Name: Musfira Ahmed

Intern ID: TN/IN01/PY/002

Email ID : ahmedmusfira3@gmail.com

Internship Domain : Python Development

Task Week : 02

Instructor Name : Hassan Ali

Task 1 :

Create a mini profile for a fictional user using variables. Store the following information:

Full name , Age , Current year, Country, Hobby, Expected graduation year (calculate it from current year + 4)

Print all details in a proper sentence format.

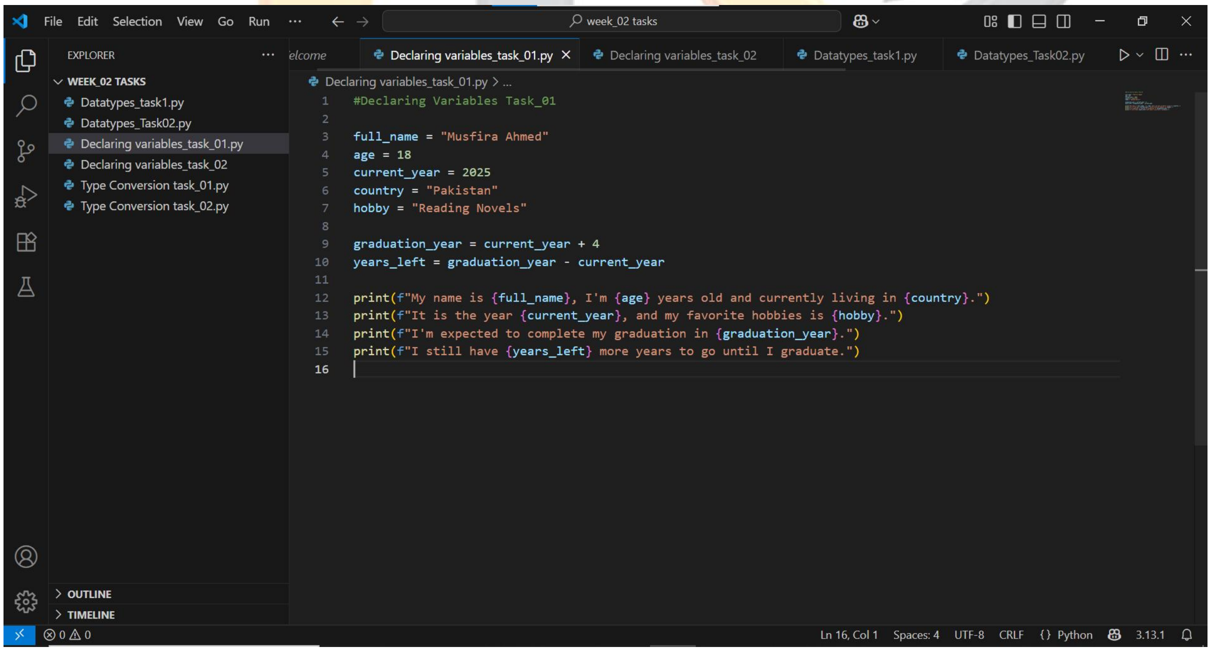
Also print how many years are left till graduation.

Solution :

What I Did (Step by Step):

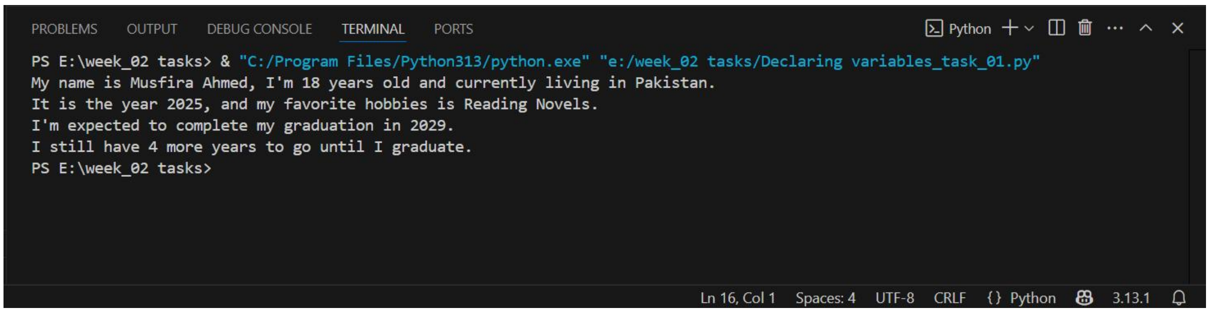
1. Defined variables for user profile (name, age, year, etc.).
2. Calculated the expected graduation year.
3. Printed the details in proper sentence format.

Code Screenshots


File
Edit
Selection
View
Go
Run
week_02 tasks
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
Type Conversion task_01.py
Type Conversion task_02.py
OUTLINE
TIMELINE
Welcome
Declaring variables_task_01.py
Declaring variables_task_02
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py > ...
Ln 16, Col 1
Spaces: 4
UTF-8
CRLF
{} Python
3.13.1

```python
#Declaring Variables Task_01

full_name = "Musfira Ahmed"
age = 18
current_year = 2025
country = "Pakistan"
hobby = "Reading Novels"

graduation_year = current_year + 4
years_left = graduation_year - current_year

print(f"My name is {full_name}, I'm {age} years old and currently living in {country}.")
print(f"It is the year {current_year}, and my favorite hobbies is {hobby}.")
print(f"I'm expected to complete my graduation in {graduation_year}.")
print(f"I still have {years_left} more years to go until I graduate.")

```

Output Screenshot



```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/Declaring variables_task_01.py"
My name is Musfira Ahmed, I'm 18 years old and currently living in Pakistan.
It is the year 2025, and my favorite hobbies is Reading Novels.
I'm expected to complete my graduation in 2029.
I still have 4 more years to go until I graduate.
PS E:\week_02 tasks>
```

Ln 16, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

Learnings and Challenges:

1) Learned about variables, string formatting using f-strings.
2) Practiced combining text and calculations in output.

Task 02:

Design a command-line survey that:

Asks the user 5 different questions (e.g., name, favorite food, birth year, favorite number, favorite language)

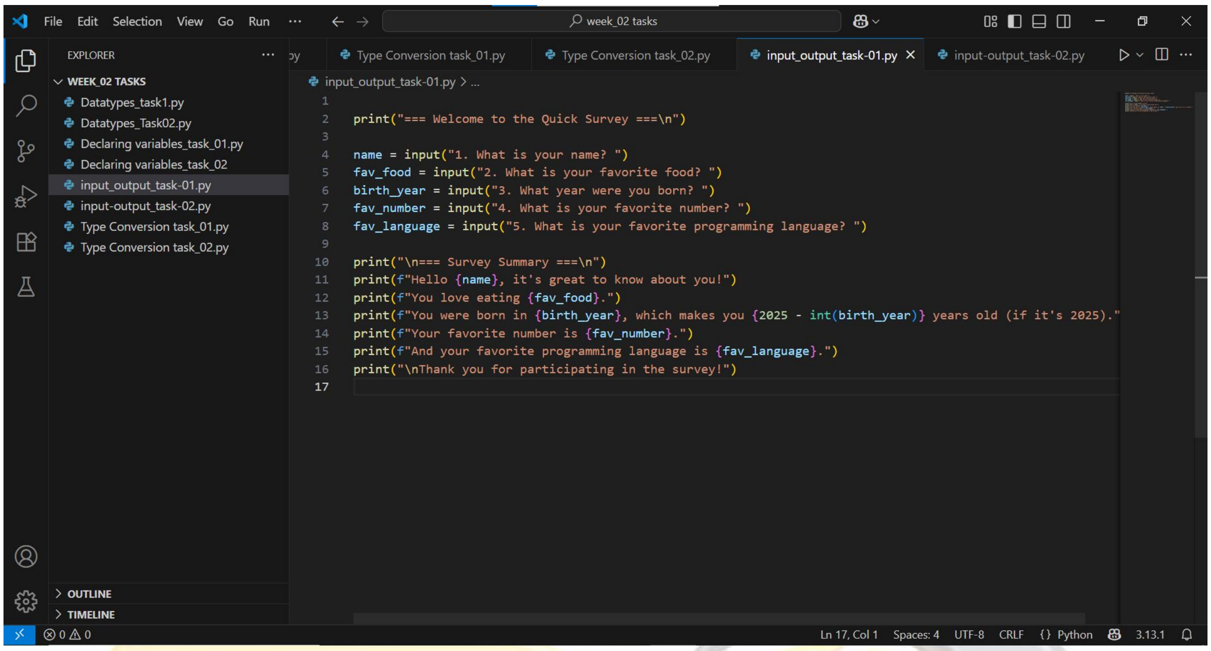
Then prints a summary of all responses in sentence format.

Use formatting to make the output look professional (e.g., f-strings).

What I Did (Step by Step):

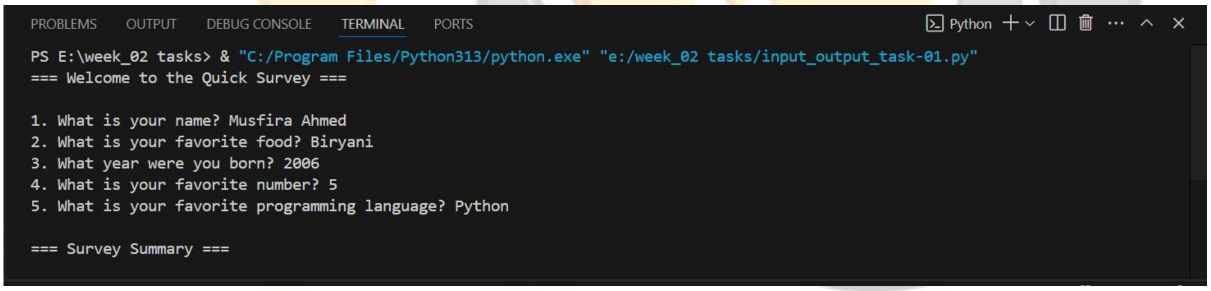
1. Collected user input using input() for name, food, birth year, number, and language.
2. Converted birth year to int and calculated age using 2025 - birth_year.
3. Used f-strings for a clean, formatted summary with emojis for better readability.

Code Screenshots

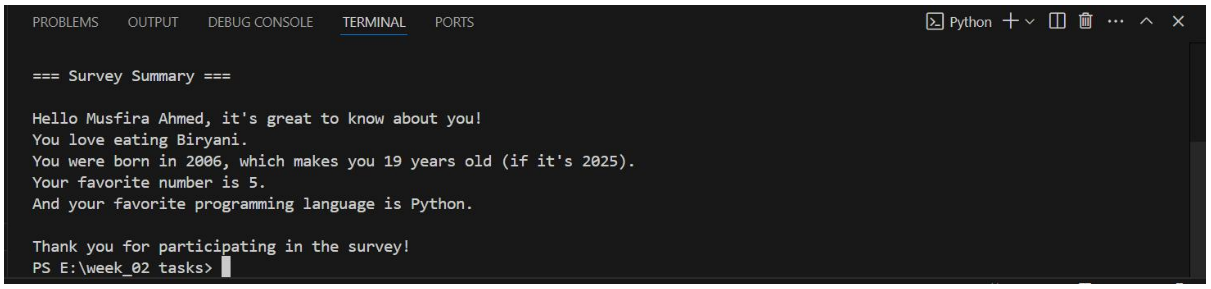

File Edit Selection View Go Run
week_02 tasks
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
input_output_task-01.py
input-output_task-02.py
Type Conversion task_01.py
Type Conversion task_02.py
OUTLINE
TIMELINE
Ln 17, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.1

```python

print("=== Welcome to the Quick Survey ===\n")

name = input("1. What is your name? ")
fav_food = input("2. What is your favorite food? ")
birth_year = input("3. What year were you born? ")
fav_number = input("4. What is your favorite number? ")
fav_language = input("5. What is your favorite programming language? ")

print("\n=== Survey Summary ===\n")
print(f"Hello {name}, it's great to know about you!")
print(f"You love eating {fav_food}.")
print(f"You were born in {birth_year}, which makes you {2025 - int(birth_year)} years old (if it's 2025).")
print(f"Your favorite number is {fav_number}.")
print(f"And your favorite programming language is {fav_language}.")
print("\nThank you for participating in the survey!")

```

Output Screenshot



```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/input_output_task-01.py"
=== Welcome to the Quick Survey ===

1. What is your name? Musfira Ahmed
2. What is your favorite food? Biryani
3. What year were you born? 2006
4. What is your favorite number? 5
5. What is your favorite programming language? Python

=== Survey Summary ===
```



```
=== Survey Summary ===

Hello Musfira Ahmed, it's great to know about you!
You love eating Biryani.
You were born in 2006, which makes you 19 years old (if it's 2025).
Your favorite number is 5.
And your favorite programming language is Python.

Thank you for participating in the survey!
PS E:\week_02 tasks> 
```

Learnings and Challenges:

1. Learned to collect user input and store it in variables.
2. Practiced converting strings to integers for age calculation.
3. Improved use of f-strings for clean, personalized output.
4. Focused on user-friendly formatting with emojis and spacing.

