

COMPETITIVE PROGRAMMING LAB ASSIGNMENT - 8.3

Name:J.Sai Kumar || Batch-09 || Ht.no-2303A51562

Task 1: Email Validation using TDD.

Lab-8.3.py X

Lab-8.3.py > ...

```
1 # Task 1: Email Validation using TDD
2
3 import re
4 def is_valid_email(email):
5     # Check if email contains exactly one @ symbol
6     if email.count('@') != 1:
7         return False
8
9     # Check if (function) startswith: Any special characters
10    if email.startswith(('@', '.')) or email.endswith(('@', '.')):
11        return False
12
13    # Check if email contains at least one . after the @ symbol
14    local_part, domain_part = email.split('@')
15    if '.' not in domain_part:
16        return False
17
18    # Check for valid characters in local and domain parts
19    if not re.match(r'^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$', email):
20        return False
21
22    return True
23
24 # Test cases
25 test_cases = [
26     ("valid@example.com", True),
27     ("invalid_email", False)
]
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
```

PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-8.3.py"

Testing: valid@example.com | Expected: True | Result: True | Pass: True

Testing: invalid_email | Expected: False | Result: False | Pass: True

Testing: @example.com | Expected: False | Result: False | Pass: True

Testing: user@.com | Expected: False | Result: False | Pass: True

Testing: user@example.com | Expected: False | Result: False | Pass: True

Testing: user@example..com | Expected: False | Result: True | Pass: False

Testing: .user@example.com | Expected: False | Result: False | Pass: True

Testing: user@example..com | Expected: False | Result: False | Pass: True

PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING>

Task 2: Grade Assignment using Loops.

The screenshot shows a code editor window titled "Lab-8.3.py". The code implements a function to assign grades based on scores. It includes test cases for various score values. The code is written in Python and uses a series of if-elif statements to determine the grade. The output pane shows the results of running the code with different test cases, including boundary values like 90, 80, 70, 60, and 50, as well as invalid inputs like -5 and 105.

```
41 # Task 2: Grade Assignment using Loops.
42
43 def assign_grade(score):
44     # Handle invalid inputs
45     if not isinstance(score, (int, float)):
46         return "Invalid input"
47     if score < 0 or score > 100:
48         return "Invalid input"
49
50     # Assign grade based on score ranges
51     if score >= 90:
52         return "A"
53     elif score >= 80:
54         return "B"
55     elif score >= 70:
56         return "C"
57     elif score >= 60:
58         return "D"
59     else:
60         return "F"
61
62 # Test cases
63 test_cases = [
64     # Valid cases - boundary values
65     (90, "A"),

```

PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-8.3.py"

- Testing: 90 | Expected: A | Result: A | Pass: True
- Testing: 80 | Expected: B | Result: B | Pass: True
- Testing: 70 | Expected: C | Result: C | Pass: True
- Testing: 60 | Expected: D | Result: D | Pass: True
- Testing: 59 | Expected: F | Result: F | Pass: True
- Testing: 95 | Expected: A | Result: A | Pass: True
- Testing: 85 | Expected: B | Result: B | Pass: True
- Testing: 75 | Expected: C | Result: C | Pass: True
- Testing: 65 | Expected: D | Result: D | Pass: True
- Testing: 50 | Expected: F | Result: F | Pass: True
- Testing: -5 | Expected: Invalid input | Result: Invalid input | Pass: True
- Testing: 105 | Expected: Invalid input | Result: Invalid input | Pass: True

Ln 41, Col 40 Spaces: 4 UTF-8 CRLF {} Python 8 3.14.2 Go Live

Task 3: Sentence Palindrome Checker.

