



TECHNIK NEST

INNOVATIVE MINDS, NESTING SUCCESS

Name: Marjan ikram

Intern ID: TN/IN01/PY/010

Email ID :marjanikram2005@gmail.com

Internship Domain : PYTHON

Task Week : 2

Instructor Name :Hassan ali

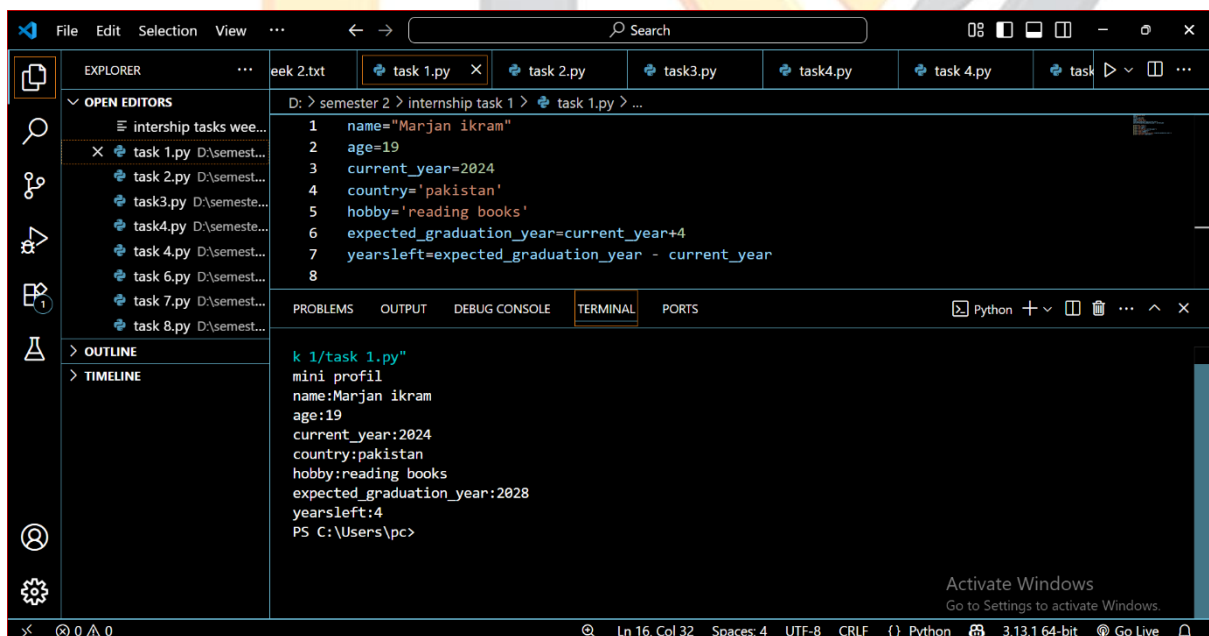
Task 1 :

Mini profile:

How I did it:

- I created a simple Python program that shows my personal mini profile.
- I stored details like name, age, country, and hobby using variables.
- I calculated the expected graduation year by adding 4 to the current year.
- I also calculated how many years are left until graduation.
- Finally i printed all the information.

Screenshot:



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project structure with files like 'task 1.py' through 'task 8.py'. The main editor area displays the code for 'task 1.py':

```
1 name="Marjan ikram"
2 age=19
3 current_year=2024
4 country='pakistan'
5 hobby='reading books'
6 expected_graduation_year=current_year+4
7 yearsleft=expected_graduation_year - current_year
8
```

Below the code editor, the TERMINAL panel shows the output of running the script:

```
k 1/task 1.py"
mini profil
name:Marjan ikram
age:19
current_year:2024
country:pakistan
hobby:reading books
expected_graduation_year:2028
yearsleft:4
PS C:\Users\pc>
```

The status bar at the bottom indicates the file is at line 16, column 32, using UTF-8 encoding and CRLF line endings, with the Python interpreter set to 3.13.1 64-bit.

