

## LAB ASSIGNMENT – 1.5

Hall Ticket No.: 2303A510E6

Batch-29

### Task 1: AI-Generated Logic Without Modularization (String Reversal Without Functions)

Prompt:

### AI-Generated Logic Without Modularization (String Reversal Without Functions)

Code:

The screenshot displays the Visual Studio Code environment with a Python file named `aac1.py` open in the editor. The file contains a function `reverse_string` that takes a string `s` and returns its reverse. The main code calls this function with the string `"Hello, World!"` and prints the result.

The interface includes a sidebar on the left with the following sections:

- RUN AND DEBUG**: Contains a "Run and Debug" button and instructions on how to customize run and debug settings.
- BREAKPOINTS**: Contains a "Raised Exceptions" checkbox and a "User Uncaught Exception" checkbox.

The main editor shows the code for `aac1.py`:

```
1 String Reversal Without Functions
2 input_string = "Hello, World!"
3 reversed_string = ""
4 for char in input_string:
5     reversed_string = char + reversed_string
6     print(reversed_string)
7
8 """ Readability Improvement)
9 def reverse_string(s):
10     reversed_s = ""
11     for char in s:
12         reversed_s = char + reversed_s
13     return reversed_s
14 input_string = "Hello, World!"
15 print(reverse_string(input_string))"""
16
```

The bottom panel shows the "TERMINAL" tab with the following output:

```
python aac1.py
Hello, World!
```

The status bar at the bottom indicates the file is at line 1, column 1, with 4 spaces, in UTF-8 encoding, with a CRLF line ending, and the Python interpreter is set to the default.

## Task 2: Efficiency & Logic Optimization (Readability Improvement)

Prompt:

### Efficiency & Logic Optimization (Readability Improvement)

Code:

The screenshot shows the Visual Studio Code interface with the following components:

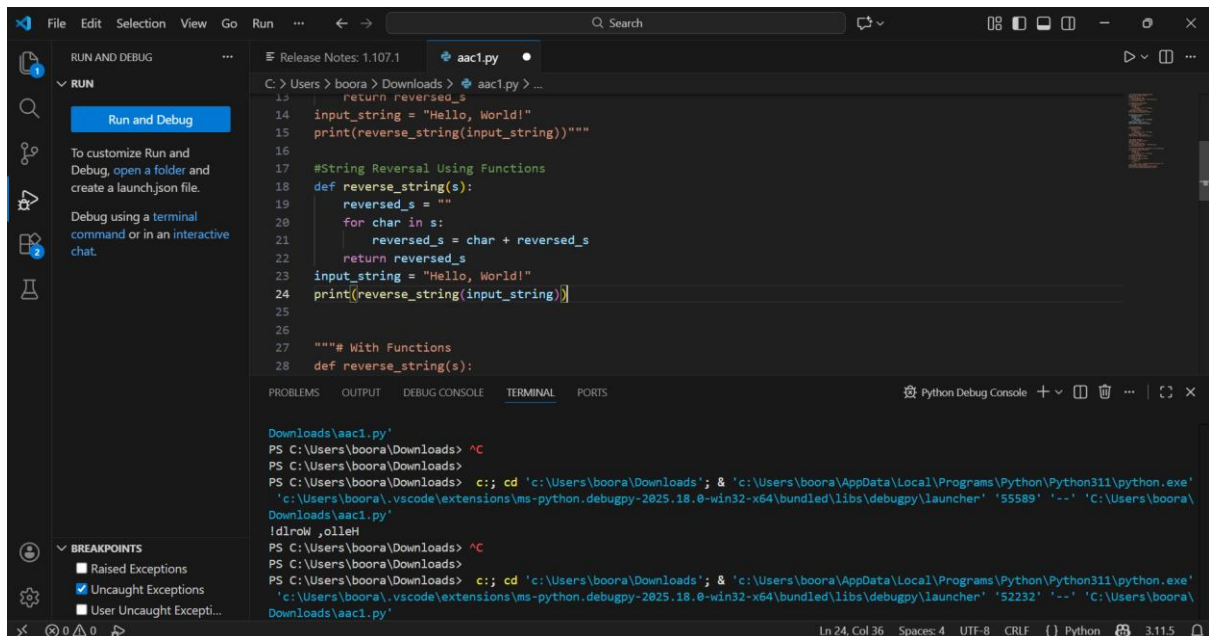
- Menu Bar:** File, Edit, Selection, View, Go, Run, ...
- Search Bar:** Q Search
- Run and Debug Sidebar:**
  - RUN AND DEBUG:** Contains a 'Run and Debug' button and instructions: 'To customize Run and Debug, open a folder and create a launch.json file.' and 'Debug using a terminal command or in an interactive chat.'
  - BREAKPOINTS:** Contains a list of breakpoints: 'Raised Exceptions', 'Uncaught Exceptions', and 'User Uncaught Exceptions'.
- Editor:**
  - File Explorer:** Shows the file 'aac1.py'.
  - Code Editor:** Displays the content of 'aac1.py', which is a Python script for string reversal. The script includes a function 'reverse\_string(s)' and a main block that takes user input and prints the reversed string.
- Terminal:** Shows the command 'python aac1.py' and the output 'Hello, World!'.

