

AI-ASSISTED CODING

Assignment Number:8.3

Name – sai akshith

BATCH – 45

Task 1: Email Validation using TDD

PROMT:

valid email address You are developing a user registration system that requires reliable email input validation.it must cointain "@" and "." and should not contain any spaces and not allow multiple"@and should cointain only small letters and should cointain gmail.com or yahoo.com or outlook.com

CODE:

```
email = input("Enter your email address: ")
```

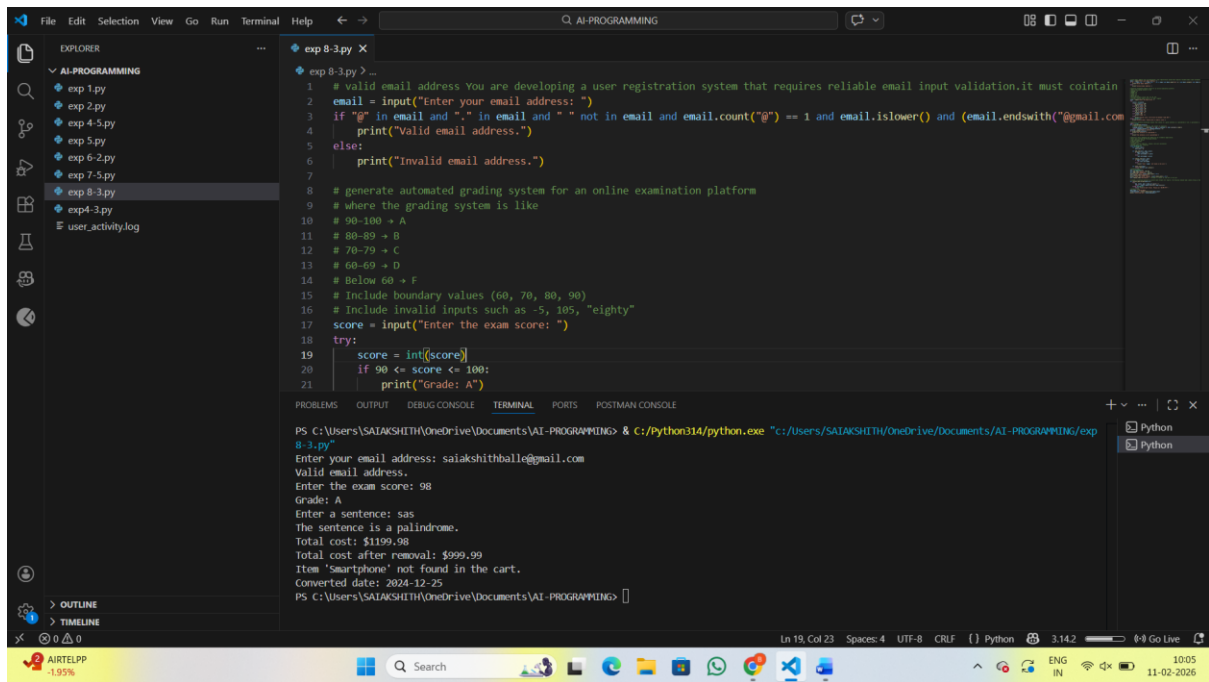
```
if "@" in email and "." in email and " " not in email and email.count("@") == 1 and  
email.islower() and (email.endswith("@gmail.com") or email.endswith("@yahoo.com") or  
email.endswith("@outlook.com")):
```

```
    print("Valid email address.")
```

```
else:
```

```
    print("Invalid email address.")
```

OUTPUT:



Task 2: Grade Assignment using Loops

PROMT:

generate automated grading system for an online examination platform

where the grading system is like

90–100 → A

80–89 → B

70–79 → C

60–69 → D

Below 60 → F

Include boundary values (60, 70, 80, 90)

Include invalid inputs such as -5, 105, "eighty"