Task 03:

Ask the user to:

Enter their year of birth , Calculate their age (based on current year) ,Check if the user is eligible to vote (18+ years)

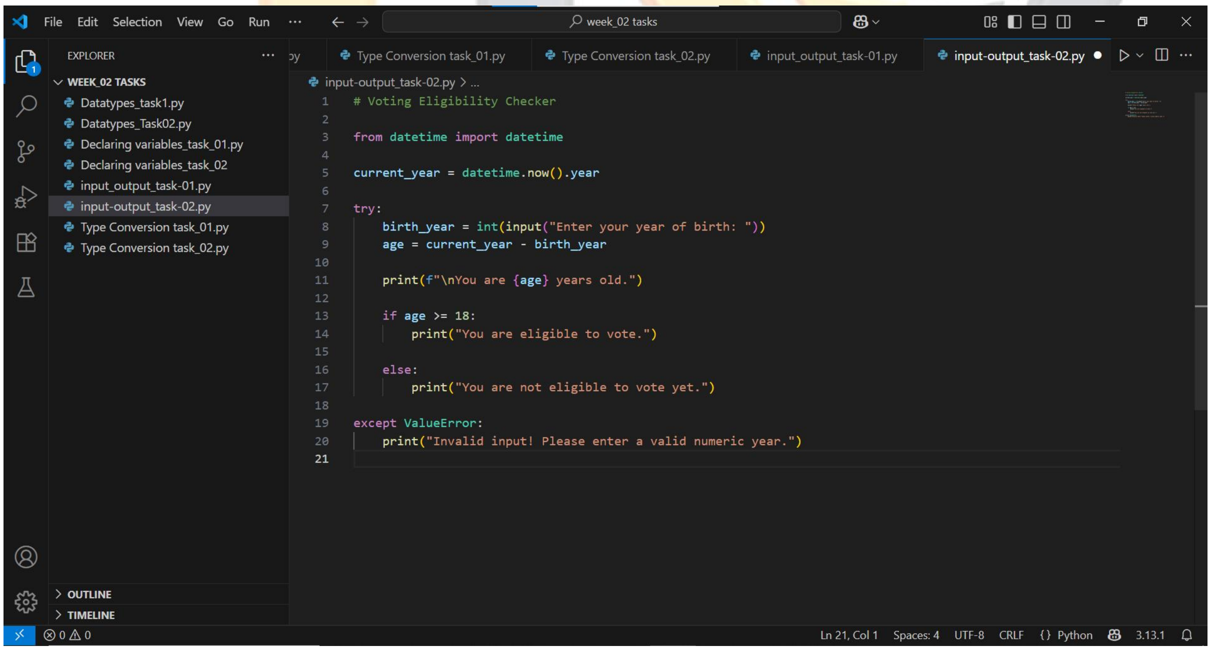
Display a message:

"You are eligible to vote." or **"You are not eligible to vote yet."**

What I Did (Step by Step):

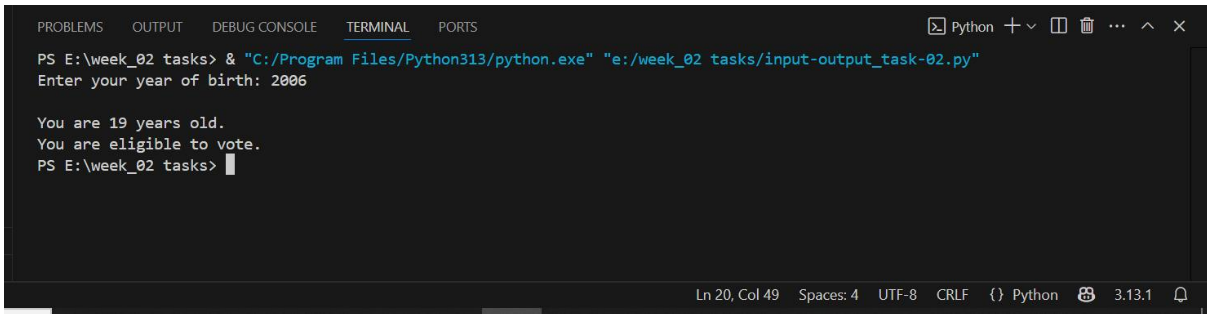
1) Asks for the user's year of birth,
2) Calculates their age,
3) Checks if they're eligible to vote (18+),
4) And prints a clear message based on the result.

Code Screenshots



```python
# Voting Eligibility Checker

from datetime import datetime

current_year = datetime.now().year

try:
    birth_year = int(input("Enter your year of birth: "))
    age = current_year - birth_year

    print(f"\nYou are {age} years old.")

    if age >= 18:
        print("You are eligible to vote.")

    else:
        print("You are not eligible to vote yet.")

except ValueError:
    print("Invalid input! Please enter a valid numeric year.")
```

Output Screenshot



```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/input-output_task-02.py"
Enter your year of birth: 2006

You are 19 years old.
You are eligible to vote.
PS E:\week_02 tasks>
```

Task 04:

Create 3 different user profiles (using variables). For each profile, include:

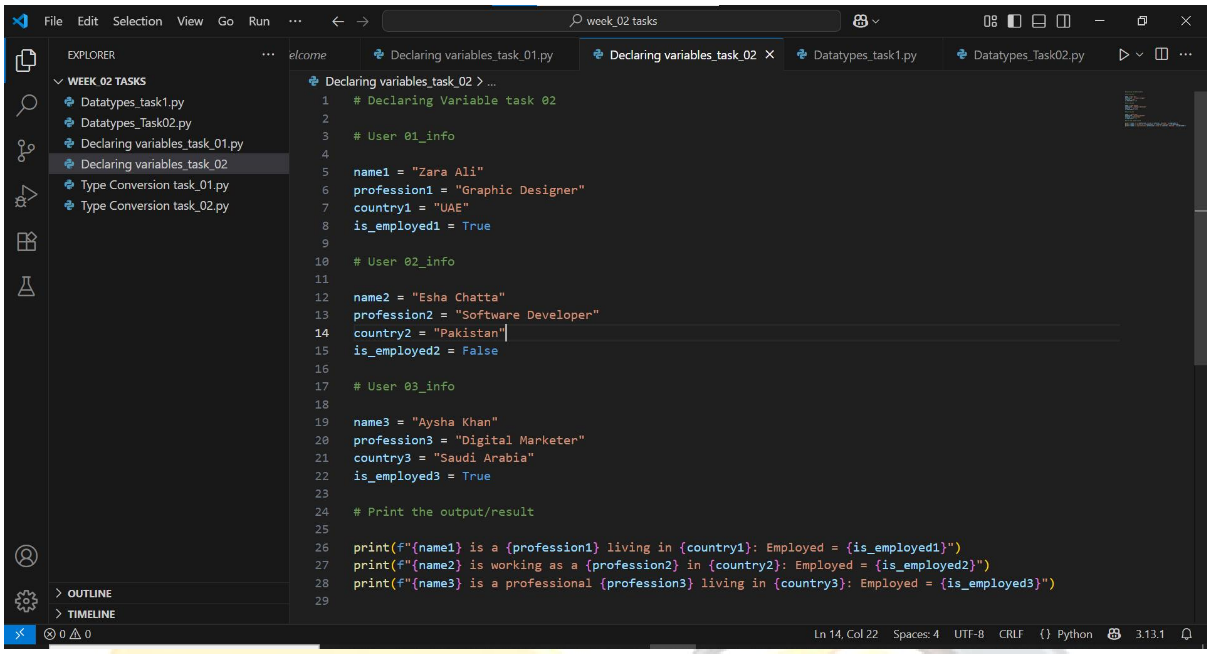
Name, profession, country, is_employed (Boolean)

Print their data in a tabular format using print() (not with external libraries).

What I Did (Step by Step):

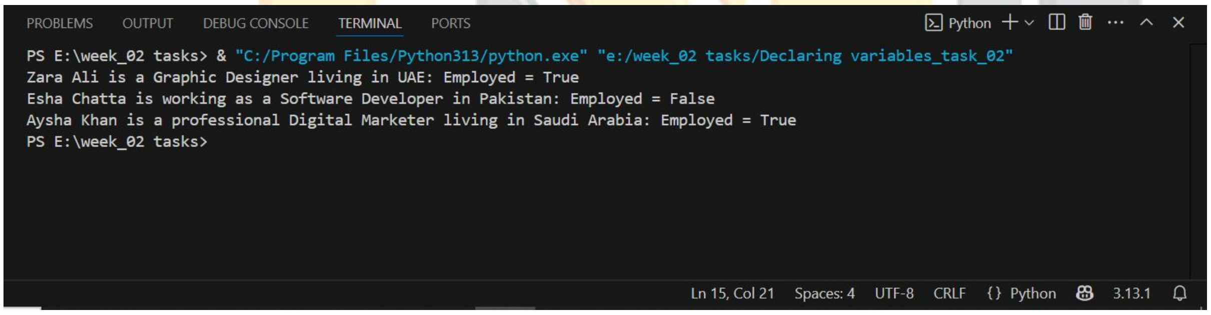
1) Created three user profiles with their details.
2) Printed all information in a structured format.

Code Screenshots


File Edit Selection View Go Run
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
Type Conversion task_01.py
Type Conversion task_02.py
OUTLINE
TIMELINE
Ln 14, Col 22 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```python
# Declaring Variable task 02

# User 01_info

name1 = "Zara Ali"
profession1 = "Graphic Designer"
country1 = "UAE"
is_employed1 = True

# User 02_info

name2 = "Esha Chatta"
profession2 = "Software Developer"
country2 = "Pakistan"
is_employed2 = False

# User 03_info

name3 = "Aysha Khan"
profession3 = "Digital Marketer"
country3 = "Saudi Arabia"
is_employed3 = True

# Print the output/result

print(f"{name1} is a {profession1} living in {country1}: Employed = {is_employed1}")
print(f"{name2} is working as a {profession2} in {country2}: Employed = {is_employed2}")
print(f"{name3} is a professional {profession3} living in {country3}: Employed = {is_employed3}")
```

Output Screenshots


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Ln 15, Col 21 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/Declaring variables_task_02"
Zara Ali is a Graphic Designer living in UAE: Employed = True
Esha Chatta is working as a Software Developer in Pakistan: Employed = False
Aysha Khan is a professional Digital Marketer living in Saudi Arabia: Employed = True
PS E:\week_02 tasks>
```

Learnings and Challenges:

1. Understood how to manage and format multiple sets of data.
2. Learned about boolean values and consistent formatting.

Task 05:

Write a program that:

Declares five different variables , Stores a different data type in each (e.g., string, integer, float, boolean, complex)

Prints their values and data types

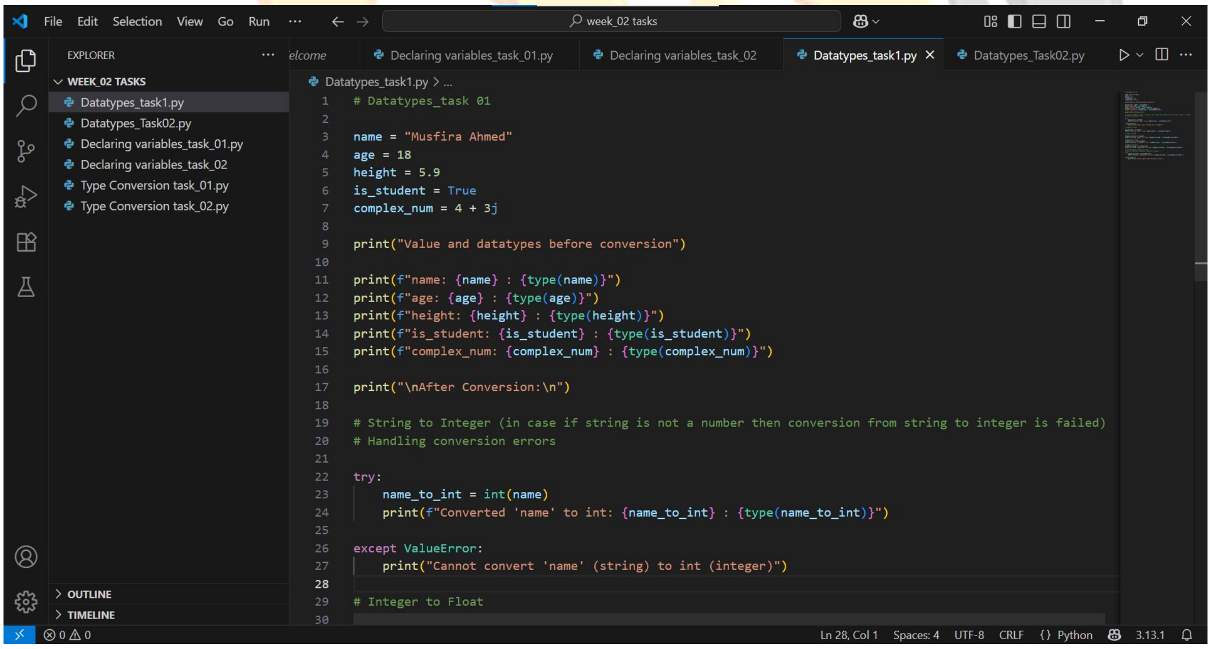
Then, converts each variable to a different type (where possible) and prints the new types

Note: You may not be able to convert all types — handle errors or comment why.

