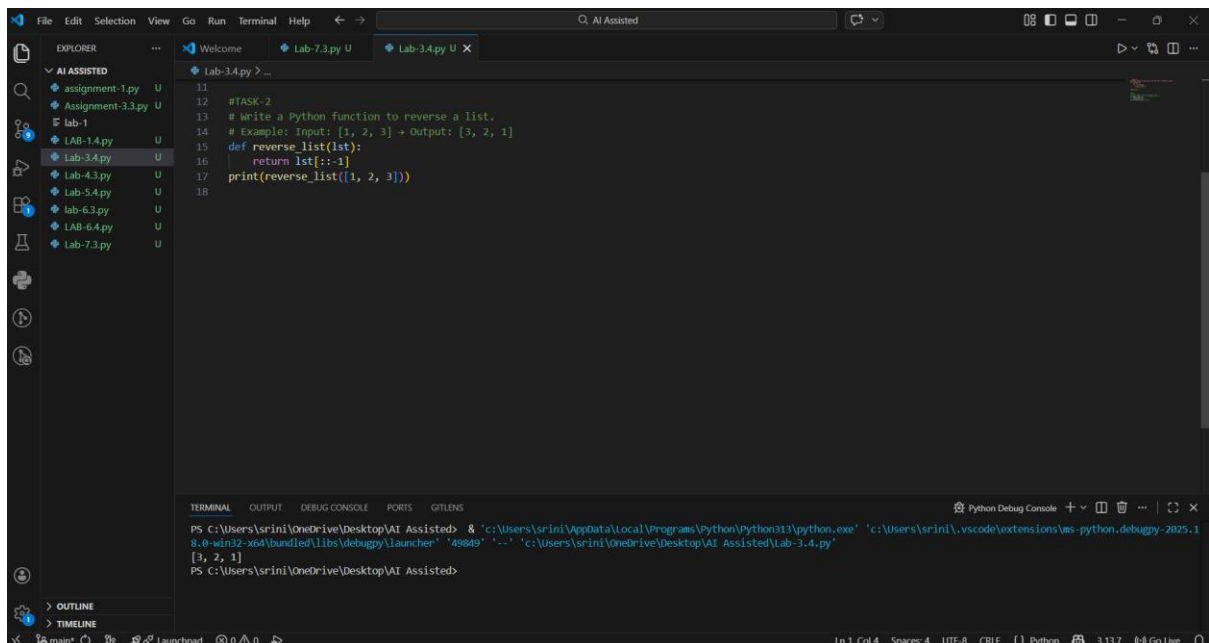
Write a Python function to reverse a list.

Example: Input: [1, 2, 3] → Output: [3, 2, 1]

CODE:

```
def reverse_list(lst):  
    return lst[::-1]  
  
print(reverse_list([1, 2, 3]))
```

Output:



The screenshot shows a VS Code editor window with a file explorer on the left containing several Python files. The main editor area displays the code for Task 2: a function `reverse_list` that returns a list reversed using slicing, and a print statement. The terminal at the bottom shows the command to run the script and the output `[3, 2, 1]`.

```
#TASK-2  
# Write a Python function to reverse a list.  
# Example: Input: [1, 2, 3] + Output: [3, 2, 1]  
def reverse_list(lst):  
    return lst[::-1]  
print(reverse_list([1, 2, 3]))
```

```
PS C:\Users\sriniv\OneDrive\Desktop\AI Assisted> & "c:\Users\sriniv\AppData\Local\Programs\Python\Python313\python.exe" "c:\Users\sriniv\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher" "49849" "-" "c:\Users\sriniv\OneDrive\Desktop\AI Assisted\Lab-3.4.py"  
[3, 2, 1]  
PS C:\Users\sriniv\OneDrive\Desktop\AI Assisted>
```

Analysis:

The function `reverse_list(lst)` reverses a list using Python slicing.

TASK-3:

Prompt:

Write a function `is_valid(s)` that returns True if a string starts with a capital letter and ends with a period.