CODE:

score = input("Enter the exam score: ")

try:

```

score = int(score)

if 90 <= score <= 100:

    print("Grade: A")

elif 80 <= score < 90:

    print("Grade: B")

elif 70 <= score < 80:

    print("Grade: C")

elif 60 <= score < 70:

    print("Grade: D")

elif score < 60:

    print("Grade: F")

else:

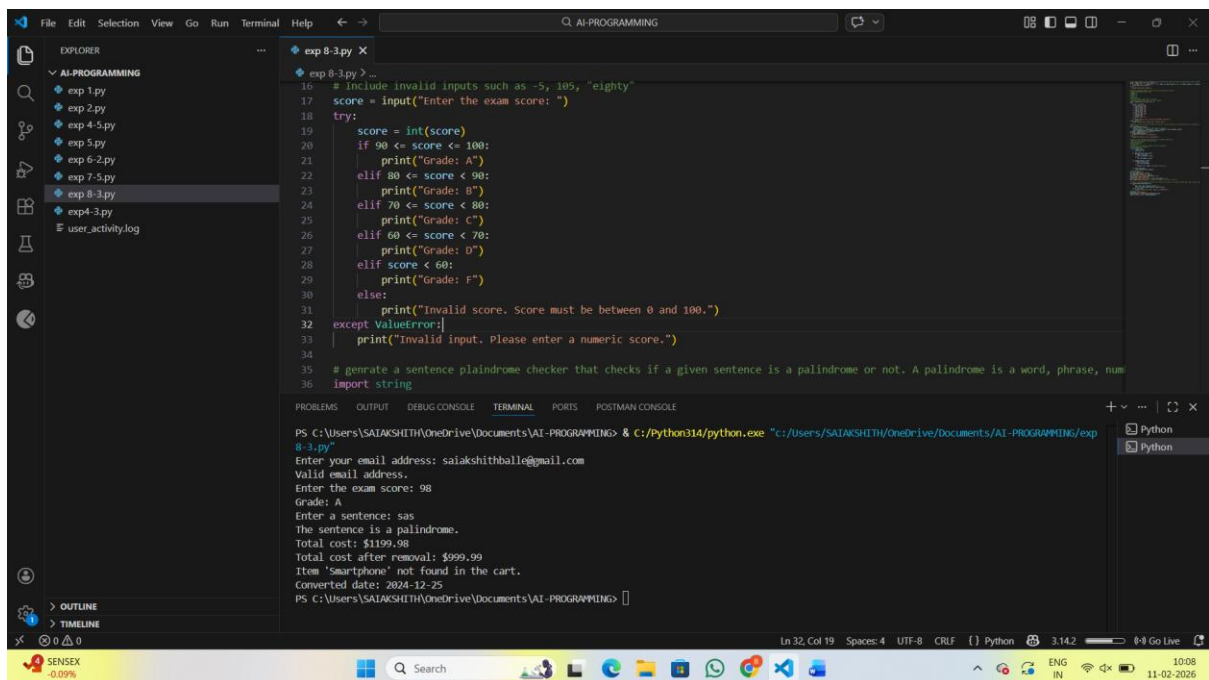
    print("Invalid score. Score must be between 0 and 100.")

except ValueError:

    print("Invalid input. Please enter a numeric score.")

```

OUTPUT:



The screenshot shows a Visual Studio Code editor window with a Python file named `exp 8-3.py` open. The code in the file is a grade calculator that takes a score as input and prints the corresponding grade (A, B, C, D, F) or an error message if the score is invalid or the input is not a numeric value. The code uses a try-except block to handle `ValueError`.

The terminal output shows the execution of the script. It prompts the user to enter an email address, which is `saiaakshithballe@gmail.com`. Then it prompts for an exam score, which is `98`. The output shows the grade `A`. After that, it prompts for a sentence, which is `sas`. The output shows that the sentence is not a palindrome. Finally, it prompts for a total cost, which is `$1199.98`. The output shows the total cost after removal of a smartphone, which is `$999.99`. The terminal also shows the current date and time: `2024-12-25`.

```

PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "C:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp
8-3.py"
Enter your email address: saiaakshithballe@gmail.com
Valid email address.
Enter the exam score: 98
Grade: A
Enter a sentence: sas
The sentence is a palindrome.
Total cost: $1199.98
Total cost after removal: $999.99
Item 'Smartphone' not found in the cart.
Converted date: 2024-12-25
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>

```

Task 3: Sentence Palindrome Checker

PROMT:

genrate a sentence plaindrome checker that checks if a given sentence is a palindrome or not. A palindrome is a word, phrase, number, or other sequence of characters that reads the same forward and backward (ignoring spaces, punctuation, and capitalization).