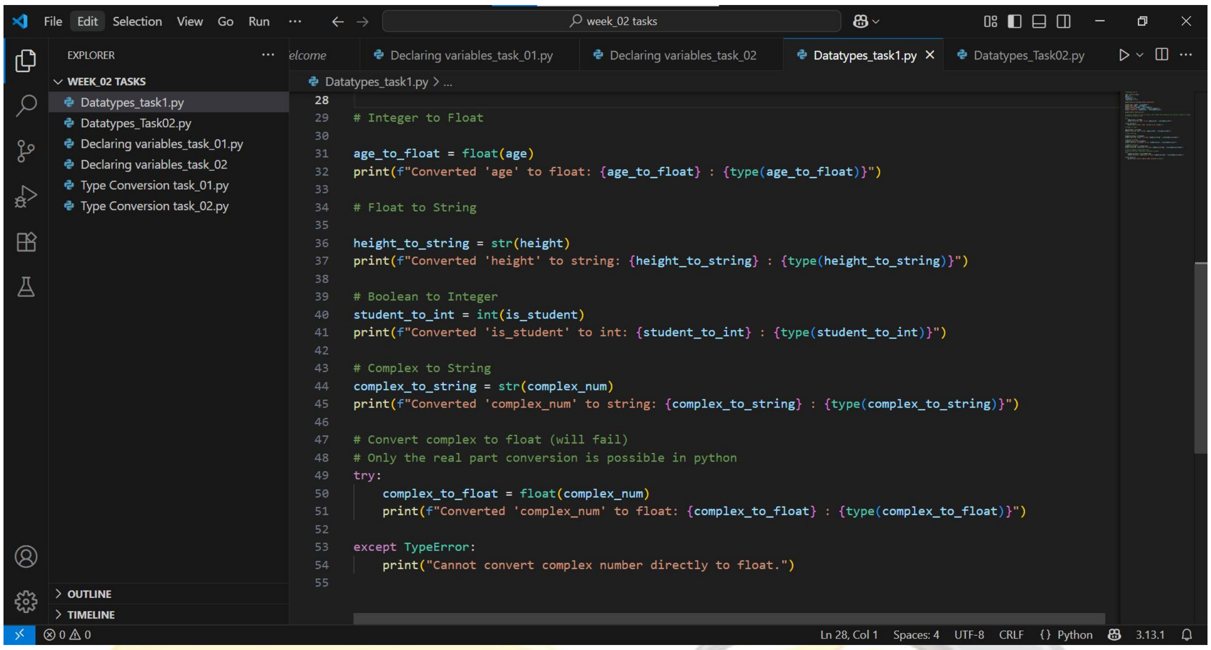
What I Did (Step by Step:

1) Declared variables of 5 different types.
2) Printed their original types.
3) Attempted conversions with try-except.

Code Screenshots

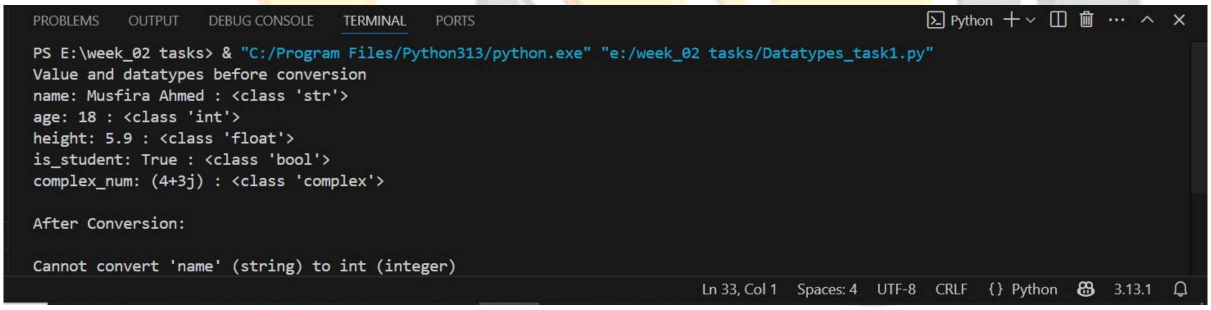

File Edit Selection View Go Run
week_02 tasks
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
Type Conversion task_01.py
Type Conversion task_02.py
OUTLINE
TIMELINE
Datatypes_task1.py > ...
Ln 28, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```python
# Datatypes_task 01

name = "Musfira Ahmed"
age = 18
height = 5.9
is_student = True
complex_num = 4 + 3j

print("Value and datatypes before conversion")

print(f"name: {name} : {type(name)}")
print(f"age: {age} : {type(age)}")
print(f"height: {height} : {type(height)}")
print(f"is_student: {is_student} : {type(is_student)}")
print(f"complex_num: {complex_num} : {type(complex_num)}")

print("\nAfter Conversion:\n")

# String to Integer (in case if string is not a number then conversion from string to integer is failed)
# Handling conversion errors

try:
    name_to_int = int(name)
    print(f"Converted 'name' to int: {name_to_int} : {type(name_to_int)}")

except ValueError:
    print("Cannot convert 'name' (string) to int (integer)")

