Here's a Python solution for the weather tracking application:

Weather Tracking Code

import datetime

# Data structure to store weather data

weather\_data = []

def log\_weather(temperature, humidity, condition):

"""Log daily weather data"""

weather\_data.append((temperature, humidity, condition, datetime.date.today()))

def display\_summary():

"""Display weekly summary of average temperature and humidity"""

if len(weather\_data) < 7:

print("Insufficient data for weekly summary.")

return

temperatures = [data[0] for data in weather\_data[-7:]]

humidities = [data[1] for data in weather\_data[-7:]]

avg\_temp = sum(temperatures) / len(temperatures)

avg\_humidity = sum(humidities) / len(humidities)

print(f"Weekly Summary (Last 7 days):")

print(f"Average Temperature: {avg\_temp:.2f}°C")

print(f"Average Humidity: {avg\_humidity:.2f}%")

def check\_extreme\_conditions():

"""Generate alerts for extreme weather conditions"""

today\_data = weather\_data[-1]

temperature, humidity, condition = today\_data[0], today\_data[1], today\_data[2]

if temperature > 35:

print("Heatwave Alert!")

elif temperature < 0:

print("Freezing Alert!")

if humidity > 90 and condition == "Rainy":

print("Heavy Rain Alert!")

elif humidity < 30 and condition == "Sunny":

print("Dry Spell Alert!")

def main():

while True:

print("\nWeather Tracking App")

print("1. Log Weather Data")

print("2. Display Summary")

print("3. Check Extreme Conditions")

print("4. Exit")

choice = input("Choose an option: ")

if choice == "1":

temperature = float(input("Enter temperature (°C): "))

humidity = float(input("Enter humidity (%): "))

condition = input("Enter weather condition (Sunny, Rainy, Cloudy): ")

log\_weather(temperature, humidity, condition)

elif choice == "2":

display\_summary()

elif choice == "3":

check\_extreme\_conditions()

elif choice == "4":

break

else:

print("Invalid option. Please choose again.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Learning Outcomes

1. Data Handling: Students learn to handle time-series data using lists and tuples.

2. Conditional Statements: Students understand how to use conditionals to generate alerts for extreme weather conditions.

3. Tuple Assignment: Students learn to group daily temperature, humidity, and condition data using tuples for easier manipulation.

Variables for Daily Temperature, Humidity, and Weather Conditions

- temperature (float): Stores daily temperature in degrees Celsius.

- humidity (float): Stores daily humidity as a percentage.

- condition (str): Stores weather condition (Sunny, Rainy, Cloudy).

- date (datetime.date): Stores the date of the weather data.

Conditional Statements for Extreme Conditions

- Temperature > 35°C: Heatwave Alert!

- Temperature < 0°C: Freezing Alert!

- Humidity > 90% and condition == "Rainy": Heavy Rain Alert!

- Humidity < 30% and condition == "Sunny": Dry Spell Alert!

Using Only Conditionals (Without Functions)

While possible, using only conditionals without functions would lead to:

- Decreased readability due to repetitive code.

- Reduced flexibility, making it harder to modify or add new features.

- Increased maintenance time.

Tuple Assignment for Data Grouping

Tuple assignment helps group daily temperature, humidity, and condition data for easier manipulation. For example:

temperature, humidity, condition, date = weather\_data[-1]

This assigns the last logged weather data to individual variables, making it easier to access and use the data.

Let me know if you have any further questions or need additional clarification!