The screenshot shows a code editor window with the following Python script:

```
Lab-8.3.py U
Lab-8.3.py > is_sentence_palindrome

92 # Task 3: Sentence Palindrome Checker
93 import re
94 def is_sentence_palindrome(sentence):
95     # Remove non-alphanumeric characters and convert to lowercase
96     cleaned_sentence = re.sub(r'[^A-Za-z0-9]', '', sentence).lower()
97
98     # Check if cleaned sentence is equal to its reverse
99     return cleaned_sentence == cleaned_sentence[::-1]
100
101 # Test cases
102 test_cases = [
103     ("A man a plan a canal Panama", True),
104     ("No 'x' in Nixon", True),
105     ("Was it a car or a cat I saw?", True),
106     ("Not a palindrome", False),
107     ("12321", True),
108     ("12345", False),
109     ("", True),
110     ("!@#$%^&*()_+", True),
111     ("Madam In Eden, I'm Adam", True),
112     ("Hello, World!", False)
113 ]
114
115 print("=" * 80)
116 print(f"{'Test Case':<40} {'Expected':<10} {'Result':<10} {'Pass':<10}")

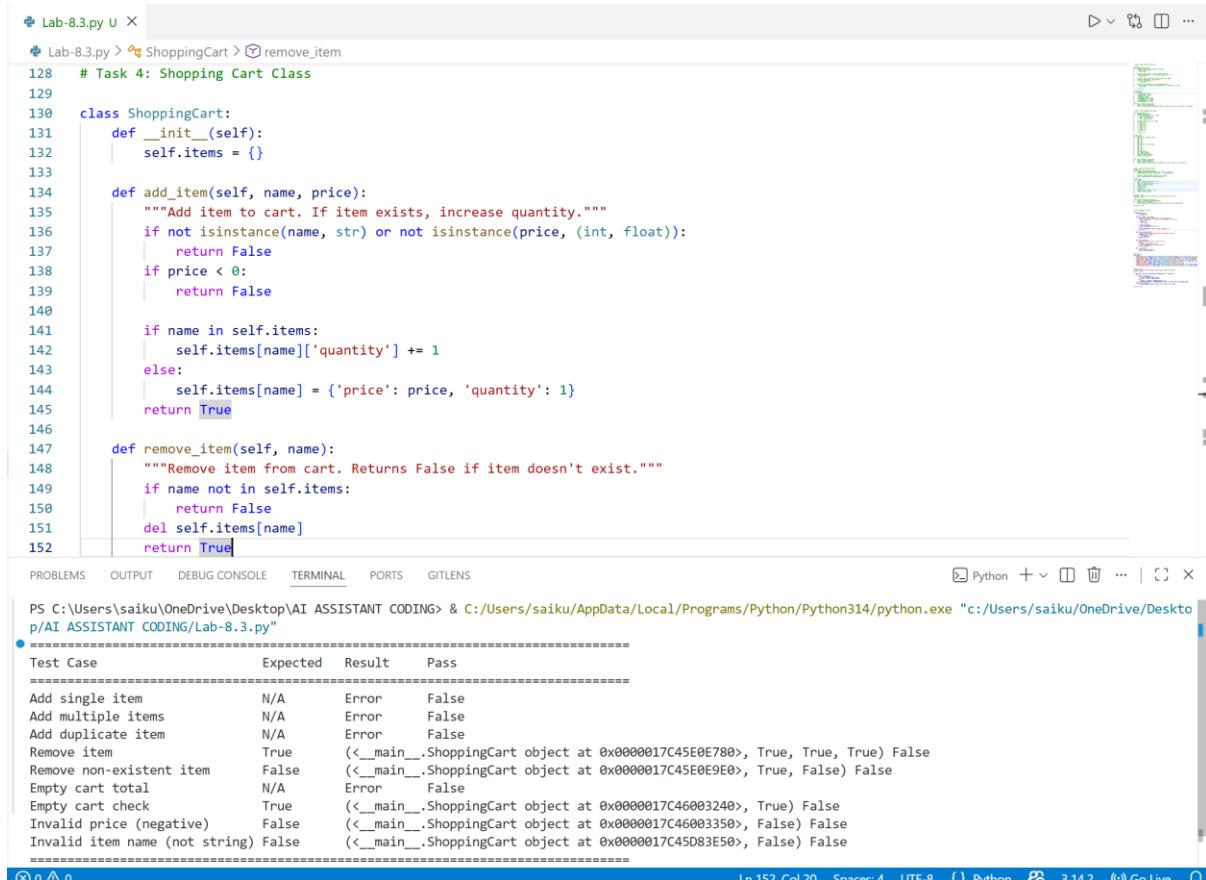
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
```

The terminal output shows the results of running the script with the provided test cases:

```
PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING & C:/Users/saiku/AppData/Local/Programs/Python/Python314/p
rive/Desktop/AI ASSISTANT CODING/Lab-8.3.py"
A man a plan a canal Panama      True   True   True
No 'x' in Nixon                True   True   True
Was it a car or a cat I saw?    True   True   True
Not a palindrome                 False  False  True
12321                           True   True   True
12345                           False  False  True
!@#$%^&*()_+                   True   True   True
Madam In Eden, I'm Adam        True   True   True
Hello, World!                   False  False  True
=====
PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING>
```

Bottom status bar: Ln 96, Col 69 | Spaces: 4 | UTF-8 | CRLF | {} Python | 3.14.2 | Go Live |

Task 4: ShoppingCart Class.



```
Lab-8.3.py X
Lab-8.3.py > ShoppingCart > remove_item

128 # Task 4: Shopping Cart Class
129
130 class ShoppingCart:
131     def __init__(self):
132         self.items = {}
133
134     def add_item(self, name, price):
135         """Add item to cart. If item exists, increase quantity."""
136         if not isinstance(name, str) or not isinstance(price, (int, float)):
137             return False
138         if price < 0:
139             return False
140
141         if name in self.items:
142             self.items[name]['quantity'] += 1
143         else:
144             self.items[name] = {'price': price, 'quantity': 1}
145
146         return True
147
148     def remove_item(self, name):
149         """Remove item from cart. Returns False if item doesn't exist."""
150         if name not in self.items:
151             return False
152         del self.items[name]
153
154         return True
```