# Integer to Float
```

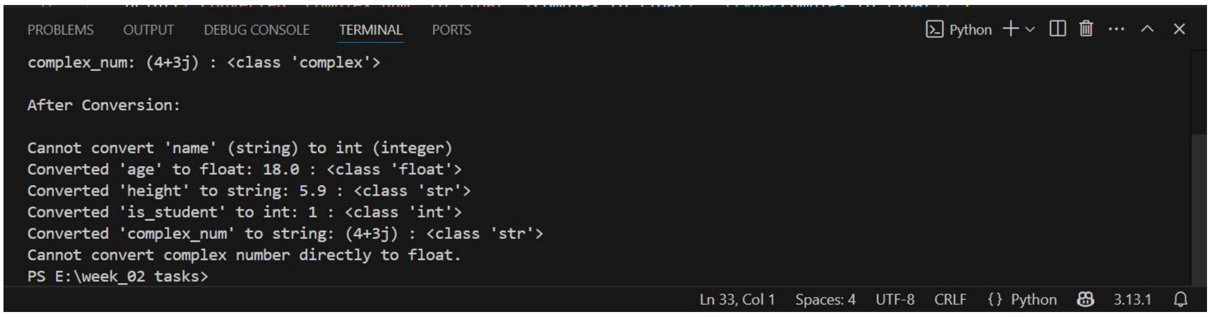


```python
# Integer to Float

age_to_float = float(age)
print(f"Converted 'age' to float: {age_to_float} : {type(age_to_float)}")

# Float to String

height_to_string = str(height)
print(f"Converted 'height' to string: {height_to_string} : {type(height_to_string)}")

# Boolean to Integer
student_to_int = int(is_student)
print(f"Converted 'is_student' to int: {student_to_int} : {type(student_to_int)}")

# Complex to String
complex_to_string = str(complex_num)
print(f"Converted 'complex_num' to string: {complex_to_string} : {type(complex_to_string)}")

# Convert complex to float (will fail)
# Only the real part conversion is possible in python
try:
    complex_to_float = float(complex_num)
    print(f"Converted 'complex_num' to float: {complex_to_float} : {type(complex_to_float)}")

except TypeError:
    print("Cannot convert complex number directly to float.")
```

Output Screenshots



```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/Datatypes_task1.py"
Value and datatypes before conversion
name: Musfira Ahmed : <class 'str'>
age: 18 : <class 'int'>
height: 5.9 : <class 'float'>
is_student: True : <class 'bool'>
complex_num: (4+3j) : <class 'complex'>

After Conversion:

Cannot convert 'name' (string) to int (integer)
```



```
complex_num: (4+3j) : <class 'complex'>

After Conversion:

Cannot convert 'name' (string) to int (integer)
Converted 'age' to float: 18.0 : <class 'float'>
Converted 'height' to string: 5.9 : <class 'str'>
Converted 'is_student' to int: 1 : <class 'int'>
Converted 'complex_num' to string: (4+3j) : <class 'str'>
Cannot convert complex number directly to float.
PS E:\week_02 tasks>
```

Learnings and Challenges:

1) Learned about data types and what conversions are allowed.
2) Learned that complex numbers can't be directly converted to float.
3) Improved error handling skills.

Task 06:

Create a data type tester:

Ask the user to input any value. , Detect and print what Python guesses its type as (use type()).

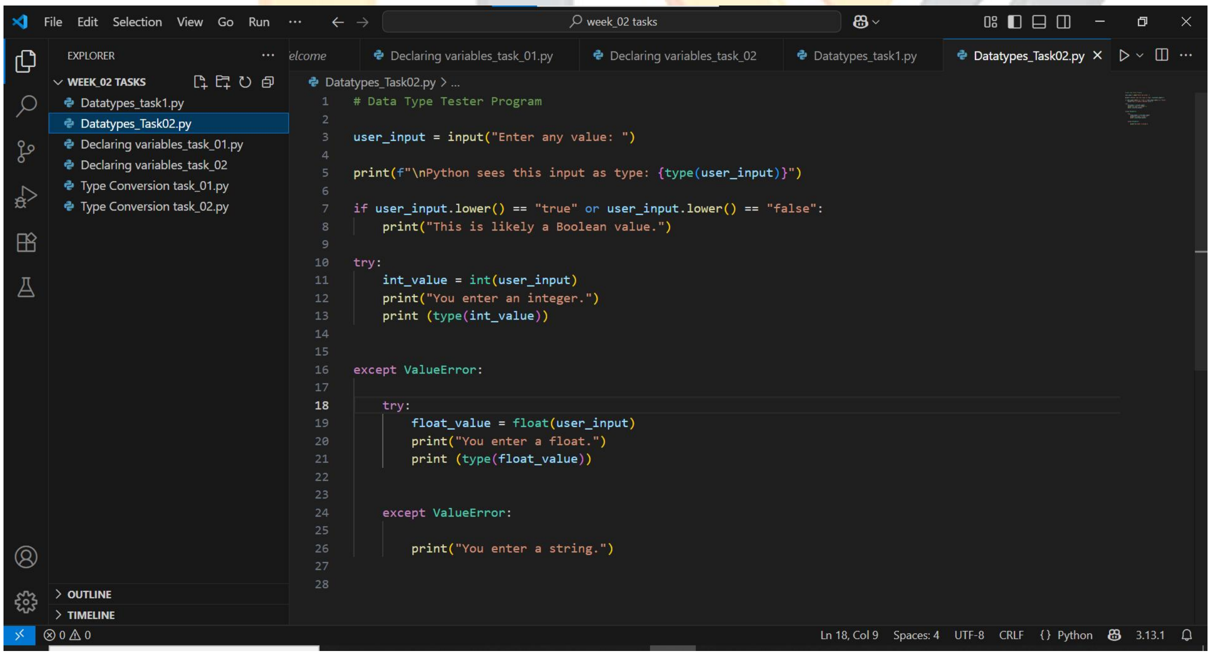
Add conditions to identify if it's likely an integer, float, or string, and print a message like:

"You entered a float!"

What I Did (Step by Step):

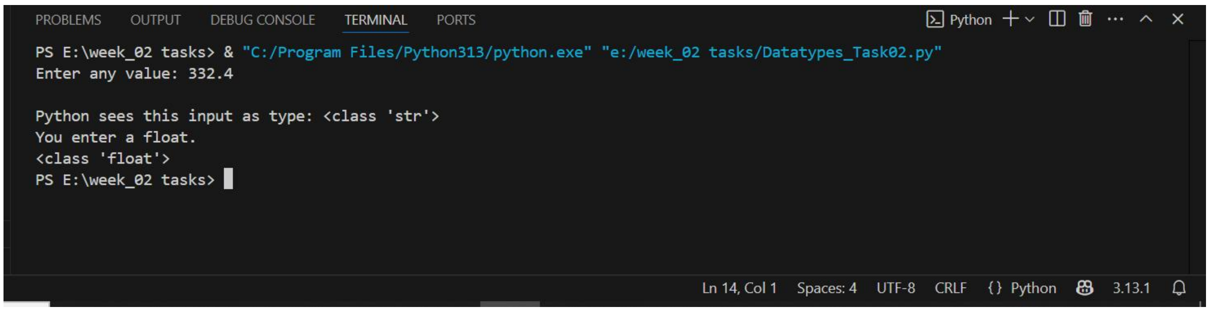
1. Took user input and checked the type.
2. Detected booleans, integers, floats, or strings.

Code Screenshots


File Edit Selection View Go Run ···
week_02 tasks
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
Type Conversion task_01.py
Type Conversion task_02.py
OUTLINE
TIMELINE
elcome
Declaring variables_task_01.py
Declaring variables_task_02
Datatypes_task1.py
Datatypes_Task02.py
Datatypes_Task02.py > ...
Ln 18, Col 9 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```python
# Data Type Tester Program

user_input = input("Enter any value: ")

print(f"\nPython sees this input as type: {type(user_input)}")

if user_input.lower() == "true" or user_input.lower() == "false":
    print("This is likely a Boolean value.")

try:
    int_value = int(user_input)
    print("You enter an integer.")
    print (type(int_value))


except ValueError:

    try:
        float_value = float(user_input)
        print("You enter a float.")
        print (type(float_value))


    except ValueError:

        print("You enter a string.")
```