Examples:

"Hello." → True

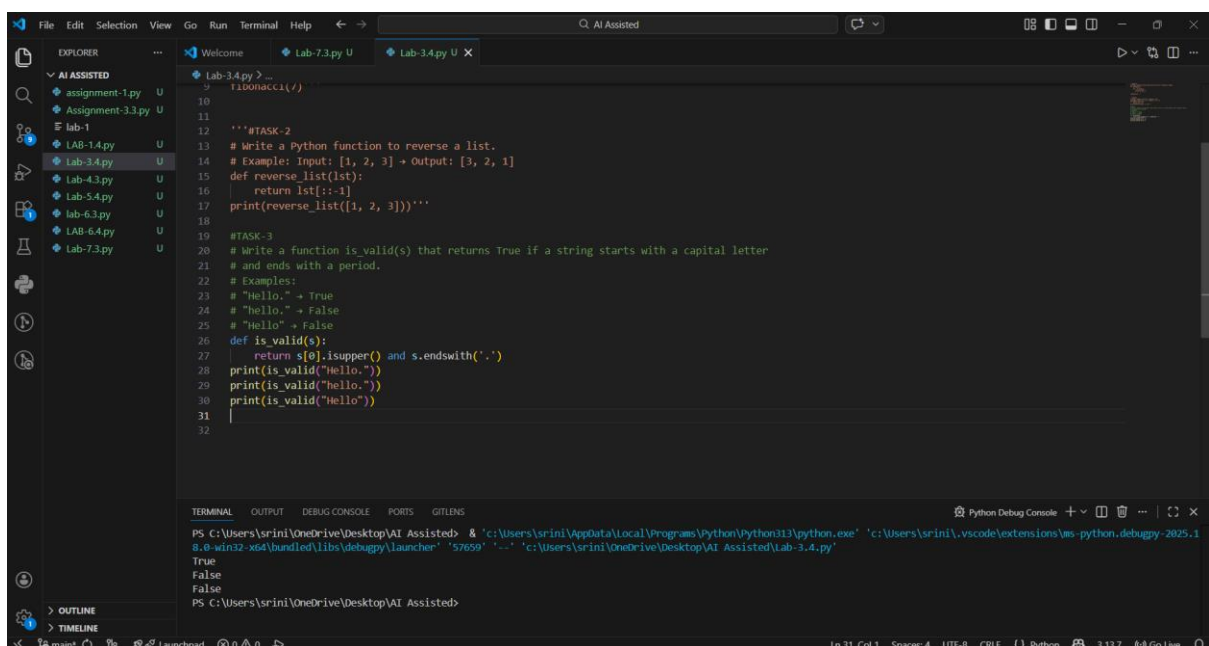
"hello." → False

"Hello" → False

CODE:

```
def is_valid(s):  
    return s[0].isupper() and s.endswith('.')  
  
print(is_valid("Hello."))  
print(is_valid("hello."))  
print(is_valid("Hello"))
```

Output:



```
Lab-3.4.py 2  
10 fibonacci(r)  
11  
12 '''TASK-2  
13 # Write a Python function to reverse a list.  
14 # Example: Input: [1, 2, 3] → Output: [3, 2, 1]  
15 def reverse_list(lst):  
16     return lst[::-1]  
17     print(reverse_list([1, 2, 3]))'''  
18  
19 #TASK-3  
20 # Write a function is_valid(s) that returns True if a string starts with a capital letter  
21 # and ends with a period.  
22 # Examples:  
23 # "Hello." → True  
24 # "hello." → False  
25 # "Hello" → False  
26 def is_valid(s):  
27     return s[0].isupper() and s.endswith('.')  
28     print(is_valid("Hello."))  
29     print(is_valid("hello."))  
30     print(is_valid("Hello"))  
31  
32  
TERMINAL  
PS C:\Users\sriini\OneDrive\Desktop\AI Assisted> & 'c:\Users\sriini\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\sriini\.vscode\extensions\ms-python.debugpy-2025.1  
8.0-win32-x64\bundle\libs\debugpy\launcher' '57659' '--' 'c:\Users\sriini\OneDrive\Desktop\AI Assisted\Lab-3.4.py'  
True  
False  
False  
PS C:\Users\sriini\OneDrive\Desktop\AI Assisted>
```

Analysis:

The function checks two conditions:

First letter is uppercase → `s[0].isupper()`

String ends with a period → `s.endswith('.')`

TASK-4:

Prompt:

Write a python function to validate an email address

CODE:

```
import re  
  
def is_valid_email(email):
```

```
pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
```

```
return re.match(pattern, email) is not None
```

```
if __name__ == "__main__":
```

```
    email = input("Enter an email address: ")
```

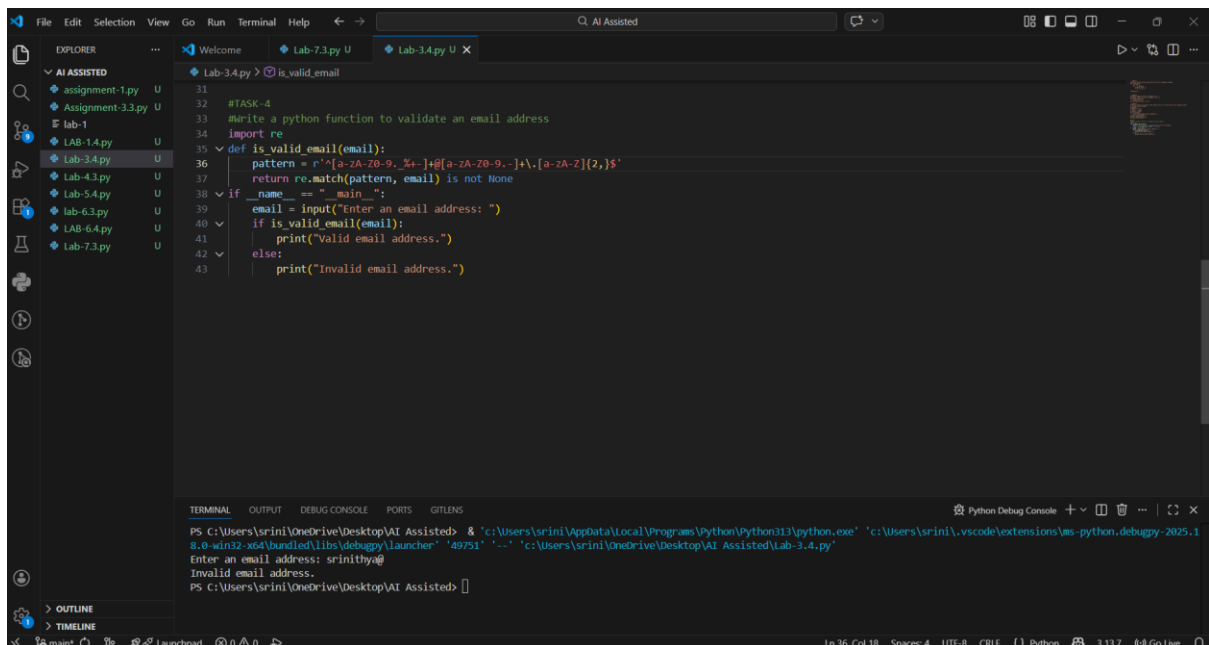
```
    if is_valid_email(email):
```

```
        print("Valid email address.")
```

```
    else:
```

```
        print("Invalid email address.")
```

Output:



The screenshot shows a VS Code editor with a Python file named 'Lab-3.4.py'. The code defines a function 'is_valid_email' that uses a regular expression to validate email addresses. The main block prompts the user to enter an email address and prints the result. The terminal output shows the user entering 'srinithya@' and receiving the message 'Invalid email address.'.

```
31
32 #TASK-4
33 #Write a python function to validate an email address
34 import re
35 def is_valid_email(email):
36     pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
37     return re.match(pattern, email) is not None
38 if __name__ == "__main__":
39     email = input("Enter an email address: ")
40     if is_valid_email(email):
41         print("Valid email address.")
42     else:
43         print("Invalid email address.")
```

```
PS C:\Users\srinithya\OneDrive\Desktop\AI Assisted> & 'c:\Users\srinithya\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\srinithya\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\lib\debugpy\launcher' '49751' '-.' 'c:\Users\srinithya\OneDrive\Desktop\AI Assisted\Lab-3.4.py'
Enter an email address: srinithya@
Invalid email address.
PS C:\Users\srinithya\OneDrive\Desktop\AI Assisted>
```

Analysis:

The function uses regular expressions (regex) to validate email format.