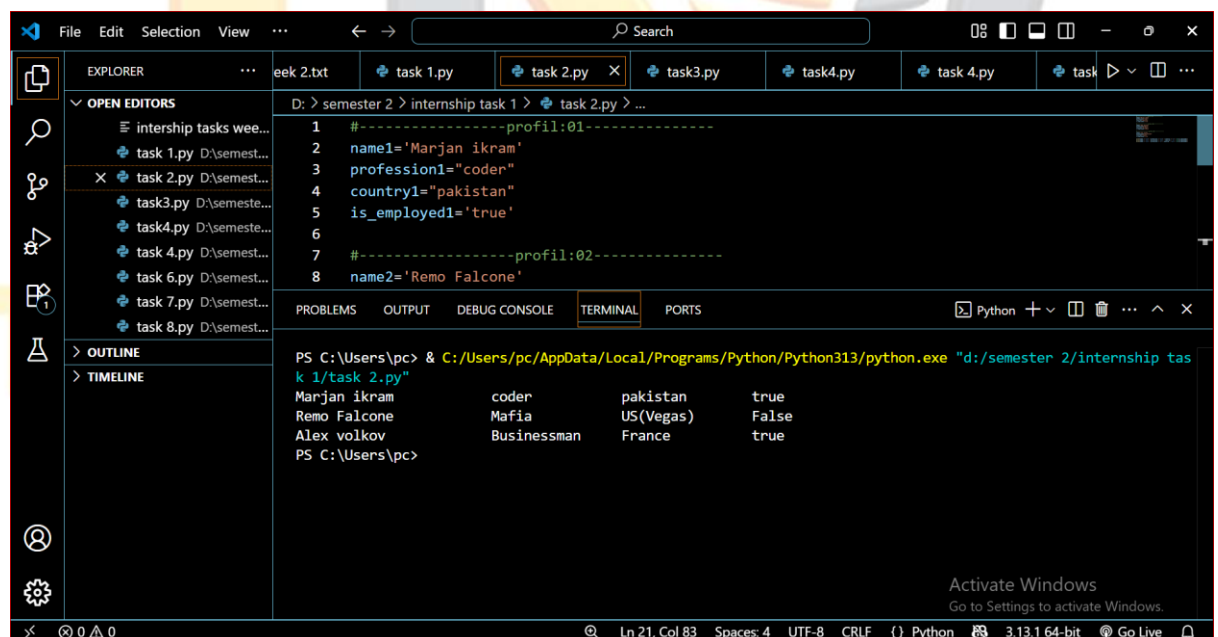
Task 2:

Creating 3 user profile:

How I did it:

- Firstly, I created 3 separate profiles using variables (name, profession, country, employment status).
- Each profile has values like name, profession, country, and whether the person is employed.
- In the I print all 3 profiles in a tabular format, line by line.

Screenshot:



The screenshot shows a Python IDE with a file explorer on the left and a terminal at the bottom. The code in the editor defines three user profiles using variables. The terminal output shows the profiles printed in a tabular format.

```
1 #-----profil:01-----
2 name1='Marjan ikram'
3 profession1="coder"
4 country1="pakistan"
5 is_employed1='true'
6
7 #-----profil:02-----
8 name2='Remo Falcone'
```

Name	Profession	Country	Employment Status
Marjan ikram	coder	pakistan	true
Remo Falcone	Mafia	US(Vegas)	False
Alex volkov	Businessman	France	true

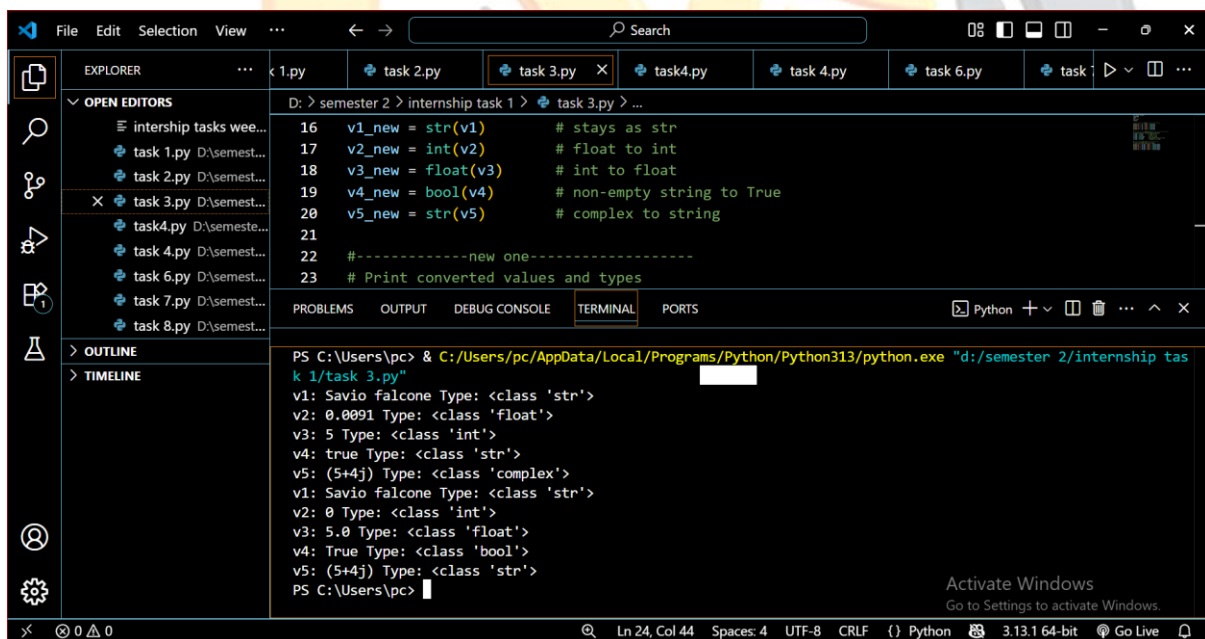
Task 03:

Data types and type conversion:

How I did:

- First of all I created 5 variables (v1 to v5) with different data types like string, float, integer, boolean and complex number.
- Then I used type() to print and check their original data types.
- I converted each variable to a different simple data type (e.g., float to int, int to float, etc.).
- Finally, I printed the converted values and their new data types..

Screenshot:



The screenshot shows a Python IDE with a dark theme. The Explorer pane on the left shows a project structure with files named task 1.py through task 8.py. The main editor window displays the following Python code:

```
16 v1_new = str(v1)      # stays as str
17 v2_new = int(v2)      # float to int
18 v3_new = float(v3)    # int to float
19 v4_new = bool(v4)     # non-empty string to True
20 v5_new = str(v5)      # complex to string
21
22 #-----new one-----
23 # Print converted values and types
```

Below the code editor, the TERMINAL pane shows the output of the program:

```
PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship task 1/task 3.py"
v1: Savio falcone Type: <class 'str'>
v2: 0.0091 Type: <class 'float'>
v3: 5 Type: <class 'int'>
v4: true Type: <class 'str'>
v5: (5+4j) Type: <class 'complex'>
v1: Savio falcone Type: <class 'str'>
v2: 0 Type: <class 'int'>
v3: 5.0 Type: <class 'float'>
v4: True Type: <class 'bool'>
v5: (5+4j) Type: <class 'str'>
```

The status bar at the bottom indicates the file is at line 24, column 44, using UTF-8 encoding and CRLF line endings. It also shows the Python version as 3.13.1 64-bit.

Task 04:

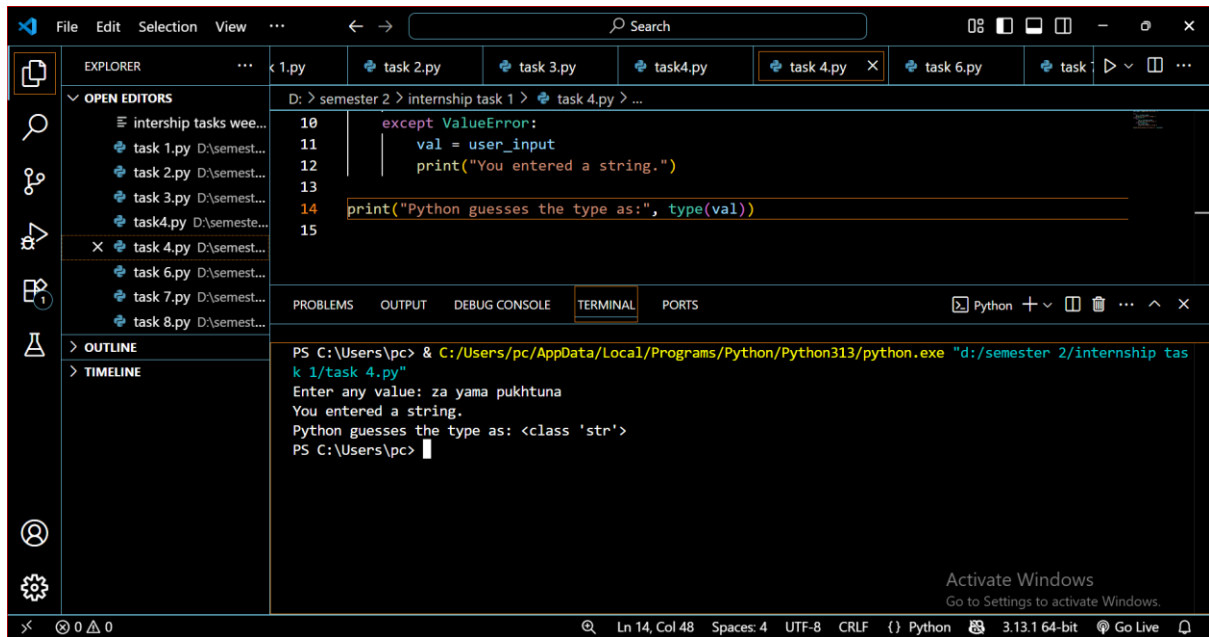
Data type tester:

How I did it:

- Takes input from the user using input().
- Tries to convert the input to an int.

- If that fails, it tries to convert the input to a float.
- If both conversions fail, it treats the input as a string.
- Prints the detected data type using `type()`.

Screenshot:



The screenshot shows a Python IDE with the following content:

EXPLORER: Lists files including `task 1.py`, `task 2.py`, `task 3.py`, `task 4.py`, `task 6.py`, `task 7.py`, and `task 8.py`.

EDITOR: Displays the code for `task 4.py`:

```

10
11     except ValueError:
12         val = user_input
13         print("You entered a string.")
14
15     print("Python guesses the type as:", type(val))

```

TERMINAL: Shows the execution output:

```

PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship tas
k 1/task 4.py"
Enter any value: za yama pukhtuna
You entered a string.
Python guesses the type as: <class 'str'>
PS C:\Users\pc>

```

The status bar at the bottom indicates: `Ln 14, Col 48`, `Spaces: 4`, `UTF-8`, `CRLF`, `{ } Python`, `3.13.1 64-bit`, and `Go Live`.

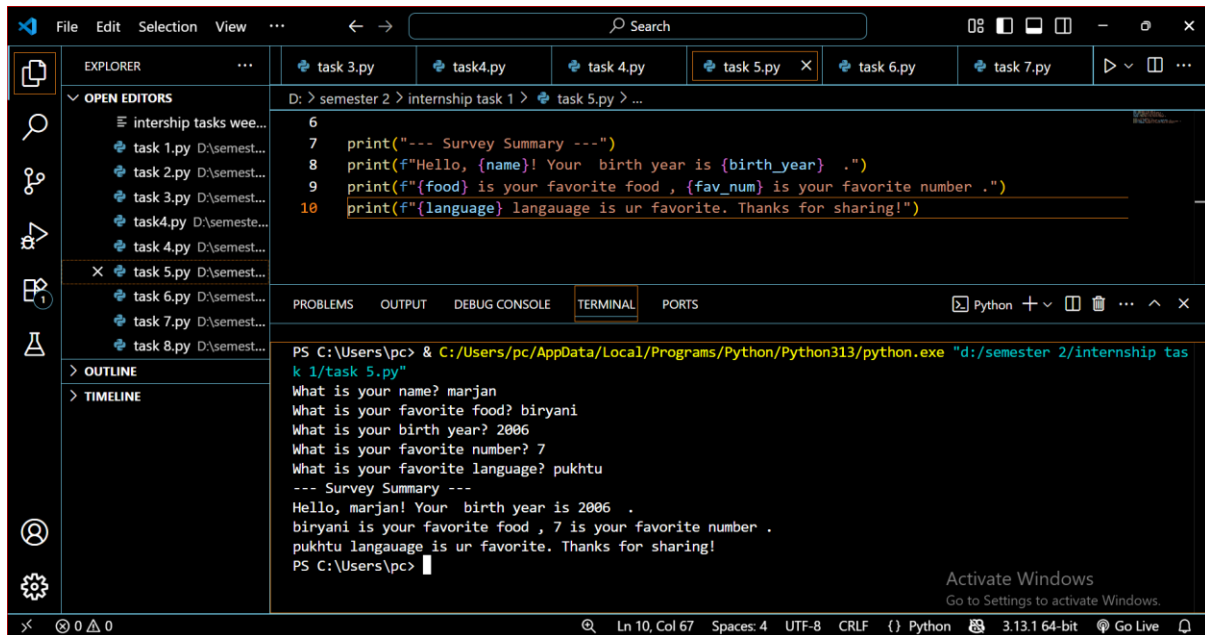
Task 05:

Command line survey:

How I did it:

- The code asks you some questions like your name, favorite food, birth year, etc.
- All answers are saved in different boxes (called variables).
- It then shows a heading: `--- Survey Summary ---`.
- After that, it prints a short message using your answers.
- In last it print the summary of what u have shared.

Screenshot:



The screenshot shows a Python IDE with a dark theme. The Explorer panel on the left lists several Python files, including 'task 5.py'. The main editor displays the code for 'task 5.py', which is a survey program. The code prompts the user for their name, favorite food, birth year, favorite number, and favorite language, then prints a summary and a greeting. The Terminal panel at the bottom shows the program's execution, with user input and the program's output. The status bar at the bottom indicates the file is at line 10, column 67, using UTF-8 encoding and CRLF line endings.

```
6
7 print("--- Survey Summary ---")
8 print(f"Hello, {name}! Your birth year is {birth_year} .")
9 print(f"{food} is your favorite food , {fav_num} is your favorite number .")
10 print(f"{language} langauage is ur favorite. Thanks for sharing!")
```

```
PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship task 1/task 5.py"
What is your name? marjan
What is your favorite food? biryani
What is your birth year? 2006
What is your favorite number? 7
What is your favorite language? pukhtu
--- Survey Summary ---
Hello, marjan! Your birth year is 2006 .
biryani is your favorite food , 7 is your favorite number .
pukhtu langauage is ur favorite. Thanks for sharing!
PS C:\Users\pc>
```

Task 06:

Calcualting age:

How I did it:

- The program asks you to enter your year of birth.
- It stores the current year as 2025.
- It calculates your age by subtracting your birth year from 2025.
- It prints how old you are.
- Then it checks:
 - If you are 18 or older, it says you can vote.
 - If you are under 18, it says you can't vote yet

Screenshot



The screenshot shows a Visual Studio Code editor window. The Explorer sidebar on the left lists several Python files named 'task 1.py' through 'task 8.py'. The main editor area displays the code for 'task 6.py', which is a simple age validation script. The code prompts the user for their age and prints a message based on whether they are 18 or older. The output pane at the bottom shows the script being executed in a terminal, with the user entering '2006' and the program outputting 'You are 19 years old.' and 'You are eligible to vote.'

```
4 print(f"You are {age} years old.")
5 if age >= 18:
6     print("You are eligible to vote.")
7 else:
8     print("You are not eligible to vote yet.")
```

```
PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship task 1/task 6.py"
Enter your year of birth: 2006
You are 19 years old.
You are eligible to vote.
PS C:\Users\pc>
```

Task 07:

Percentage :

How I did it:

- The code asks the user to enter marks (out of 100) for 5 subjects.
- It uses try-except to catch errors if the user enters anything other than numbers.
- It adds all 5 marks to calculate the total, then finds the percentage by dividing the total by 5.
- It prints the total marks, percentage, and assigns a grade based on the percentage:
- 90 or above → Grade A
- 80–89 → Grade B
- 70–79 → Grade C

- Below 70 → Fail
- If the user enters something that is not a number, it shows a message: "Please enter only numeric values for marks."

Screenshot:

```

18     else:
19         grade = "Fail"
20         print(f"Grade: {grade}")
21     except ValueError:
22         print("Please enter only numeric values for marks.")

