TECHNIK NEST

INNOVATIVE MINDS, NESTING SUCCESS

Name: Maaz Sher Muhammad

Intern ID: TN/1N01/003

Email ID : maazshermuhammadofficial@gmail.com

Internship Domain : Python Internee

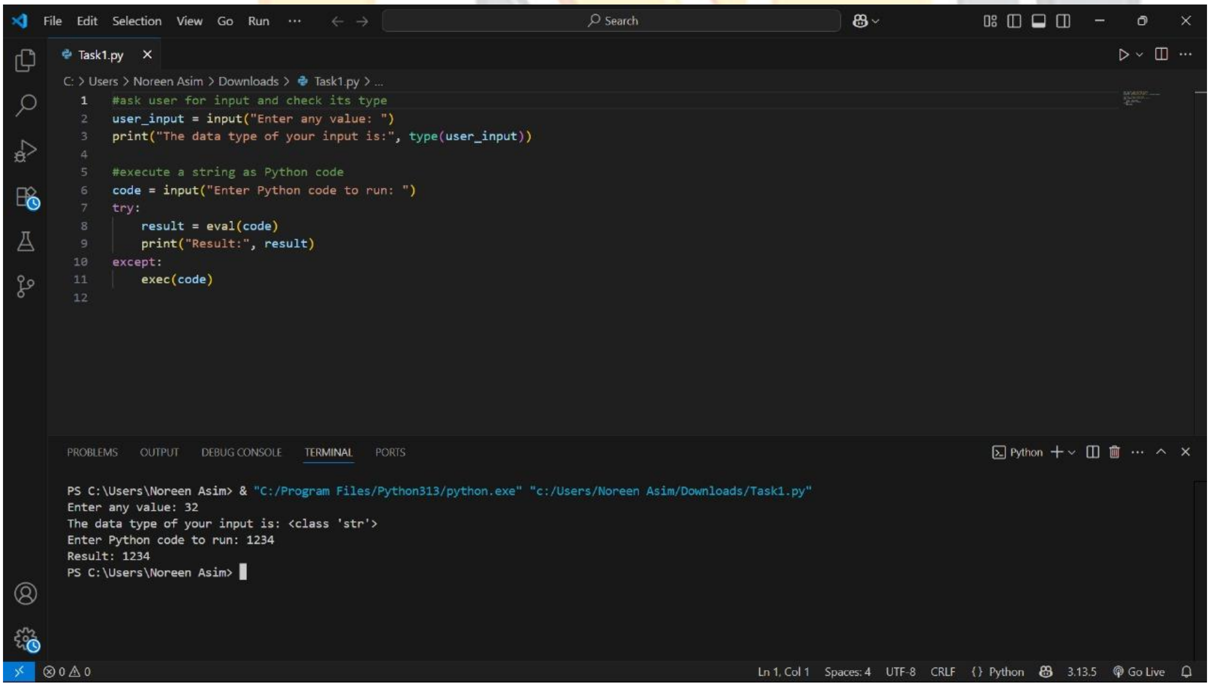
Task Week : 4

Instructor Name : Hassan Ali

Task 1 :

Code Snippet & Screenshot

```
#ask user for input and check its type

user_input = input("Enter any value: ")

print("The data type of your input is:", type(user_input))


#execute a string as Python code

code = input("Enter Python code to run: ")

try:

    result = eval(code)

    print("Result:", result)

except:

    exec(code)
```



```
#ask user for input and check its type
user_input = input("Enter any value: ")
print("The data type of your input is:", type(user_input))

#execute a string as Python code
code = input("Enter Python code to run: ")
try:
    result = eval(code)
    print("Result:", result)
except:
    exec(code)
```

```
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task1.py"
Enter any value: 32
The data type of your input is: <class 'str'>
Enter Python code to run: 1234
Result: 1234
PS C:\Users\Noreen Asim> █
```

Challenges:

□ Couldn’t differentiate between eval() and exec() clearly at first — led to wrong usage.

