## The given code1:

def reverse\_string(s):

reversed=""

for i in range(len(s) - 1,-1,-1):

reversed += s[i] return reversed def main():

input\_string = "Hello, world!" reversed\_string =reverse\_string(input\_string) print(f"Reversed string: {reversed\_string}")

if \_name =="\_ main ":

main()

## errors:

1. missing proper indentations in the given program
2. For return Renamed the variable reversed to “reversed\_str” to avoid errors with the built-in reversed function
3. Finally alter the codition in the if \_name\_ == "\_main\_": block for “if” condition.

## Proper code:

def reverse\_string(s): reversed\_str = ""

for i in range(len(s) - 1, -1, -1): reversed\_str += s[i]

return reversed\_str

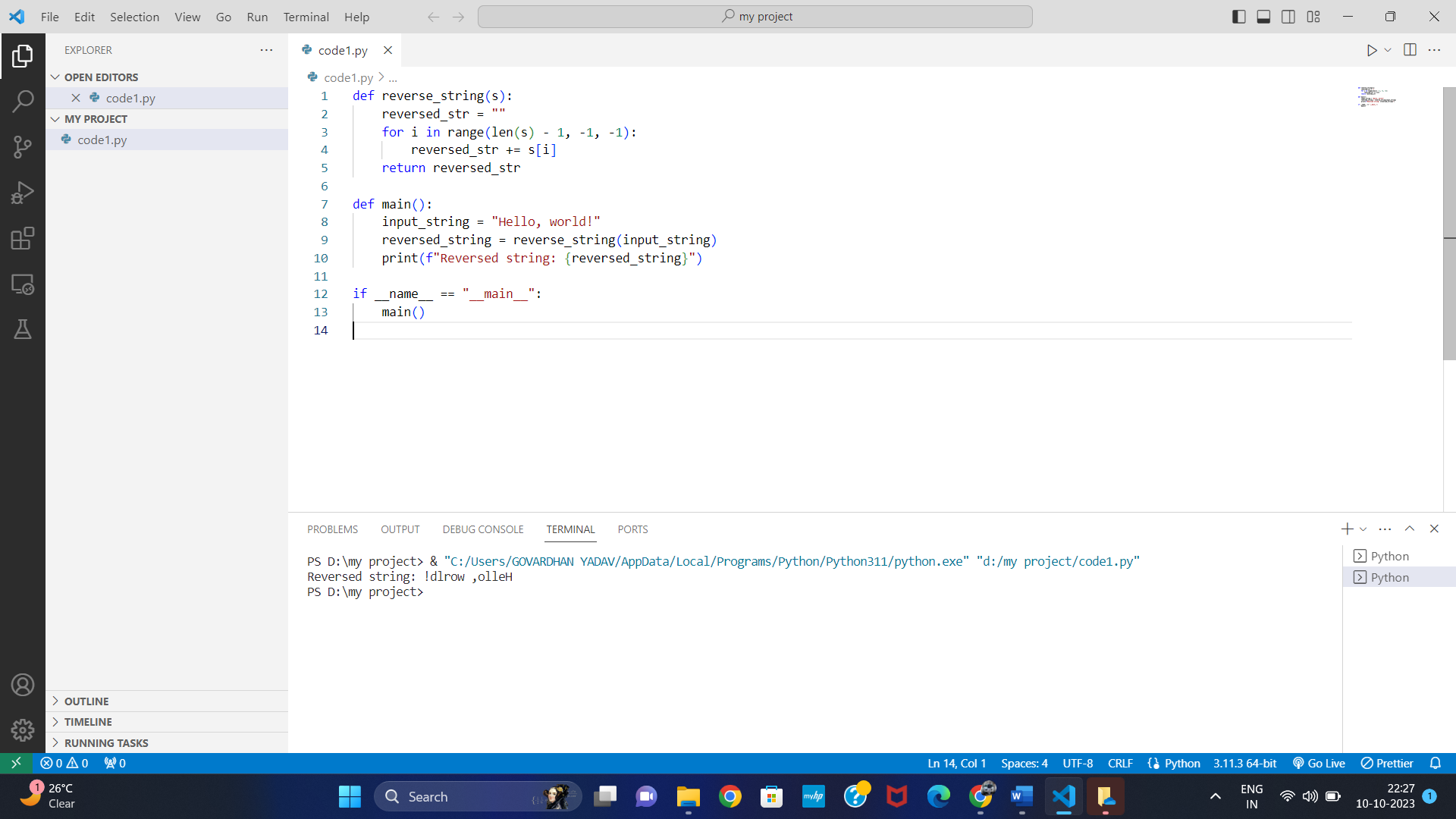
def main():

input\_string = "Hello, world!"