Test Case	Expected	Result	Pass
Add single item	N/A	Error	False
Add multiple items	N/A	Error	False
Add duplicate item	N/A	Error	False
Remove item	True	(<__main__.ShoppingCart object at 0x0000017C45E0E780>, True, True, True)	False
Remove non-existent item	False	(<__main__.ShoppingCart object at 0x0000017C45E0E9E0>, True, False)	False
Empty cart total	N/A	Error	False
Empty cart check	True	(<__main__.ShoppingCart object at 0x0000017C46003240>, True)	False
Invalid price (negative)	False	(<__main__.ShoppingCart object at 0x0000017C46003350>, False)	False
Invalid item name (not string)	False	(<__main__.ShoppingCart object at 0x0000017C45D83E50>, False)	False

Task 5: Date Format Conversion.

```

Lab-8.3.py X
Lab-8.3.py > ...

200 # Task 5: Date Format Conversion.
201
202 def convert_date_format(date_str):
203     """Convert date from YYYY-MM-DD to DD-MM-YYYY format."""
204     try:
205         # Validate input is a string
206         if not isinstance(date_str, str):
207             return "Invalid input"
208
209         # Validate format
210         if len(date_str) != 10 or date_str.count('-') != 2:
211             return "Invalid format"
212
213         # Split the date
214         parts = date_str.split('-')
215         if len(parts) != 3:
216             return "Invalid format"
217
218         year, month, day = parts
219
220         # Validate each part
221         if not (year.isdigit() and month.isdigit() and day.isdigit()):
222             return "Invalid format"
223
224         year_int = int(year)
225         month_int = int(month)
226         day_int = int(day)
227
228         # Validate ranges
229         if month_int < 1 or month_int > 12:
230             return "Invalid input"
231         if day_int < 1 or day_int > 31:
232             return "Invalid input"
233
234         # Return converted format
235         return f"{day}-{month}-{year}"
236
237     except Exception:
238         return "Invalid input"
239

```

Ln 200, Col 34 Spaces: 4 UTF-8 CRLF {} Python 3.14.2 Go Live

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS	GITLENS	
PS C:\Users\saiku\OneDrive\Desktop\AI ASSISTANT CODING> & C:/Users/saiku/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/saiku/OneDrive/Desktop/AI ASSISTANT CODING/Lab-8.3.py"						Python + ▾
====						...
Test Input	Expected	Result	Pass			✖
====						
2023-10-15	15-10-2023	15-10-2023	True			
2024-01-01	01-01-2024	01-01-2024	True			
2023-12-31	31-12-2023	31-12-2023	True			
2023-02-28	28-02-2023	28-02-2023	True			
2000-05-15	15-05-2000	15-05-2000	True			
1999-12-25	25-12-1999	25-12-1999	True			
2023-13-01	Invalid input	Invalid input	True			
2023-00-15	Invalid input	Invalid input	True			
2023-10-32	Invalid input	Invalid input	True			
2023-10-0	Invalid input	Invalid format	False			
invalid-date	Invalid format	Invalid format	True			
2023/10/15	Invalid format	Invalid format	True			
10-15-2023	Invalid format	Invalid input	False			
	Invalid format	Invalid format	True			
123	Invalid input	Invalid input	True			
2023-13-01	Invalid input	Invalid input	True			
2023-00-15	Invalid input	Invalid input	True			
2023-10-32	Invalid input	Invalid input	True			
2023-10-0	Invalid input	Invalid format	False			
invalid-date	Invalid format	Invalid format	True			
2023/10/15	Invalid format	Invalid format	True			
10-15-2023	Invalid format	Invalid input	False			
	Invalid format	Invalid format	True			
123	Invalid input	Invalid input	True			
2023-00-15	Invalid input	Invalid input	True			
2023-10-32	Invalid input	Invalid input	True			
2023-10-0	Invalid input	Invalid format	False			
invalid-date	Invalid format	Invalid format	True			
2023/10/15	Invalid format	Invalid format	True			
10-15-2023	Invalid format	Invalid input	False			
	Invalid format	Invalid format	True			
123	Invalid input	Invalid input	True			
2023-10-32	Invalid input	Invalid input	True			
2023-10-0	Invalid input	Invalid format	False			
invalid-date	Invalid format	Invalid format	True			
2023/10/15	Invalid format	Invalid format	True			
10-15-2023	Invalid format	Invalid input	False			
	Invalid format	Invalid format	True			
123	Invalid input	Invalid input	True			
2023-10-0	Invalid input	Invalid format	False			
invalid-date	Invalid format	Invalid format	True			
2023/10/15	Invalid format	Invalid format	True			
10-15-2023	Invalid format	Invalid input	False			
	Invalid format	Invalid format	True			

Ln 200, Col 34 Spaces: 4 UTF-8 CRLF {} Python 3.14.2 Go Live