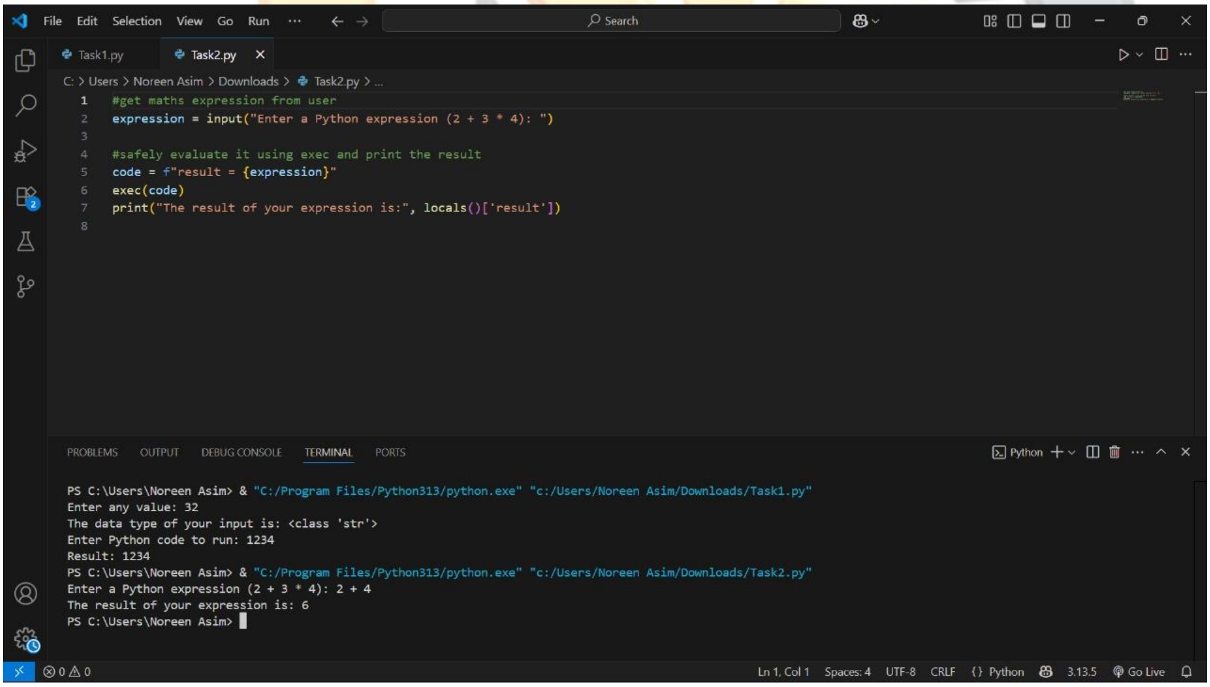
□ Struggled to display proper error messages when code execution failed.

□ Faced issue when users entered input with syntax errors — caused full program crash without traceback.

Task 2 :

Code Snippet & Screenshot

```
#get maths expression from user

expression = input("Enter a Python expression (2 + 3 * 4): ")


#safely evaluate it using exec and print the result

code = f"result = {expression}"

exec(code)

print("The result of your expression is:", locals()['result'])
```



```
#get maths expression from user
expression = input("Enter a Python expression (2 + 3 * 4): ")

#safely evaluate it using exec and print the result
code = f"result = {expression}"
exec(code)
print("The result of your expression is:", locals()['result'])
```

```
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task1.py"
Enter any value: 32
The data type of your input is: <class 'str'>
Enter Python code to run: 1234
Result: 1234
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task2.py"
Enter a Python expression (2 + 3 * 4): 2 + 4
The result of your expression is: 6
PS C:\Users\Noreen Asim> 
```

Challenges:

□ Didn’t initially validate the user’s expression for malicious code (e.g., os.system()).