Output Screenshots



```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/Datatypes_Task02.py"
Enter any value: 332.4

Python sees this input as type: <class 'str'>
You enter a float.
<class 'float'>
PS E:\week_02 tasks>
```

Learnings and Challenges:

1) Learnings and Challenges:
2) Learned how to determine the actual data type.
3) Handled errors smoothly using try-except.

Task 07:

Create a marks percentage calculator:

Ask user to input marks for 5 subjects (input as strings) , Convert them to integers

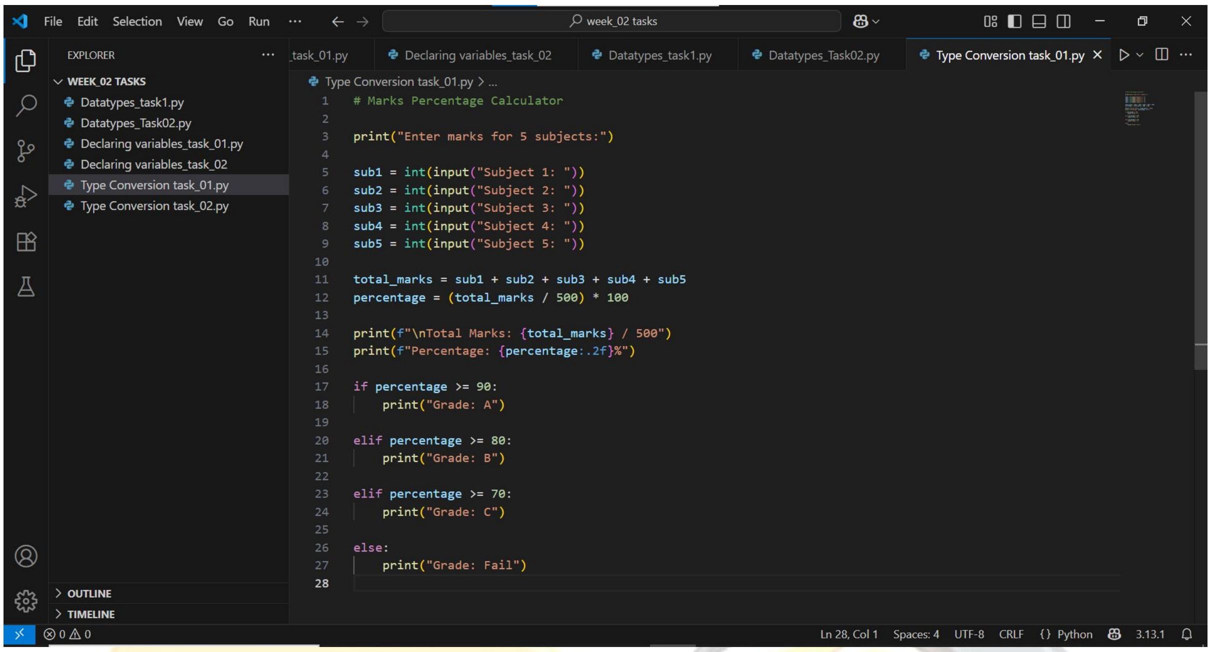
Calculate the total and percentage

Print percentage along with a grade: **A (90+), B (80-89), C (70-79), Fail (<70).**

What I Did (Step by Step):

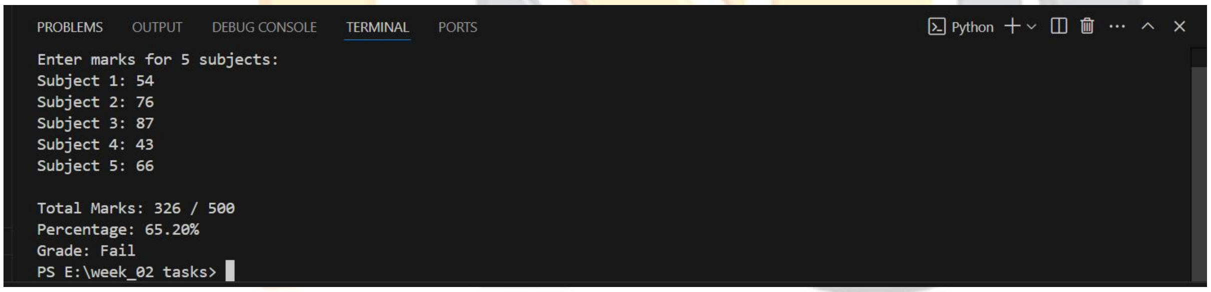
1. Collected marks, calculated total and percentage.
2. Used conditions to assign grades.

Code Screenshots



```python
# Marks Percentage Calculator

print("Enter marks for 5 subjects:")

sub1 = int(input("Subject 1: "))
sub2 = int(input("Subject 2: "))
sub3 = int(input("Subject 3: "))
sub4 = int(input("Subject 4: "))
sub5 = int(input("Subject 5: "))

total_marks = sub1 + sub2 + sub3 + sub4 + sub5
percentage = (total_marks / 500) * 100

print(f"\nTotal Marks: {total_marks} / 500")
print(f"Percentage: {percentage:.2f}%")

if percentage >= 90:
    print("Grade: A")

elif percentage >= 80:
    print("Grade: B")

elif percentage >= 70:
    print("Grade: C")

else:
    print("Grade: Fail")
```

Output Screenshots



```
Enter marks for 5 subjects:
Subject 1: 54
Subject 2: 76
Subject 3: 87
Subject 4: 43
Subject 5: 66

Total Marks: 326 / 500
Percentage: 65.20%
Grade: Fail
PS E:\week_02 tasks>
```

Learnings and Challenges:

1) Learned percentage calculations.
2) Practiced using if-elif-else logic.
3) Ensured that user input was correctly converted from string to integer.