TASK-5:

Prompt:

Write a function to return sum of digits of a number

CODE:

```
def sum_of_digits(n):
```

```
    total = 0
```

```
    while n > 0:
```

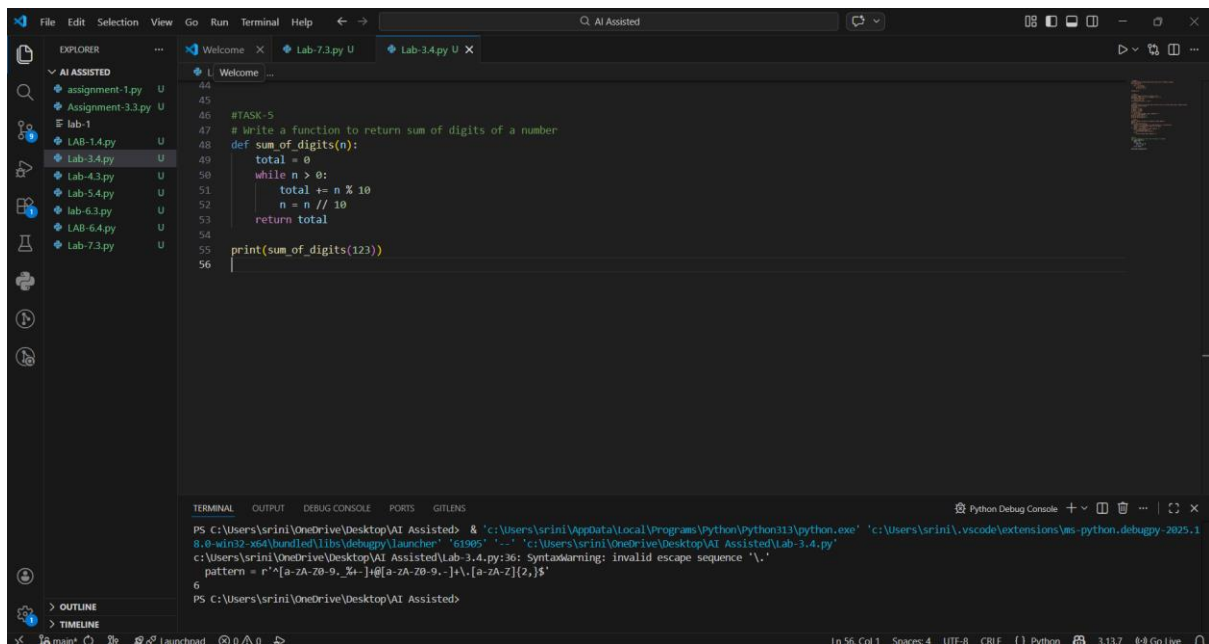
```
total += n % 10
```

```
n = n // 10
```

```
return total
```

```
print(sum_of_digits(123))
```

Output:



The screenshot shows a Visual Studio Code editor window with a Python file named 'Lab-3.4.py' open. The code defines a function 'sum_of_digits(n)' that calculates the sum of digits of a number 'n' using a while loop. The function uses modulus (%) to extract the last digit and integer division (//) to remove it. The function is then called with the argument 123, and the result is printed. The terminal at the bottom shows the command to run the script, which executes successfully, displaying the output '6'.

```
def sum_of_digits(n):
    total = 0
    while n > 0:
        total += n % 10
        n = n // 10
    return total

print(sum_of_digits(123))
```

```
PS C:\Users\sirini\OneDrive\Desktop\AI Assisted> & 'c:\Users\sirini\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\sirini\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '61905' '-.' 'c:\Users\sirini\OneDrive\Desktop\AI Assisted\Lab-3.4.py'
c:\Users\sirini\OneDrive\Desktop\AI Assisted\Lab-3.4.py:36: SyntaxWarning: invalid escape sequence '\.'
  pattern = r'^[a-zA-Z0-9_]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
6
PS C:\Users\sirini\OneDrive\Desktop\AI Assisted>
```

Analysis:

The function extracts digits using modulus (% 10) and division (// 10) until the number becomes 0.