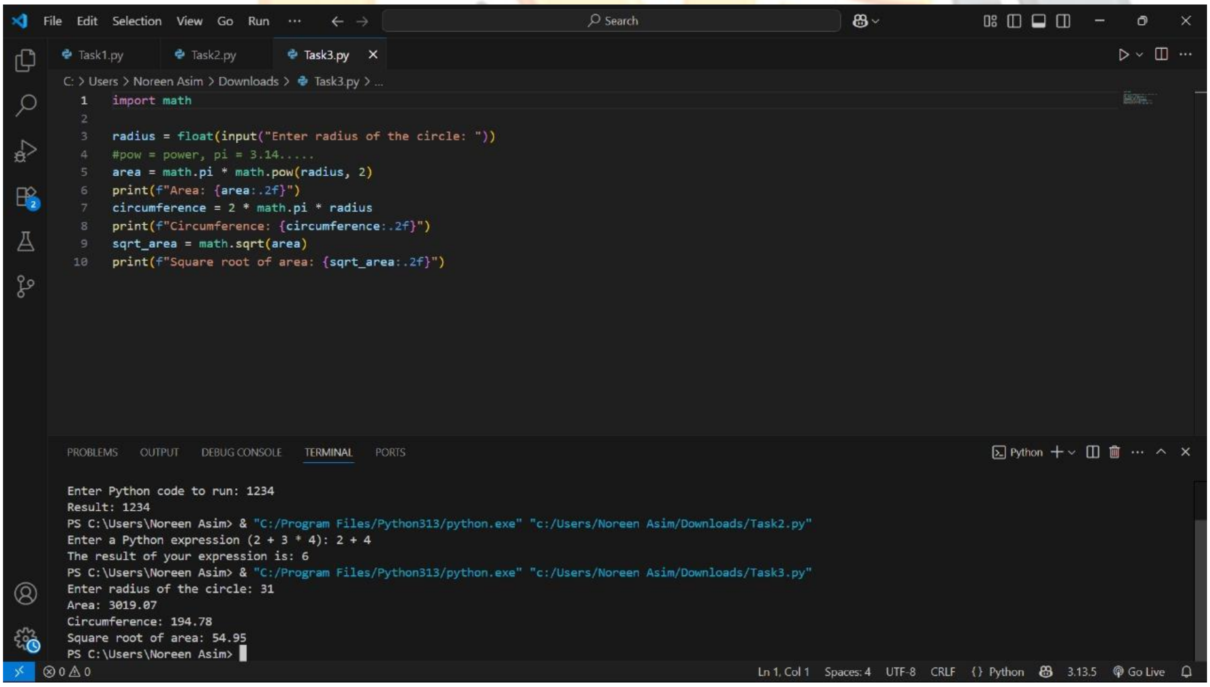
□ Difficulty understanding how locals() retrieves dynamically created variables.

□ Forgot to wrap exec() inside try-except for safer error handling.

Task 3 :

Code Snippet & Screenshot

```
import math


radius = float(input("Enter radius of the circle: "))

#pow = power, pi = 3.14.....

area = math.pi * math.pow(radius, 2)

print(f"Area: {area:.2f}")

circumference = 2 * math.pi * radius

print(f"Circumference: {circumference:.2f}")

sqrt_area = math.sqrt(area)

print(f"Square root of area: {sqrt_area:.2f}")
```



```
import math

radius = float(input("Enter radius of the circle: "))
#pow = power, pi = 3.14.....
area = math.pi * math.pow(radius, 2)
print(f"Area: {area:.2f}")
circumference = 2 * math.pi * radius
print(f"Circumference: {circumference:.2f}")
sqrt_area = math.sqrt(area)
print(f"Square root of area: {sqrt_area:.2f}")
```

```
Enter Python code to run: 1234
Result: 1234
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task2.py"
Enter a Python expression (2 + 3 * 4): 2 + 4
The result of your expression is: 6
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task3.py"
Enter radius of the circle: 31
Area: 3019.07
Circumference: 194.78
Square root of area: 54.95
PS C:\Users\Noreen Asim>
```

Challenges:

□ Initially used ** instead of math.pow() and got inconsistent results.