Task 08:

Create a temperature converter:

Ask the user to input temperature in Celsius.

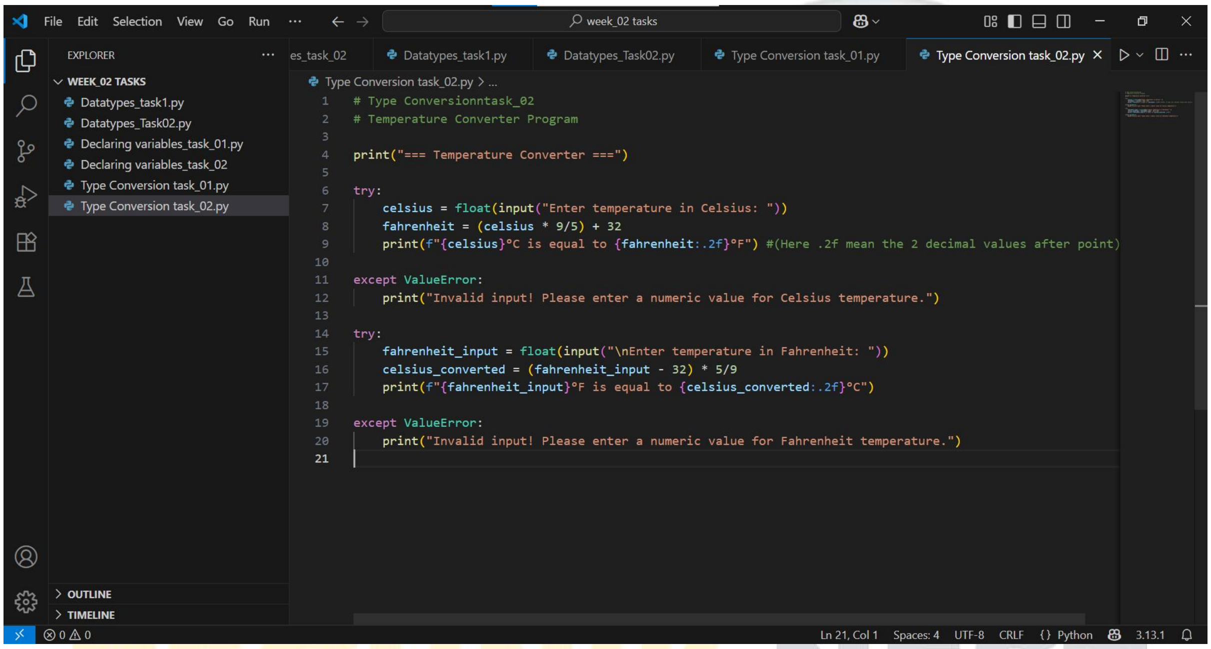
Convert it to Fahrenheit using: F = (C * 9/5) + 32 , Then reverse: Ask for Fahrenheit, convert it to Celsius.

Handle wrong input types using try-except.

What I Did (Step by Step):

1. Took input in Celsius and Fahrenheit.
2. Applied temperature conversion formulas.
3. Handled invalid inputs with try-except.

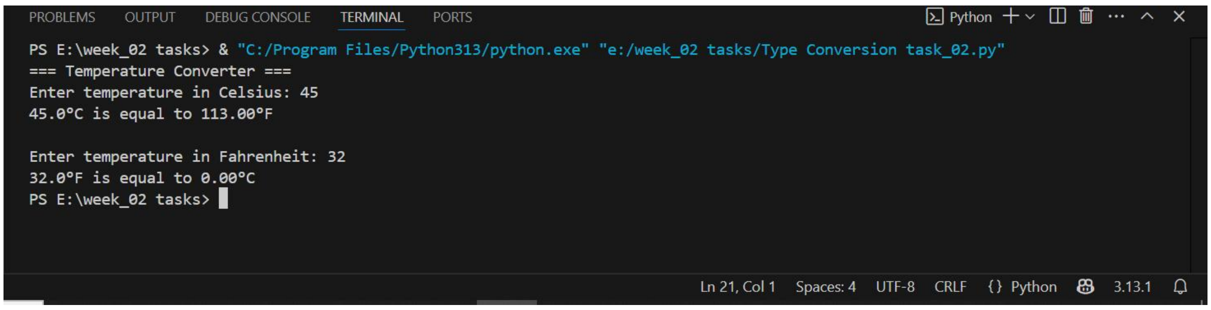
Code Screenshots



File Edit Selection View Go Run
week_02 tasks
EXPLORER
WEEK_02 TASKS
Datatypes_task1.py
Datatypes_Task02.py
Declaring variables_task_01.py
Declaring variables_task_02
Type Conversion task_01.py
Type Conversion task_02.py
Type Conversion task_02.py > ...
OUTLINE
TIMELINE
Ln 21, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```python
# Type Conversionntask_02
# Temperature Converter Program

print("=== Temperature Converter ===")

try:
    celsius = float(input("Enter temperature in Celsius: "))
    fahrenheit = (celsius * 9/5) + 32
    print(f"{celsius}°C is equal to {fahrenheit:.2f}°F") #(Here .2f mean the 2 decimal values after point)

except ValueError:
    print("Invalid input! Please enter a numeric value for Celsius temperature.")

try:
    fahrenheit_input = float(input("\nEnter temperature in Fahrenheit: "))
    celsius_converted = (fahrenheit_input - 32) * 5/9
    print(f"{fahrenheit_input}°F is equal to {celsius_converted:.2f}°C")

except ValueError:
    print("Invalid input! Please enter a numeric value for Fahrenheit temperature.")
```

Output Screenshots



PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Ln 21, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.1

```
PS E:\week_02 tasks> & "C:/Program Files/Python313/python.exe" "e:/week_02 tasks/Type Conversion task_02.py"
=== Temperature Converter ===
Enter temperature in Celsius: 45
45.0°C is equal to 113.00°F

Enter temperature in Fahrenheit: 32
32.0°F is equal to 0.00°C
PS E:\week_02 tasks>
```

Learnings and Challenges:

1) Practiced using mathematical formulas in real applications.
2) Learned how to use .2f for decimal formatting.
3) Improved program reliability using exception handling.