```

```

PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship task 1/task 7.py"
Enter marks out of 100 for 5 subjects:
Subject 1: 78
Subject 2: 89
Subject 3: 90
Subject 4: 98
Subject 5: 87

Total Marks: 442/500
Percentage: 88.4%
Grade: B
PS C:\Users\pc>

```

Task 08:

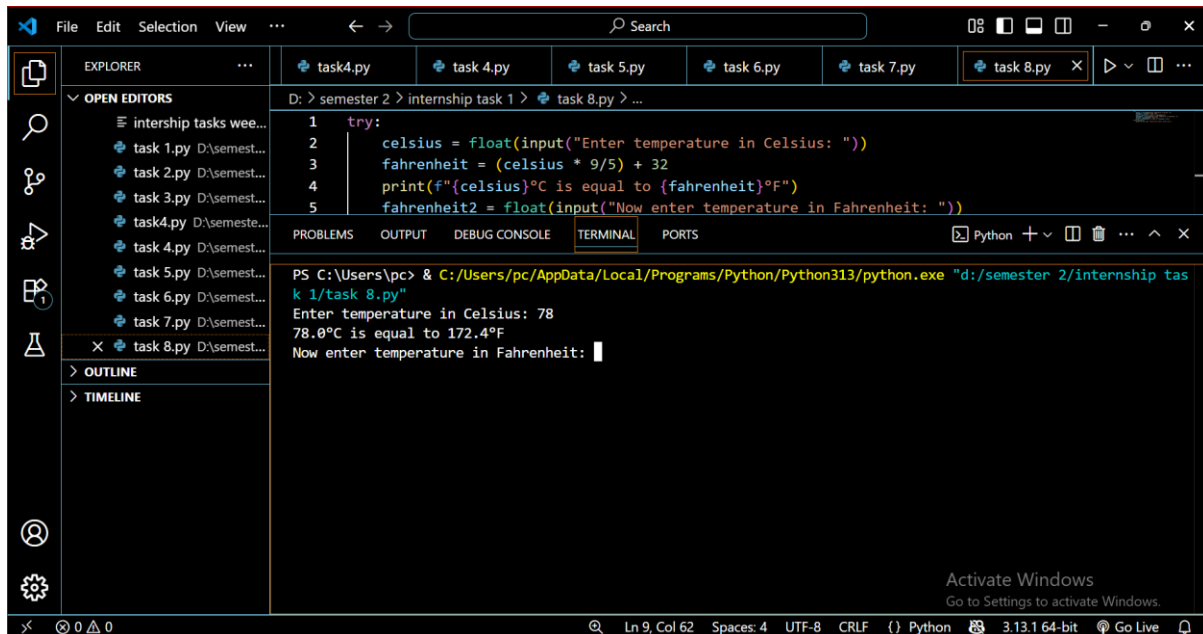
Celsius to Farenheit:

How I did it:

- The program asks the user to enter a temperature in Celsius.
- It converts the Celsius value to Fahrenheit using the formula: $(C \times 9/5) + 32$.
- Then it asks the user to enter a temperature in Fahrenheit.
- It converts the Fahrenheit value to Celsius using the formula: $(F - 32) \times 5/9$.

- If the user enters a non-numeric value, the try-except block catches the error and shows: "Invalid input. Please enter numeric values only."

Screenshot:



The screenshot shows the Visual Studio Code (VS Code) interface. The Explorer panel on the left shows a project structure with a folder named 'internship tasks' containing several Python files (task 1.py to task 8.py). The Open Editors panel shows the current file being edited is 'task 8.py'. The editor window displays the following Python code:

```
1 try:
2     celsius = float(input("Enter temperature in Celsius: "))
3     fahrenheit = (celsius * 9/5) + 32
4     print(f"{celsius}°C is equal to {fahrenheit}°F")
5     fahrenheit2 = float(input("Now enter temperature in Fahrenheit: "))
```

The terminal panel at the bottom shows the output of the script:

```
PS C:\Users\pc> & C:/Users/pc/AppData/Local/Programs/Python/Python313/python.exe "d:/semester 2/internship tas
k 1/task 8.py"
Enter temperature in Celsius: 78
78.0°C is equal to 172.4°F
Now enter temperature in Fahrenheit: |
```

The status bar at the bottom indicates the current line and column (Ln 9, Col 62), the number of spaces (4), the encoding (UTF-8), the line ending (CRLF), the language (Python), the version (3.13.1 64-bit), and the Go Live status.

TECHNIK NEST