AI ASSISTANT CODING

ASSIGNMENT 7.5

H.NO:2303A54038

TASK 1:

```
lab 7.py > ...
1   # Bug: Mutable default argument
2   def add_item(item, items=[]):
3       items.append(item)
4       return items
5   print(add_item(1))
6   print(add_item(2))
7
8
9   #after fixing the bug
0
1   # Bug: Mutable default argument
2   def add_item(item, items=None):
3       if items is None:
4           items = []
5       items.append(item)
6       return items
7   print(add_item(1))
8   print(add_item(2))
9
```

```
PS C:\AI assitant>
r' '65530' '--' 'C
[1]
[2]
```

TASK 2:

```
# Bug: Floating point precision issue
def check_sum():
    return (0.1 + 0.2) == 0.3
print(check_sum())

#after fixing the bug

# Bug: Floating point precision issue
import math
def check_sum():
    return math.isclose(0.1 + 0.2, 0.3)
print(check_sum())
```

OUTPUT:

```
PS C:\AI assitant>
True
```

TASK 3:

```python
#TASK 3

# Bug: No base case
def countdown(n):
    print(n)
    return countdown(n-1)
countdown(5)

#after fixing the bug

# Bug: No base case
def countdown(n):
    print(n)
    if n == 0:
        return
    return countdown(n-1)
countdown(5)
```

OUTPUT:

```
PS C:\AI assitant> & C
5
4
3
2
1
0
PS C:\AI assitant>
```

TASK 4:

```python
#TASK 4
# Bug: Accessing non-existing key
def get_value():
    data = {"a": 1, "b": 2}
    return data["c"]
print(get_value())

#after fixing the bug

# Bug: Accessing non-existing key
def get_value():
    data = {"a": 1, "b": 2}
    try:
        return data["c"]
    except KeyError:
        return "Key 'c' does not exist"
print(get_value())
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
Key 'c' does not exist
```

TASK 5:

```python
#TASK 5
# Bug: Infinite loop
def loop_example():
    i = 0
    while i < 5:
        print(i)

#after fixing the bug

# Bug: Infinite loop
def loop_example():
    i = 0
    while i < 5:
        print(i)
        i += 1
loop_example()
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Pyth
0
1
2
3
4
```

TASK 6:

```
#TASK 6
# Bug: Wrong unpacking
a, b = (1, 2, 3)
#after fixing the bug
# Bug: Wrong unpacking
a, b, c = (1, 2, 3)
```

TASK7:

```
#TASK 7
# Bug: Mixed indentation
def func():
    x = 5
        y = 10
    return x+y
#after fixing the bug
# Bug: Mixed indentation
def func():
    x = 5
    y = 10
    return x+y
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
4.0
```

TASK 8:

```
#TASK 8
# Bug: Wrong import
import maths
print(maths.sqrt(16))

#after fixing the bug

# Bug: Wrong import
import math
print(math.sqrt(16))
```

TASK 9:

```python
#TASK 9
# Bug: Early return inside loop
def total(numbers):
    for n in numbers:
        return n
print(total([1,2,3]))

#after fixing the bug

# Bug: Early return inside loop
def total(numbers):
    result = 0
    for n in numbers:
        result += n
    return result
print(total([1,2,3]))
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
6
```

TASK 10:

```python
#TASK 10
# Bug: Using undefined variable
def calculate_area():
    return length * width  import width
print(calculate_area())

#after fixing the bug

# Bug: Using undefined variable
def calculate_area(length, width):
    return length * width

print(calculate_area(5, 10))

# Test cases
assert calculate_area(5, 10) == 50
assert calculate_area(0, 10) == 0
assert calculate_area(7, 3) == 21
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
50
```

TASK 11:

```python
#TASK 11
# Bug: Adding integer and string
def add_values():
    return 5 + "10"
print(add_values())

#after fixing the bug

# Bug: Adding integer and string
def add_values():
    return 5 + int("10")
print(add_values())

# Test cases
assert add_values() == 15
assert 5 + int("10") == 15
assert int("5") + int("10") == 15
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
15
```

TASK 12:

```python
#TASK  12

# Bug: Adding string and list
def combine():
    return "Numbers: " + [1, 2, 3]
print(combine())

#after fixing the bug

# Bug: Adding string and list
def combine():
    return "Numbers: " + str([1, 2, 3])
print(combine())
```

Output:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
Numbers: [1, 2, 3]
```

TASK 13:

```python
#TASK 13
# Bug: Multiplying string by float
def repeat_text():
    return "Hello" * 2.5
print(repeat_text())

#after fixing the bug

# Bug: Multiplying string by float
def repeat_text():
    return "Hello" * int(2.5)
print(repeat_text())

# Test cases
assert repeat_text() == "HelloHello"
assert "Hello" * int(3.7) == "HelloHelloHello"
assert "Hi" * int(1.9) == "Hi"
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
HelloHello
```

TASK 14:

```python
#TASK 14
# Bug: Adding None and integer
def compute():
    value = None
    return value + 10

print(compute())


#after fixing the bug


# Bug: Adding None and integer
def compute():
    value = 0  # Default value must be numeric, not None
    return value + 10
print(compute())

# Test cases
assert compute() == 10
assert compute() + 5 == 15
assert compute() - 10 == 0

# Test cases
assert compute() == 10
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
10
```

TASK 15:

```python
#TASK 15
# Bug: Input remains string
def sum_two_numbers():
    a = int(input("Enter first number: "))
    b = int(input("Enter second number: "))
    return a + b

print(sum_two_numbers())

#after fixing the bug

# Bug: Input remains string
def sum_two_numbers():
    a = int(input("Enter first number: "))
    b = int(input("Enter second number: "))
    return a + b

print(sum_two_numbers())
```

OUTPUT:

```
PS C:\AI assitant> & C:/Users/thimi/AppData/Local/Programs/Python/Python313/python.exe "c:/AI assitant/lab 7.py"
Enter first number: 2
Enter second number: 3
5
```