**Code 1:**

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()

The code appears to be correct and should work without errors. It defines a reverse\_string function that reverses a given string and a main function that demonstrates the usage of reverse\_string. There are no syntax errors or logical errors in the code.

**ouput:** Reversed string: !dlrow ,olleH

**Code 2:**

The error in the code is because the age variable is returned as a string from the get\_age() function, but you are comparing it with an integer (18) in the if statement, which could lead to a TypeError.

The corrected code is:

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:**

Please enter your age: 21

You are 21 years old and eligible.

**Code 3:**

The code opens the file for writing inside the with block after opening it for reading. This will truncate the file and erase its content before reading it, resulting in the content being lost. The code reads the content and converts it to uppercase but doesn't handle potential exceptions that may occur when opening or manipulating the file.

The corrected code is:

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

try:

# Open the file for reading

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

content = file.read()

# Open the same file for writing (this will overwrite the content)

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

# Write the content in uppercase to the file

file.write(content.upper())

# Print a success message

print(f"File '{filename}' processed successfully.")

except Exception as e:

# Handle any exceptions that might occur

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

def main():

filename = "sample.txt"

read\_and\_write\_file(filename)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Code 4:**

The issue is that the merge\_sort function is not returning any value, and it relies on modifying the input arr in-place. However, in the provided code, you are not returning the sorted array explicitly. To fix this, you should add a return statement to return the sorted arr.

The corrected code is:

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

return arr # Add this line to return the sorted array

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

sorted\_arr = merge\_sort(arr) # Store the sorted array in a variable

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

Ouput: The sorted array is: [3, 9, 10, 27, 38, 43, 82]