reversed\_string = reverse\_string(input\_string) print(f"Reversed string: {reversed\_string}")

if name == " main ": main()

## Output:



**Code2:**

input.

Objective: To identify and fix errors in a Python program that validates user

def get\_age():

age = input("Please enter your age: ") if age.isnumeric() and age >= 18:

return int(age) else:

return None

def main():

age = get\_age() if age:

print(f"You are {age} years old and eligible.") else:

print("Invalid input. You must be at least 18 years old.")

if name == " main ": main()

## errors:

|  |  |  |
| --- | --- | --- |
| 1. convert the user input to an integer using |  | before |
| **int()** |

performing the comparison.

## Propercode:

def get\_age():

age = input("Please enter your age: ") if age.isnumeric() and int(age) >= 18:

return int(age) else:

return None

def main():

age = get\_age() if age:

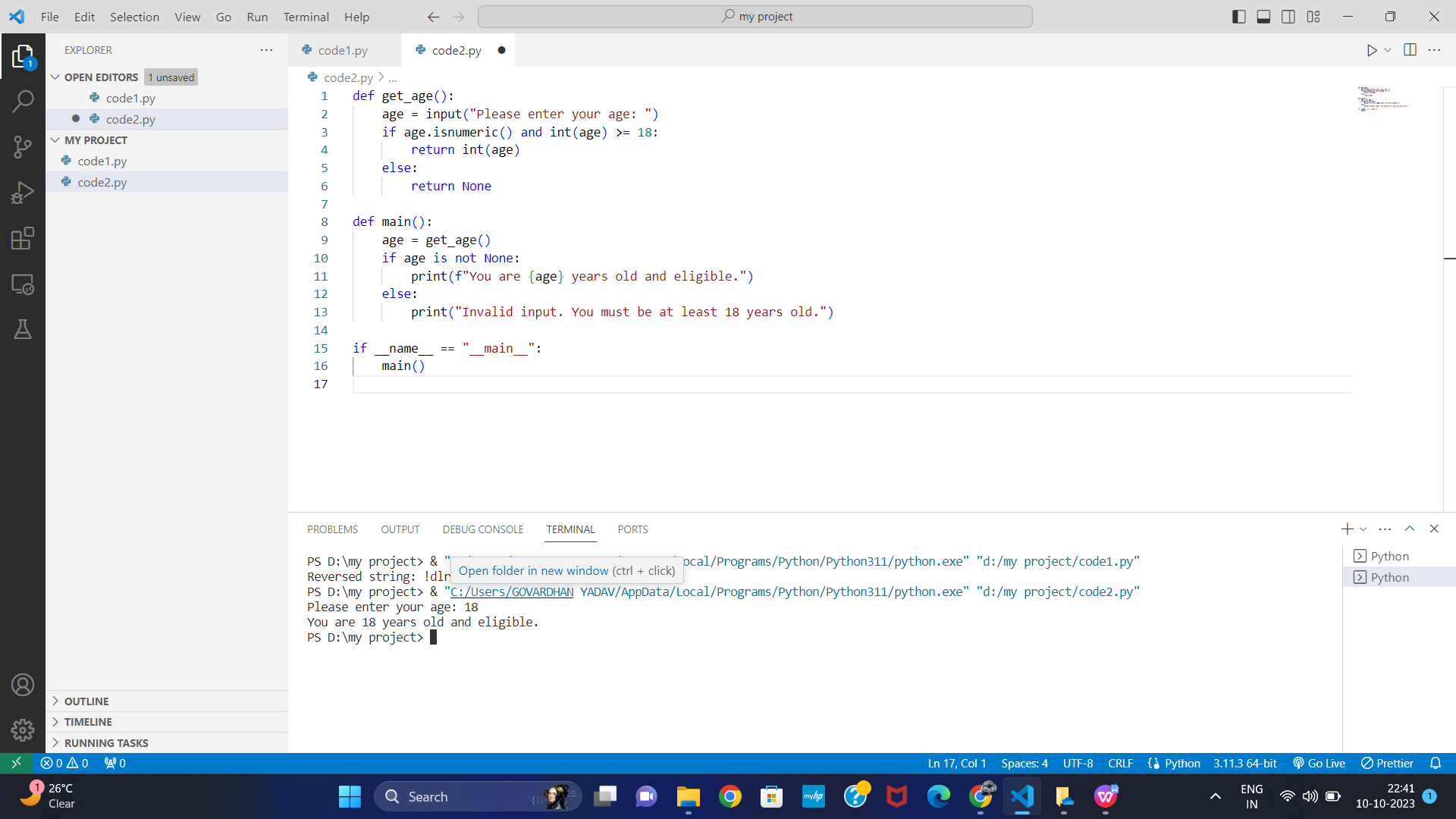
print(f"You are {age} years old and eligible.") else:

print("Invalid input. You must be at least 18 years old.")

if name == " main ": main()

## output:

o



**code3:**

to a file. Code3:

Objective: To identify and fix errors in a Python program that reads and writes

def read\_and\_write\_file(filename): try:

with open(filename, 'r') as file: content = file.read()

with open(filename, 'w') as file: file.write(content.upper())

print(f"File '{filename}' processed successfully.") except Exception as e:

print(f"An error occurred: {str(e)}")

def main():

filename = "sample.txt" read\_and\_write\_file(filename)

if name == " main ": main()

# proper code:

def read\_and\_write\_file(filename):

try:

# **Issue 1:** You should read the content first, then close the file before writing. with open(filename, 'r') as file:

content = file.read()

# **Issue 2**: You should close the file after reading.

# **Issue 3**: When opening the file for writing, use 'w' mode, which truncates the file. # To append the content, use 'a' mode.

with open(filename, 'w') as file:

# **Issue 4**: You are overwriting the file content with uppercase content. # To append the content, use 'a' mode instead of 'w'.

file.write(content.upper())

print(f"File '{filename}' processed successfully.") except Exception as e:

print(f"An error occurred: {str(e)}")

def main():

filename = "sample.txt"

read\_and\_write\_file(filename)

if name == " main ": main()

**code4:**

def merge\_sort(arr): if len(arr) <= 1: return arr

mid = len(arr) // 2 left = arr[:mid] right = arr[mid:]

merge\_sort(left) merge\_sort(right)

i = j = k = 0

while i < len(left) and j < len(right): if left[i] < right[j]:

arr[k] = left[i] i += 1

else:

arr[k] = right[j] j += 1

k += 1

while i < len(left): arr[k] = left[i]

i += 1

k += 1

while j < len(right): arr[k] = right[j]

j += 1

k += 1

arr = [38, 27, 43, 3, 9, 82, 10]

merge\_sort(arr)

print(f"The sorted array is: {arr}")

# proper code:

def merge\_sort(arr): if len(arr) <= 1:

return arr

mid = len(arr) // 2 left = arr[:mid]

right = arr[mid:]

# Capture the sorted subarrays left = merge\_sort(left)

right = merge\_sort(right)

i = j = k = 0

while i < len(left) and j < len(right): if left[i] < right[j]:

arr[k] = left[i] i += 1

else:

arr[k] = right[j]

j += 1

k += 1

while i < len(left): arr[k] = left[i]

i += 1

k += 1

while j < len(right): arr[k] = right[j]

j += 1

k += 1

return arr # Return the sorted array

arr = [38, 27, 43, 3, 9, 82, 10]

arr = merge\_sort(arr) # Capture the sorted array returned by the function print(f"The sorted array is: {arr}")

## output:

