

## Assignment -8.3

2303a51343

Batch-10

### TASK-1

**Prompt:** write a python program to develop a user registration system that requires reliable email input validation

**Code:** import re

```
class UserRegistration:
```

```
    def __init__(self, name, email):
```

```
        self.name = name
```

```
        self.email = email
```

```
    def validate_email(self):
```

```
        pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
```

```
        if re.match(pattern, self.email):
```

```
            return True
```

```
        else:
```

```
            return False
```

```
    def display(self):
```

```
        print(f"Name: {self.name}")
```

```
        print(f"Email: {self.email}")
```

```
        if self.validate_email():
```

```
            print("Email is valid.")
```

```
        else:
```

```
            print("Email is invalid.")
```

```
# Test Cases
```

```
user1 = UserRegistration("Rashmitha", "rashu@example.com")
```

```
user2 = UserRegistration("nithya", "nithya@example")
```

```
user1.display()
```

```
user2.display()
```

**Output:**

```

File Edit Selection View Go ... < > AI ASSISTED
EXPLORER SEARCH RECENT DOCUMENTS > assignments > vs-code > 8.3.py ...
8.3.py > 8.3.py ...
1 #!/usr/bin/python
2 # Write a python program to develop a user registration system that requires reliable email input validation
3
4 import re
5
6 class UserRegistration:
7     def __init__(self, name, email):
8         self.name = name
9         self.email = email
10
11     def validate_email(self):
12         pattern = "[a-zA-Z0-9]+@[a-zA-Z0-9]+\.[a-zA-Z]{2,}?"
13         if re.match(pattern, self.email):
14             return True
15         else:
16             return False
17
18     def display(self):
19         print("Name: ", self.name)
20         print("Email: ", self.email)
21
22 if __name__ == "__main__":
23     user1 = UserRegistration("nithya", "nithya@example.com")
24     user2 = UserRegistration("nithya", "nithya@example")
25
26 user2.display()
27
28 #QUESTION 2:Write a python program to build an automated grading system for an online examination platform using loops and also include invalid input like -5,eighty

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLens Python Debug 2025-10-26 09:16:08 win32 Go Live FnLock

```

PS C:\Users\Nithya\OneDrive\Desktop\VS ASSISTED> & C:\Users\Nithya\OneDrive\Desktop\VS ASSISTED\8.3.py
Name: nithya
Email: nithya@example
Email is Invalid.
PS C:\Users\Nithya\OneDrive\Desktop\VS ASSISTED>

```

**Analysis:** This program creates a User Registration class to store a user's name and email address. It checks whether the email is written in a correct format using a pattern. The program uses the re module to compare the email with this pattern. If the email follows the correct format, it prints that the email is valid; otherwise, it prints that the email is invalid.

## TASK-2

**Prompt :** write a python program to bulid an automated grading system for an online examination platform using loops and aslo include invalid input like -5,eighty

**Code:** def calculate\_grade(score):

```

if score < 0 or score > 100:
    return "Invalid input"
elif score >= 90:
    return "Grade: A"
elif score >= 80:
    return "Grade: B"
elif score >= 70:
    return "Grade: C"
elif score >= 60:
    return "Grade: D"
else:
    return "Grade: F"

scores = [95, 85, 75, 65, 55, -5, "eighty"]

for s in scores:
    if isinstance(s, int):
        print(f"Score: {s} - {calculate_grade(s)}")
    else:
        print(f"Score: {s} - Invalid input")

```

## Output:

The screenshot shows the VS Code interface with a Python file named 8.3.py open. The code defines a function calculate\_grade(score) that returns a grade based on a score. It includes a loop to test the function with various inputs. The terminal below shows the execution of the script and its output, which includes some syntax warnings and the results of the test cases.

```
vs-code > 8.3.py > calculate_grade
27
28 #ASK-2:write a python program to build an automated grading system for an online examination platform
29
30 def calculate_grade(score):
31     if score < 0 or score > 100:
32         return "Invalid input"
33     elif score > 90:
34         return "Grade: A"
35     elif score >= 80:
36         return "Grade: B"
37     elif score >= 60:
38         return "Grade: C"
39     elif score >= 40:
40         return "Grade: D"
41     else:
42         return "Grade: F"
43
44 # Test Cases
45 scores = [95, 85, 75, 65, 55, -5,
46 "eighty"]
47 for s in scores:
48     if isinstance(s, int):
49         print(f"Score: {s} - {calculate_grade(s)}")
50     else:
51         print(f"Score: {s} - Invalid input")
52
53 """#TASK-3:write a python program to check sentence palindrome checker to develop a text-processing utility
54 def is_palindrome(sentence):
```

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS	GIT LENS
0	0	0	0	0	0

```
python3.13.exe' c:\Users\rashmita\vscode\extensions\ms-python.debugpy-2025.19.2026021601-win32-x64\bundled\111
e:\OneDrive\Desktop\AI ASSISTED\vs-code\8.3.py:9: SyntaxWarning: invalid escape sequence '\.'
  patterns = r'[^a-zA-Z0-9_]+@[a-zA-Z0-9_-]+\.[a-zA-Z]{2,3}$'
score: 95 - Grade: A
score: 85 - Grade: B
score: 75 - Grade: C
Score: 65 - Grade: D
Score: 55 - Grade: F
Score: -5 - Invalid input
Score: eighty - Invalid input
PS c:\Users\rashmita\OneDrive\Desktop\AI ASSISTED>
```

**Analysis:** This program creates an automated grading system using a function and a loop. It assigns grades based on score ranges and checks for invalid values like negative numbers or non-numeric inputs. The loop processes each score and prints the result. Invalid inputs such as -5 and "eighty" are handled safely without errors.

## TASK-3

**Prompt :** write a python program to check sentence palindrome checker to develop a text-processing utility to analyze sentences

### Code:

```
def is_palindrome(sentence):

    cleaned = ".join(sentence.lower().split())

    return cleaned == cleaned[::-1]

# Test Cases

sentences = ["Madam In Eden Im Adam",

"Hello World","A man a plan a canal Panama"]

for s in sentences:

    if is_palindrome(s):

        print(f"{s} is a palindrome.")

    else:

        print(f"{s} is not a palindrome.")
```

## **Output:**

**Analysis:** This program checks if a sentence is a palindrome. It changes the sentence to lowercase and removes spaces, then compares it with its reverse. If both are the same, it prints that the sentence is a palindrome; otherwise, it prints that it is not.

## **TASK-4:**

**Prompt:** Write a python program to design a basic shopping cart module for an e-commerce application add item and remove item and total cost

**Code:** class ShoppingCart:

```
def __init__(self):
    self.cart = {}

def add_item(self, item, price):
    self.cart[item] = price
    print(f"Added {item} to cart at ₹{price}")

def remove_item(self, item):
    if item in self.cart:
        del self.cart[item]
        print(f"Removed {item} from cart")
    else:
        print(f"{item} not found in cart")

def total_cost(self):
    return sum(self.cart.values())

# Test Cases

cart = ShoppingCart()
```

```

cart.add_item("Laptop", 50000)
cart.add_item("Headphones", 2000)
cart.add_item("Mouse", 500)
print("Total Cost: ₹", cart.total_cost())
cart.remove_item("Headphones")
print("Total Cost after removal: ₹", cart.total_cost())

```

### Output:

```

File Edit Selection View Go ... ← → 🔍 AI ASSISTED
EXPLORER SEARCH vs-code > 8.3.py > ShoppingCart
> AI ASSISTED
> assignments
> vscode
# writing a program...
8.3.py
assignment3.3.py
assignment3.4.py
assignment4.py
assignment6.3.py
assignment6.4.py
labexam1.py
assignment1.py
retailshop.py
salesanalysis.py
J Sum.java
2303a51343-3.3.pdf
ai-labexam1.py
assignments-4.py
labexam1.py
J Sum.java
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
Python Debug Console
PS C:\Users\Rashmita\OneDrive\Desktop\AI ASSISTED> c:\cd "c:\Users\Rashmita\OneDrive\Desktop\AI ASSISTED"; & "c:\Users\Rashmita\AppData\Local\Microsoft\Visual Studio\2022\Community\VS Code\extensions\ms-python.python\2025.20260216.0\vscode\bin\bundled\libs\debug\launcher" -d="c:\Users\Rashmita\OneDrive\Desktop\AI ASSISTED\code\8.3.py"
c:\Users\Rashmita\OneDrive\Desktop\AI ASSISTED\code\8.3.py:9: SyntaxWarning: invalid escape sequence '\.'
    pattern = r'([a-zA-Z0-9_]+@[a-zA-Z0-9_]+\.[a-zA-Z]{2,})$'
pattern = r'([a-zA-Z0-9_]+@[a-zA-Z0-9_]+\.[a-zA-Z]{2,})$'
Added Laptop to cart at ₹50000
Added Headphones to cart at ₹2000
Added Mouse to cart at ₹500
Total Cost: ₹50500
Removed Headphones from cart
Total Cost after removal: ₹ 50500
PS C:\Users\Rashmita\OneDrive\Desktop\AI ASSISTED>

```

**Analysis:** This program creates a ShoppingCart class to add and remove items with their prices. It keeps track of items in a dictionary and calculates the total cost. The test shows adding items, removing one, and updating the total cost correctly.

### TASK-5

**Prompt:** write a python program to create a utility function to convert date formats for reports

```
from datetime import datetime
```

**Code:** from datetime import datetime

```
def convert_date_format(date_str, current_format, desired_format):
```

```
try:
```

```
    date_obj = datetime.strptime(date_str, current_format)
```

```
    return date_obj.strftime(desired_format)
```

```
except ValueError:
```

```
    return "Invalid date format"
```

```
# Test Cases
```

```
dates = [("2024-06-15", "%Y-%m-%d", "%d/%m/%Y"),
```

```
("15/06/2024", "%d/%m/%Y", "%Y-%m-%d"),
```

```
("06-15-2024", "%m-%d-%Y", "%Y/%m/%d"),
("invalid-date", "%Y-%m-%d", "%d/%m/%Y")]
```

for date\_str, current\_fmt, desired\_fmt in dates:

```
    print(f"Original: {date_str} - Converted: {convert_date_format(date_str, current_fmt, desired_fmt)}")
```

### Output:

The screenshot shows the Visual Studio Code interface. The code editor displays a Python script named '8.3.py' with the following content:

```

#TASK-5:write a python program to create a utility function to convert date formats for reports
from datetime import datetime
def convert_date_format(date_str, current_format, desired_format):
    try:
        date_obj = datetime.strptime(date_str, current_format)
        return date_obj.strftime(desired_format)
    except ValueError:
        return "Invalid date format"

# Test Cases
dates = [("2024-06-15", "%Y-%m-%d", "%d/%m/%Y"),
         ("15/06/2024", "%d/%m/%Y", "%Y-%m-%d"),
         ("06-15-2024", "%m-%d-%Y", "%Y/%m/%d"),
         ("invalid-date", "%Y-%m-%d", "%d/%m/%Y")]
for date_str, current_fmt, desired_fmt in dates:
    print(f"Original: {date_str} - Converted: {convert_date_format(date_str, current_fmt, desired_fmt)}")

```

The terminal tab shows the execution of the script and its output:

```

Original: 15/06/2024 - Converted: 2024-06-15
Original: 06-15-2024 - Converted: 2024-06-15
Original: invalid-date - Converted: Invalid date format

```

**Analysis:** This program defines a function to convert dates from one format to another using Python's datetime module. It tries to parse the input date string with the current format and then outputs it in the desired format. If the input date doesn't match the format, it returns "Invalid date format." The test cases show successful conversions and correctly handle invalid inputs like "invalid-date".