### Task 3: Modular Design Using AI Assistance (String Reversal Using Functions)

Prompt:

Modular Design Using AI Assistance (String Reversal Using Functions)

Code:



The screenshot shows the Visual Studio Code editor with a Python file named `aac1.py`. The code implements a string reversal function using a loop. The terminal output shows the command to run the script and the resulting output.

```
14 input_string = "Hello, World!"
15 print(reverse_string(input_string))"""
16
17 #String Reversal Using Functions
18 def reverse_string(s):
19     reversed_s = ""
20     for char in s:
21         reversed_s = char + reversed_s
22     return reversed_s
23 input_string = "Hello, World!"
24 print(reverse_string(input_string))
25
26
27 """# With Functions
28 def reverse_string(s):
```

Terminal Output:

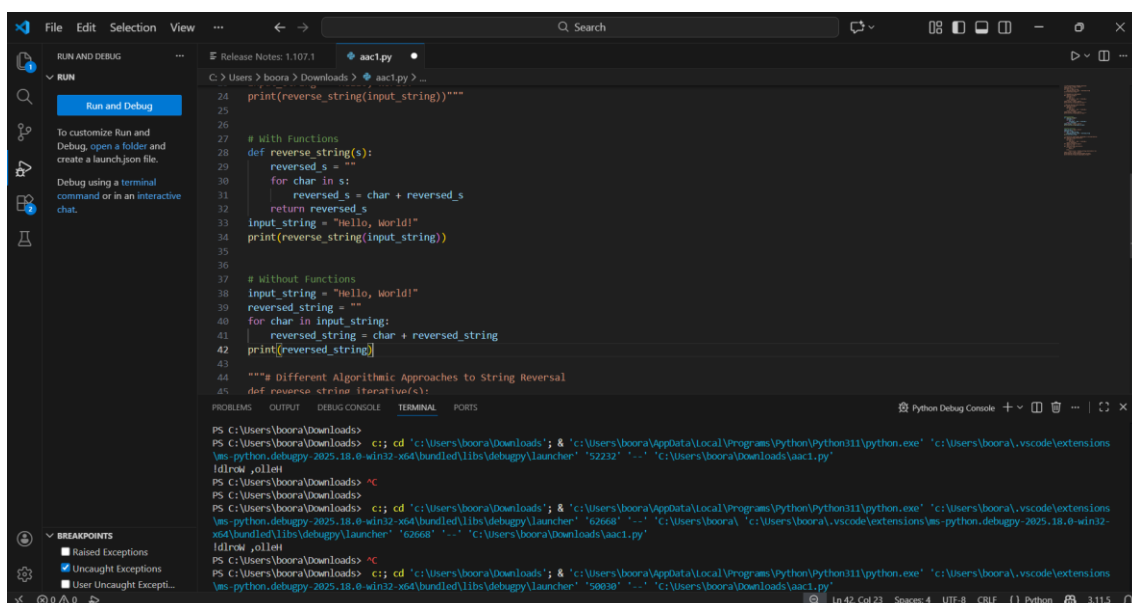
```
Downloads\aac1.py'
PS C:\Users\boora\Downloads> ^C
PS C:\Users\boora\Downloads>
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe'
'c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '55589' '-' 'c:\Users\boora\
Downloads\aac1.py'
ldrow ,olleH
PS C:\Users\boora\Downloads> ^C
PS C:\Users\boora\Downloads>
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe'
'c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '52232' '-' 'c:\Users\boora\
Downloads\aac1.py'
ldrow ,olleH
PS C:\Users\boora\Downloads> ^C
PS C:\Users\boora\Downloads>
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe'
'c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '50030' '-' 'c:\Users\boora\Downloads\aac1.py'
```

### Task 4: Comparative Analysis – Procedural vs Modular Approach (With vs Without Functions)

Prompt:

Comparative Analysis – Procedural vs Modular Approach (With vs Without Functions)

Code:



The screenshot shows the Visual Studio Code editor with a Python file named `aac1.py`. The code implements two different approaches to string reversal: a procedural approach and a modular approach using functions. The terminal output shows the command to run the script and the resulting output.

```
24 print(reverse_string(input_string))"""
25
26
27 # With Functions
28 def reverse_string(s):
29     reversed_s = ""
30     for char in s:
31         reversed_s = char + reversed_s
32     return reversed_s
33 input_string = "Hello, World!"
34 print(reverse_string(input_string))
35
36
37 # Without Functions
38 input_string = "Hello, World!"
39 reversed_string = ""
40 for char in input_string:
41     reversed_string = char + reversed_string
42 print(reversed_string)
43
44 """# Different Algorithmic Approaches to String Reversal
45 def reverse_string_iterative(s):
```

Terminal Output:

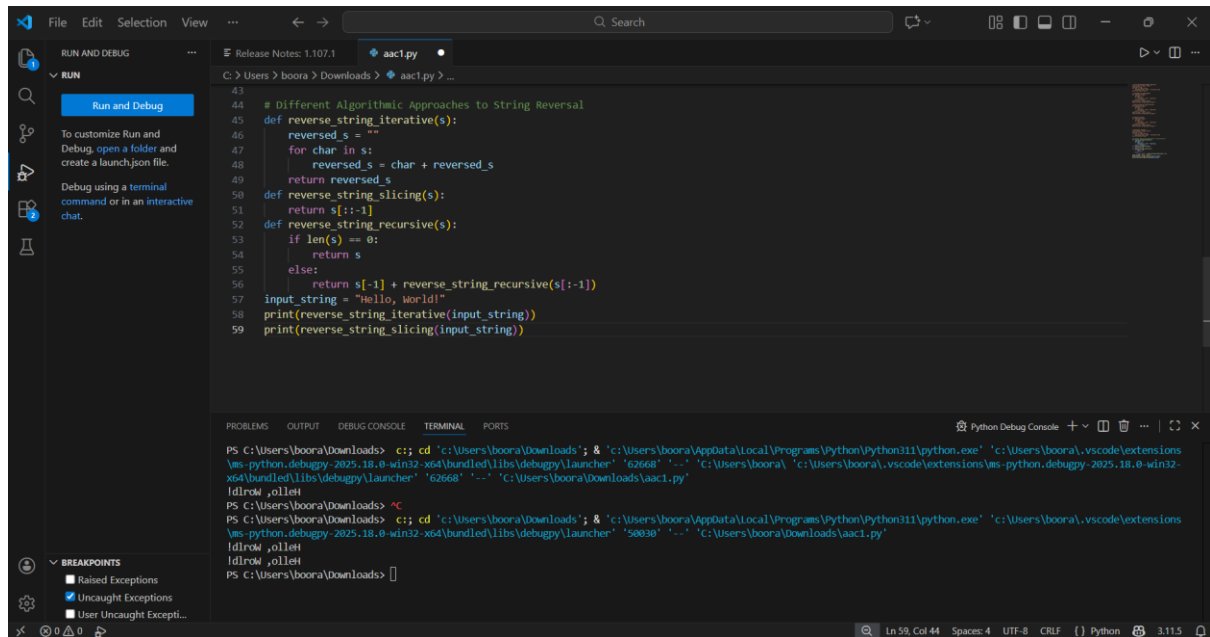
```
PS C:\Users\boora\Downloads>
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions
\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '52232' '-' 'c:\Users\boora\Downloads\aac1.py'
ldrow ,olleH
PS C:\Users\boora\Downloads> ^C
PS C:\Users\boora\Downloads>
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions
\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '6268' '-' 'c:\Users\boora\Downloads\aac1.py'
ldrow ,olleH
PS C:\Users\boora\Downloads> ^C
PS C:\Users\boora\Downloads> c;; cd 'c:\Users\boora\Downloads'; & 'c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\boora\.vscode\extensions
\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '50030' '-' 'c:\Users\boora\Downloads\aac1.py'
```

## Task 5: AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches to String Reversal)

Prompt:

AI-Generated Iterative vs Recursive Fibonacci Approaches (Different Algorithmic Approaches to String Reversal)

Code:



The screenshot shows a Visual Studio Code editor with a Python file named `aac1.py` open. The code implements three methods for string reversal: `reverse_string_iterative`, `reverse_string_slicing`, and `reverse_string_recursive`. The `reverse_string_iterative` method uses a loop to build the reversed string. The `reverse_string_slicing` method uses slicing to reverse the string. The `reverse_string_recursive` method uses recursion to reverse the string. The code also includes a main function that takes an input string "Hello, World!" and prints the results of the three reversal methods.

```
43
44 # Different Algorithmic Approaches to String Reversal
45 def reverse_string_iterative(s):
46     reversed_s = ""
47     for char in s:
48         reversed_s = char + reversed_s
49     return reversed_s
50 def reverse_string_slicing(s):
51     return s[::-1]
52 def reverse_string_recursive(s):
53     if len(s) == 0:
54         return s
55     else:
56         return s[-1] + reverse_string_recursive(s[:-1])
57 input_string = "Hello, World!"
58 print(reverse_string_iterative(input_string))
59 print(reverse_string_slicing(input_string))
```

The terminal output shows the execution of the script, displaying the reversed string "dlroW ,olleH" for both the iterative and slicing methods, and the recursive method is also shown.

```
PS C:\Users\boora\Downloads> c:\cd "c:\Users\boora\Downloads"; & "c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe" "c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher" "62668" "--" "c:\Users\boora\Downloads\aac1.py"
dlroW ,olleH
PS C:\Users\boora\Downloads> c:\cd "c:\Users\boora\Downloads"; & "c:\Users\boora\AppData\Local\Programs\Python\Python311\python.exe" "c:\Users\boora\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher" "50830" "--" "c:\Users\boora\Downloads\aac1.py"
dlroW ,olleH
PS C:\Users\boora\Downloads>
```