CODE:

```
import string

def is_palindrome(sentence):

    cleaned_sentence = ''.join(char for char in sentence if char.isalnum()).lower()

    return cleaned_sentence == cleaned_sentence[::-1]

sentence = input("Enter a sentence: ")

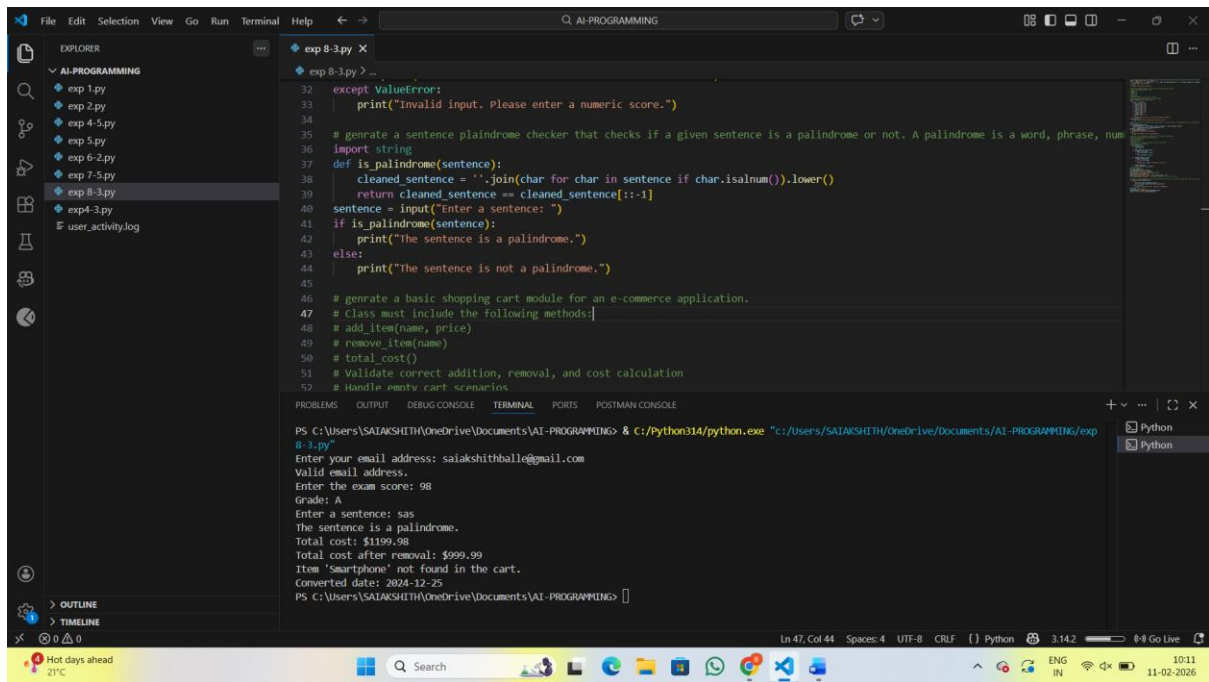
if is_palindrome(sentence):

    print("The sentence is a palindrome.")

else:

    print("The sentence is not a palindrome.")
```

OUTPUT:



Task 4: ShoppingCart Class

PROMT:

generate a basic shopping cart module for an e-commerce application.

Class must include the following methods:

`add_item(name, price)`

`remove_item(name)`

`total_cost()`

Validate correct addition, removal, and cost calculation

Handle empty cart scenarios

CODE:

class ShoppingCart:

def `__init__`(self):