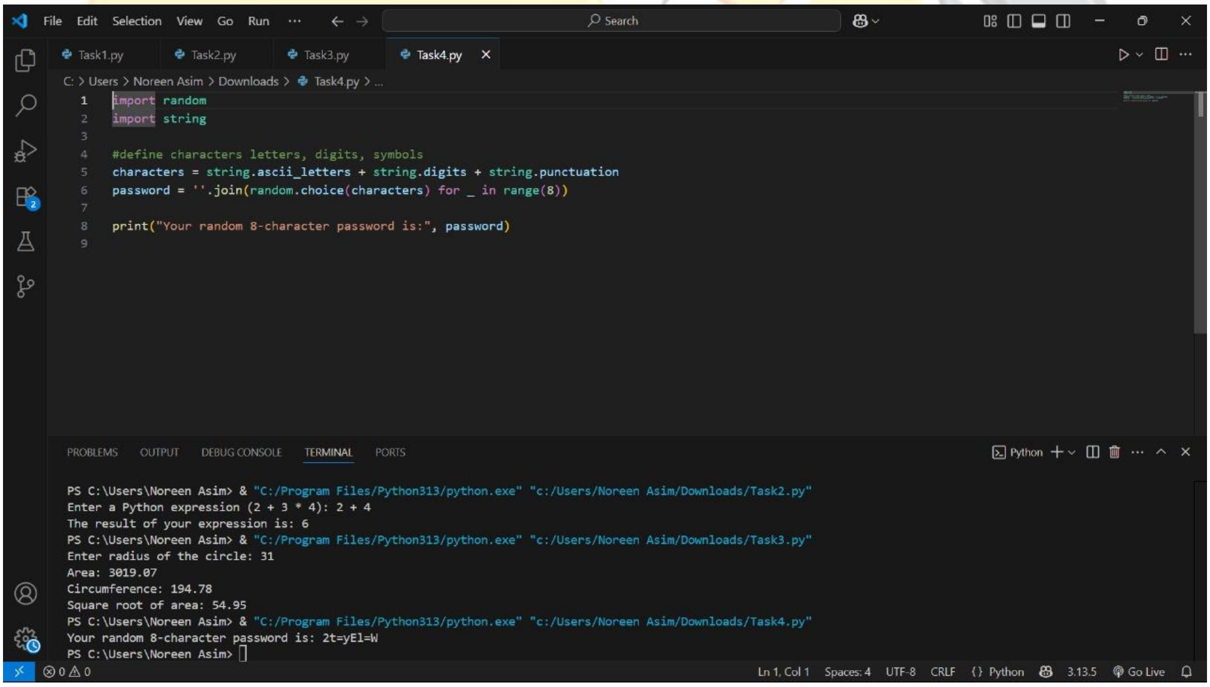
□ Confused between .format() and f-strings while formatting output.

□ Didn’t round float input which sometimes gave long decimal outputs.

Task 4 :

Code Snippet & Screenshot

```
import random

import string


#define characters letters, digits, symbols

characters = string.ascii_letters + string.digits + string.punctuation

password = ''.join(random.choice(characters) for _ in range(8))


print("Your random 8-character password is:", password)
```



```
import random
import string

#define characters letters, digits, symbols
characters = string.ascii_letters + string.digits + string.punctuation
password = ''.join(random.choice(characters) for _ in range(8))

print("Your random 8-character password is:", password)
```

```
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task2.py"
Enter a Python expression (2 + 3 * 4): 2 + 4
The result of your expression is: 6
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task3.py"
Enter radius of the circle: 31
Area: 3019.07
Circumference: 194.78
Square root of area: 54.95
PS C:\Users\Noreen Asim> & "C:/Program Files/Python313/python.exe" "c:/Users/Noreen Asim/Downloads/Task4.py"
Your random 8-character password is: 2t=yEl=W
PS C:\Users\Noreen Asim> 
```

Challenges:

□ Difficulty limiting symbols to avoid special characters that break usage (e.g., quotes).

□ Password sometimes included unreadable characters — needed filtering.

