Weekend Task

1. Temperature Data Logger (2D Array)

\* Problem Statement: Design a program to log temperature readings from multiple sensors for 24 hours, sampled every hour.

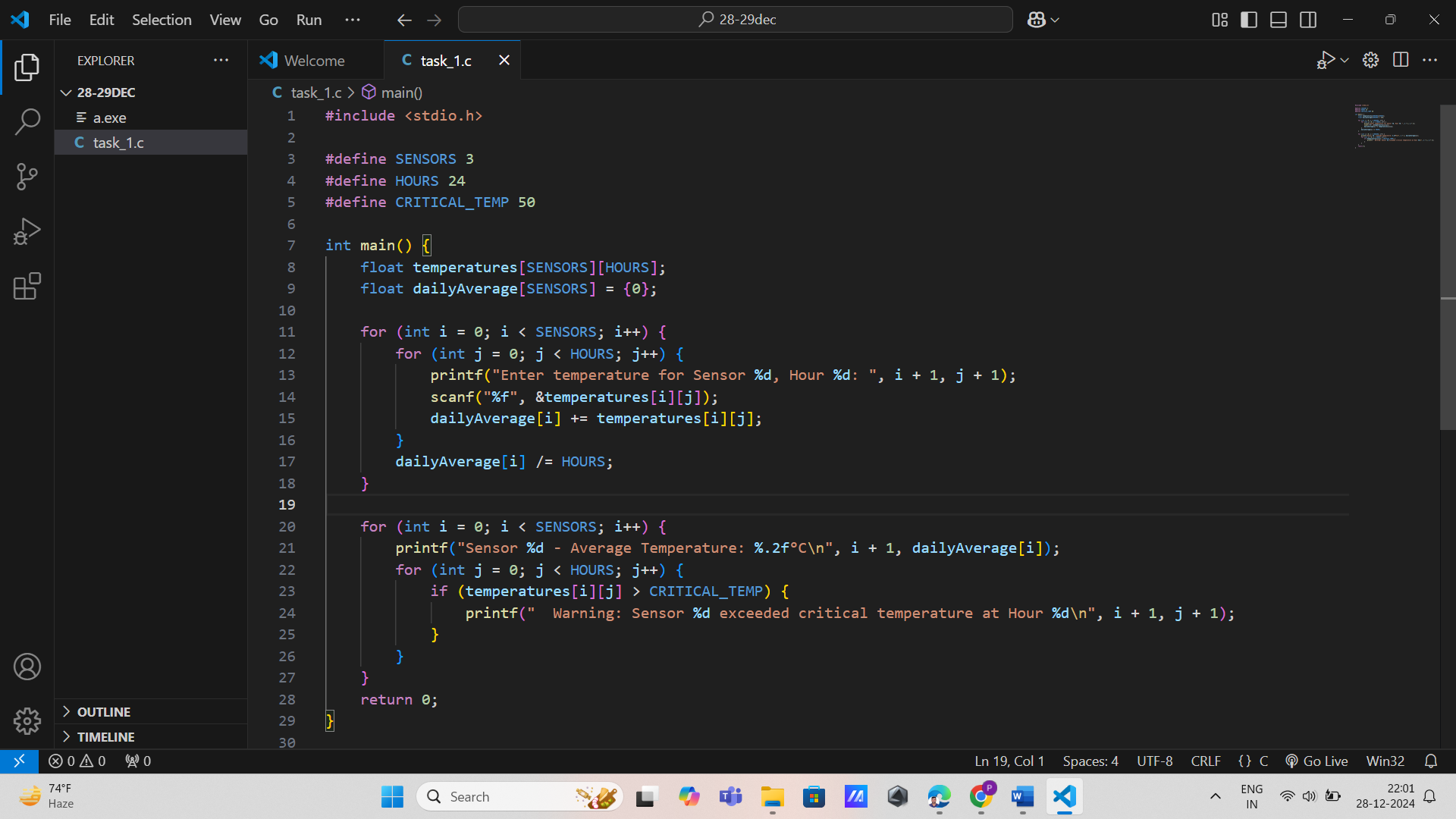
\* Requirements:

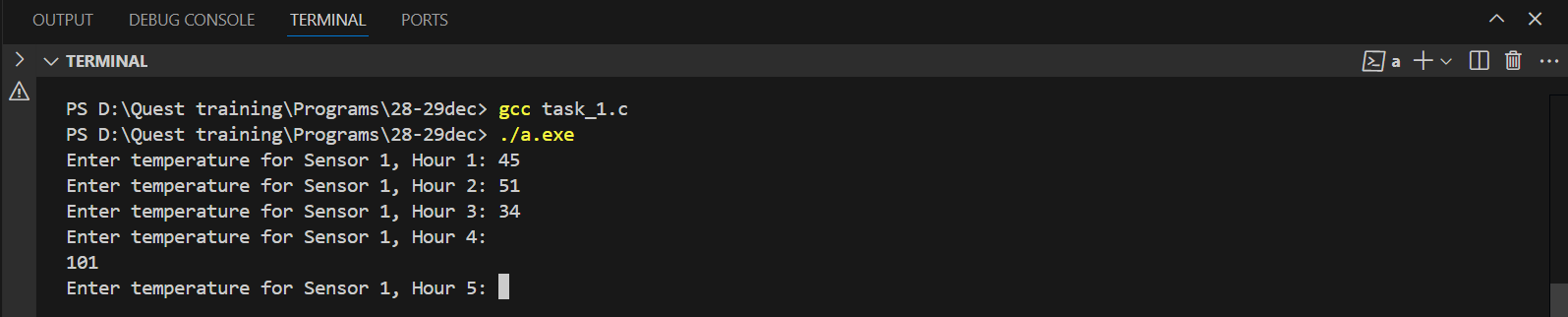
\* Use a 2D array of size [N][24] to store temperature data, where N is the number of sensors (defined as a const variable).

\* Use static variables to calculate and store the daily average temperature for each sensor.

\* Use nested for loops to populate and analyze the array.

\* Use if statements to identify sensors exceeding a critical threshold temperature.





2. LED Matrix Control (2D Array)

\* Problem Statement: Simulate the control of an LED matrix of size 8x8. Each cell in the matrix can be ON (1) or OFF (0).

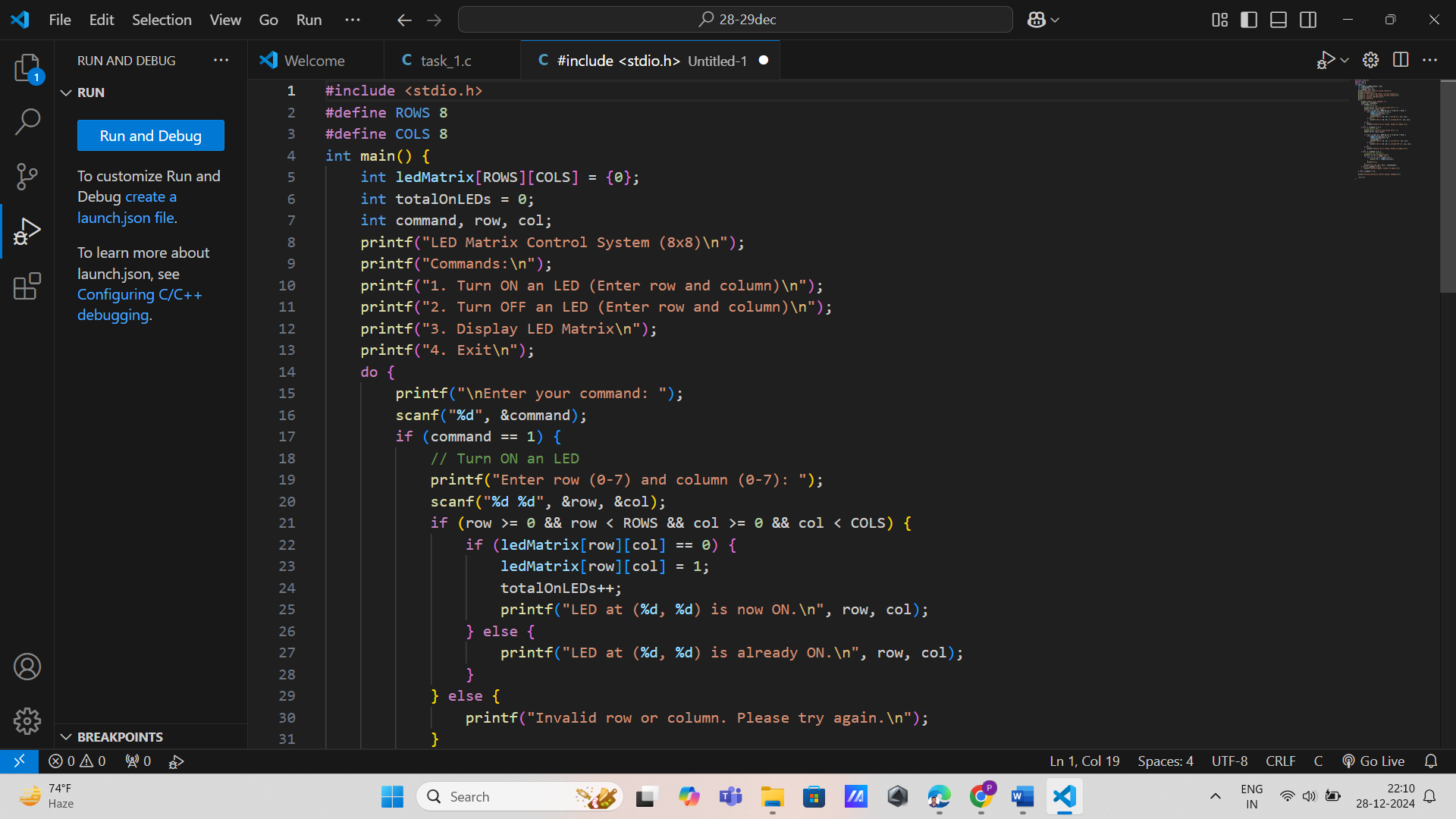
\* Requirements:

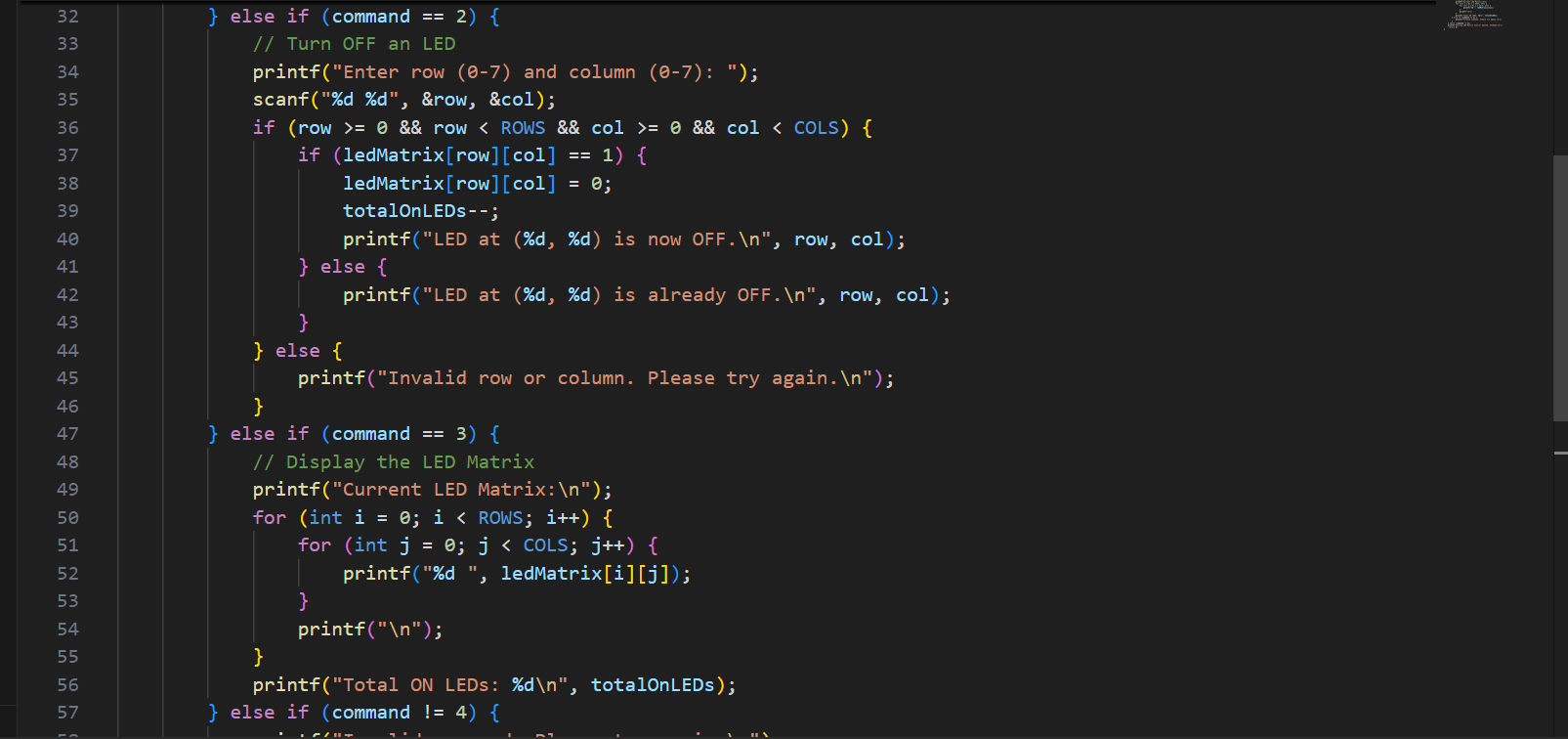
\* Use a 2D array to represent the LED matrix.