self.cart = {}

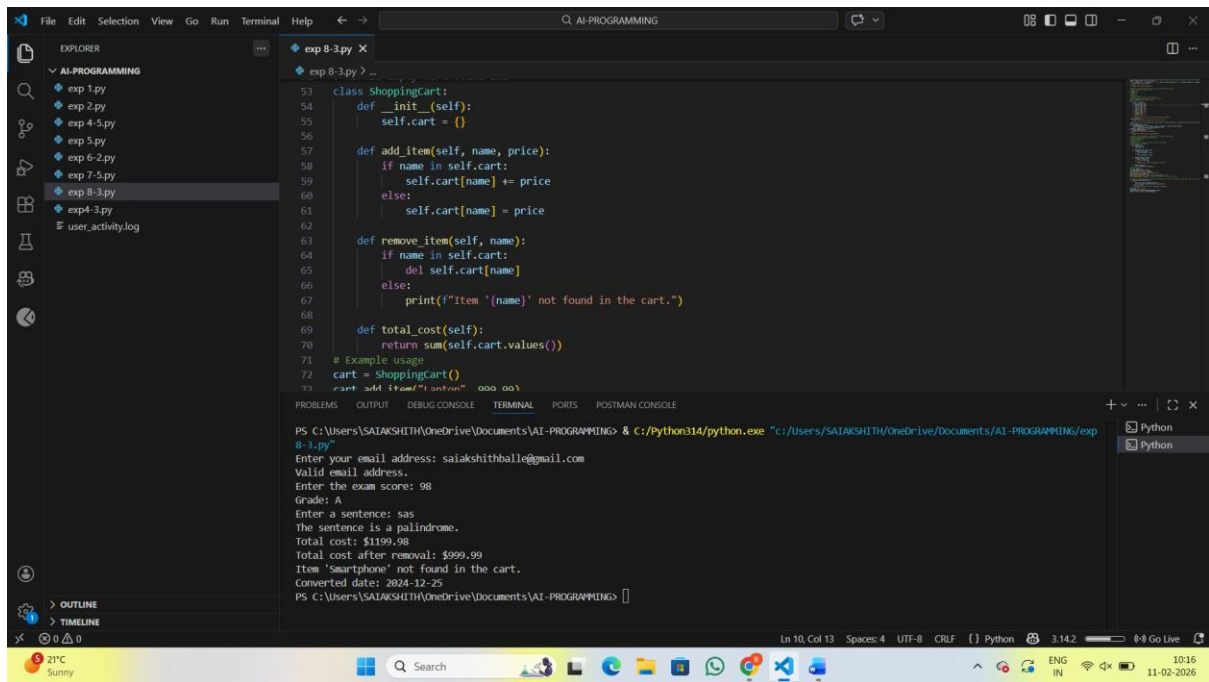
```
def add_item(self, name, price):
    if name in self.cart:
        self.cart[name] += price
    else:
        self.cart[name] = price

def remove_item(self, name):
    if name in self.cart:
        del self.cart[name]
    else:
        print(f"Item '{name}' not found in the cart.")

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

# Example usage
cart = ShoppingCart()
cart.add_item("Laptop", 999.99)
cart.add_item("Headphones", 199.99)
print(f"Total cost: ${cart.total_cost():.2f}")
cart.remove_item("Headphones")
print(f"Total cost after removal: ${cart.total_cost():.2f}")
cart.remove_item("Smartphone")
```

OUTPUT:



Task 5: Date Format Conversion

PROMT:

create a a utility function to convert date formats for reports. The function should take a date string in the format "DD/MM/YYYY" and convert it to "YYYY-MM-DD".