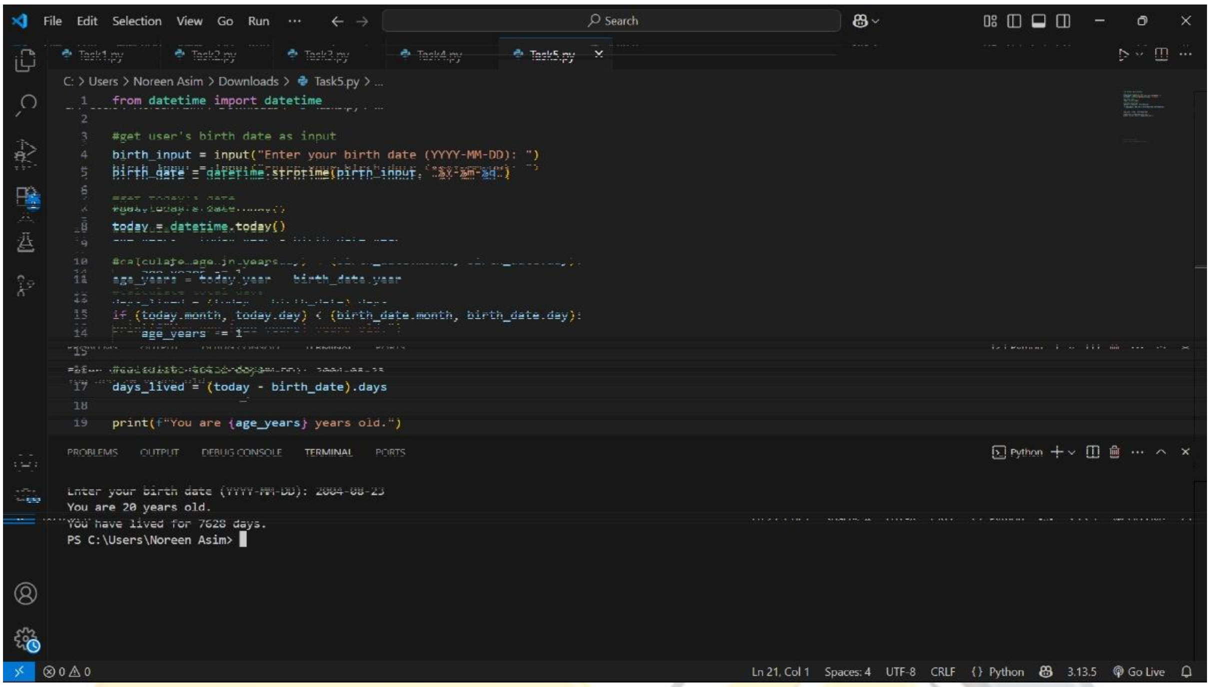
□ Trouble understanding how string concatenation and character pools work in random.choice

.

Task 5 :

Code Snippet & Screenshot

```
from datetime import datetime


#get user's birth date as input

birth_input = input("Enter your birth date (YYYY-MM-DD): ")

birth_date = datetime.strptime(birth_input, "%Y-%m-%d")


#get today's date

today = datetime.today()


#calculate age in years

age_years = today.year - birth_date.year


if (today.month, today.day) < (birth_date.month, birth_date.day):

    age_years -= 1


#calculate total days

days_lived = (today - birth_date).days


print(f"You are {age_years} years old.")

print(f"You have lived for {days_lived} days.")
```



```python
from datetime import datetime

#get user's birth date as input
birth_input = input("Enter your birth date (YYYY-MM-DD): ")
birth_date = datetime.strptime(birth_input, "%Y-%m-%d")

#get today's date
today = datetime.today()

#calculate age in years
age_years = today.year - birth_date.year


if (today.month, today.day) < (birth_date.month, birth_date.day):
    age_years -= 1

#calculate total days
days_lived = (today - birth_date).days

print(f"You are {age_years} years old.")
```

```
Enter your birth date (YYYY-MM-DD): 2004-08-23
You are 20 years old.
You have lived for 7628 days.
PS C:\Users\Noreen Asim>
```

Challenges:

- Error when input format wasn’t strictly YYYY-MM-DD; needed date validation.
- Confused by how .strptime works — initially used wrong format codes.
- Misunderstood date subtraction result (assumed it returned int, not timedelta).

Task 6 :

Code Snippet & Screenshot

import os

import re

folder = r"C:\Users\fNoreen Asim.\Desktop\Important\PowerPair Sem 4\PowerPair Sem 3\Maaz"

if os.path.exists(folder):

    print("\nMatching .txt files starting with 'report':\n")

    for file in os.listdir(folder):

        if file.endswith(".txt") and file.lower().startswith("report"):

            print(file)

```
else:
    print("Folder not found. Please check the path and try again.")
```

Challenges:

- Used startswith("Report") instead of lower().startswith() — missed files due to case sensitivity.
- Encountered PermissionError or FileNotFoundError due to incorrect folder access permissions.
- Didn’t know how to filter files using regular expressions (re.match) properly.