CODE:

```
def convert_date_format(date_str):

    try:

        day, month, year = date_str.split('/')

        return f'{year}-{month.zfill(2)}-{day.zfill(2)}'

    except ValueError:

        return "Invalid date format. Please use 'DD/MM/YYYY'."

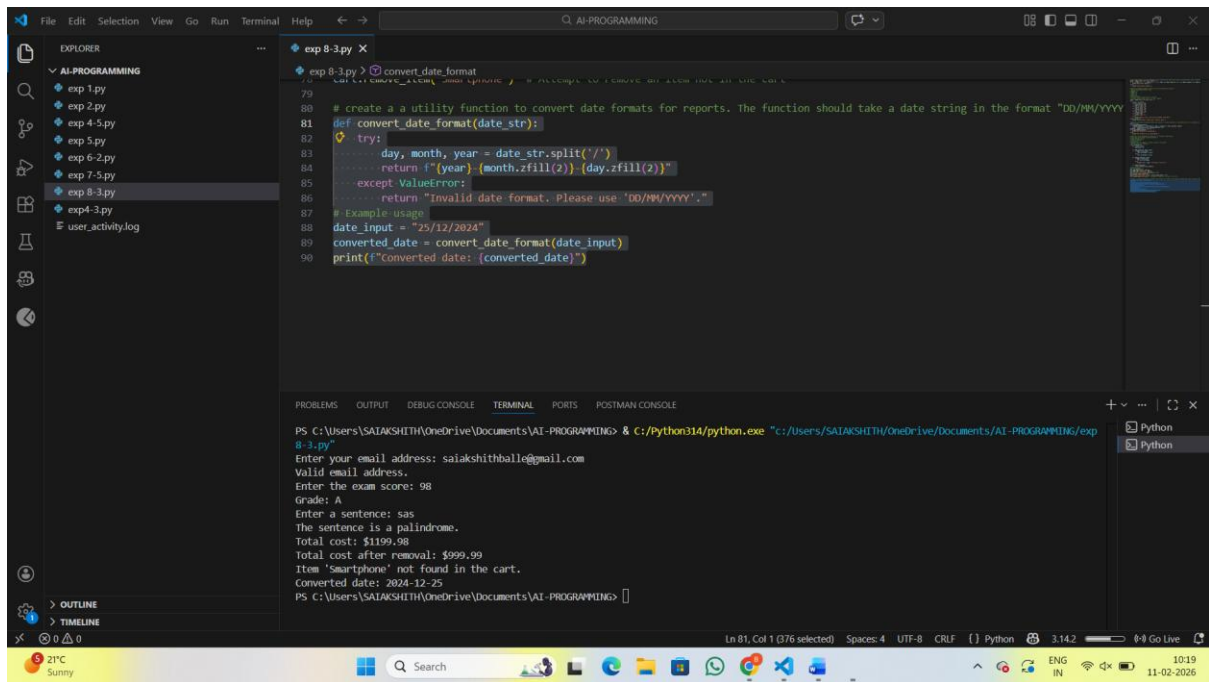
# Example usage

date_input = "25/12/2024"

converted_date = convert_date_format(date_input)

print(f"Converted date: {converted_date}")
```

OUTPUT:



The screenshot displays the Visual Studio Code (VS Code) interface. The Explorer panel on the left shows a project named 'AI-PROGRAMMING' with several files, including 'exp 8-3.py'. The main editor window is open to 'exp 8-3.py', which contains a Python function `convert_date_format` designed to convert date strings from 'DD/MM/YYYY' to 'YYYY-MM-DD'. The function uses `date.strptime` for parsing and `date.strftime` for formatting. Below the function, an example usage is provided, converting the date '25/12/2024' to '2024-12-25'.

```
79
80
81 # create a utility function to convert date formats for reports. The function should take a date string in the format "DD/MM/YYYY"
82 def convert_date_format(date_str):
83     try:
84         day, month, year = date_str.split('/')
85         return f"{year}-{month.zfill(2)}-{day.zfill(2)}"
86     except ValueError:
87         return "Invalid date format. Please use 'DD/MM/YYYY'."
88 # Example usage
89 date_input = "25/12/2024"
90 converted_date = convert_date_format(date_input)
91 print(f"Converted date: {converted_date}")
```

The Terminal panel at the bottom shows the command prompt output after running the script. It displays the execution of the `convert_date_format` function for various inputs, including email addresses, exam scores, grades, and dates, resulting in formatted outputs.

```
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING> & C:/Python314/python.exe "C:/Users/SAIAKSHITH/OneDrive/Documents/AI-PROGRAMMING/exp
8-3.py"
Enter your email address: saiakshithballe@gmail.com
Valid email address.
Enter the exam score: 98
Grade: A
Enter a sentence: sas
The sentence is a palindrome.
Total cost: $1199.98
Total cost after removal: $999.99
Item 'Smartphone' not found in the cart.
Converted date: 2024-12-25
PS C:\Users\SAIAKSHITH\OneDrive\Documents\AI-PROGRAMMING>
```

The status bar at the bottom indicates the current file is 'exp 8-3.py' at line 81, column 1, with 0 characters selected. The editor is configured for Python 3.14.2, using UTF-8 encoding and CRLF line endings.