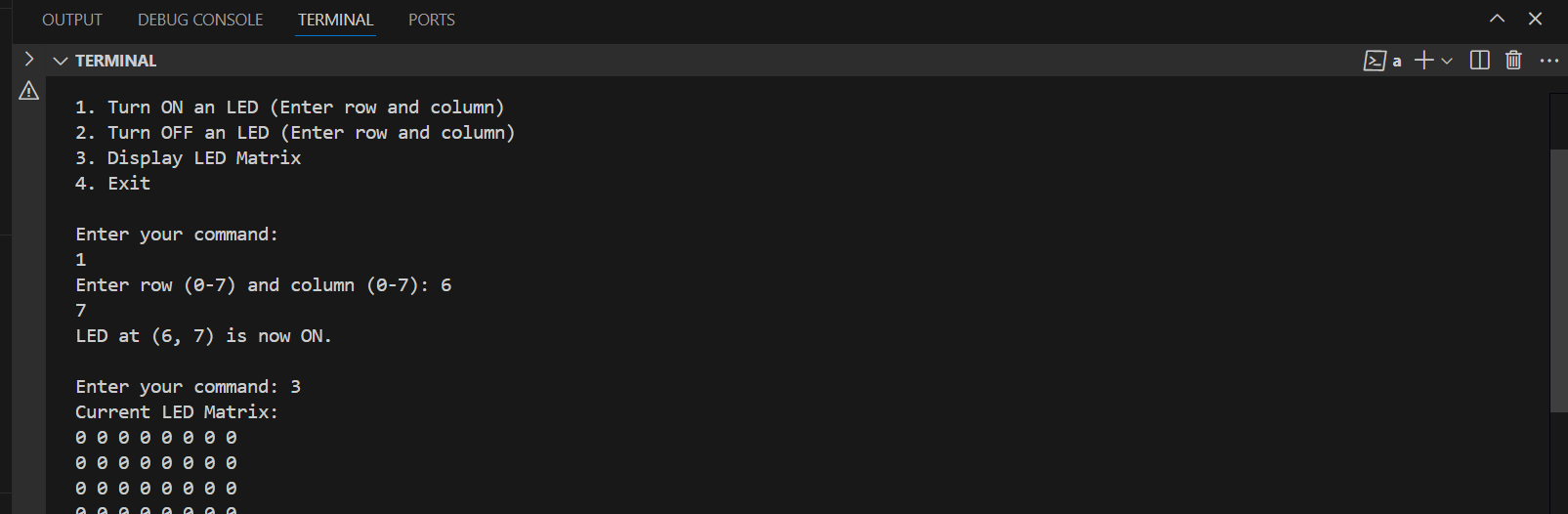
\* Use static variables to count the number of ON LEDs.

\* Use nested for loops to toggle the state of specific LEDs based on input commands.

\* Use if statements to validate commands (e.g., row and column indices).







3. Robot Path Mapping (2D Array)

\* Problem Statement: Track the movement of a robot on a grid of size M x N.

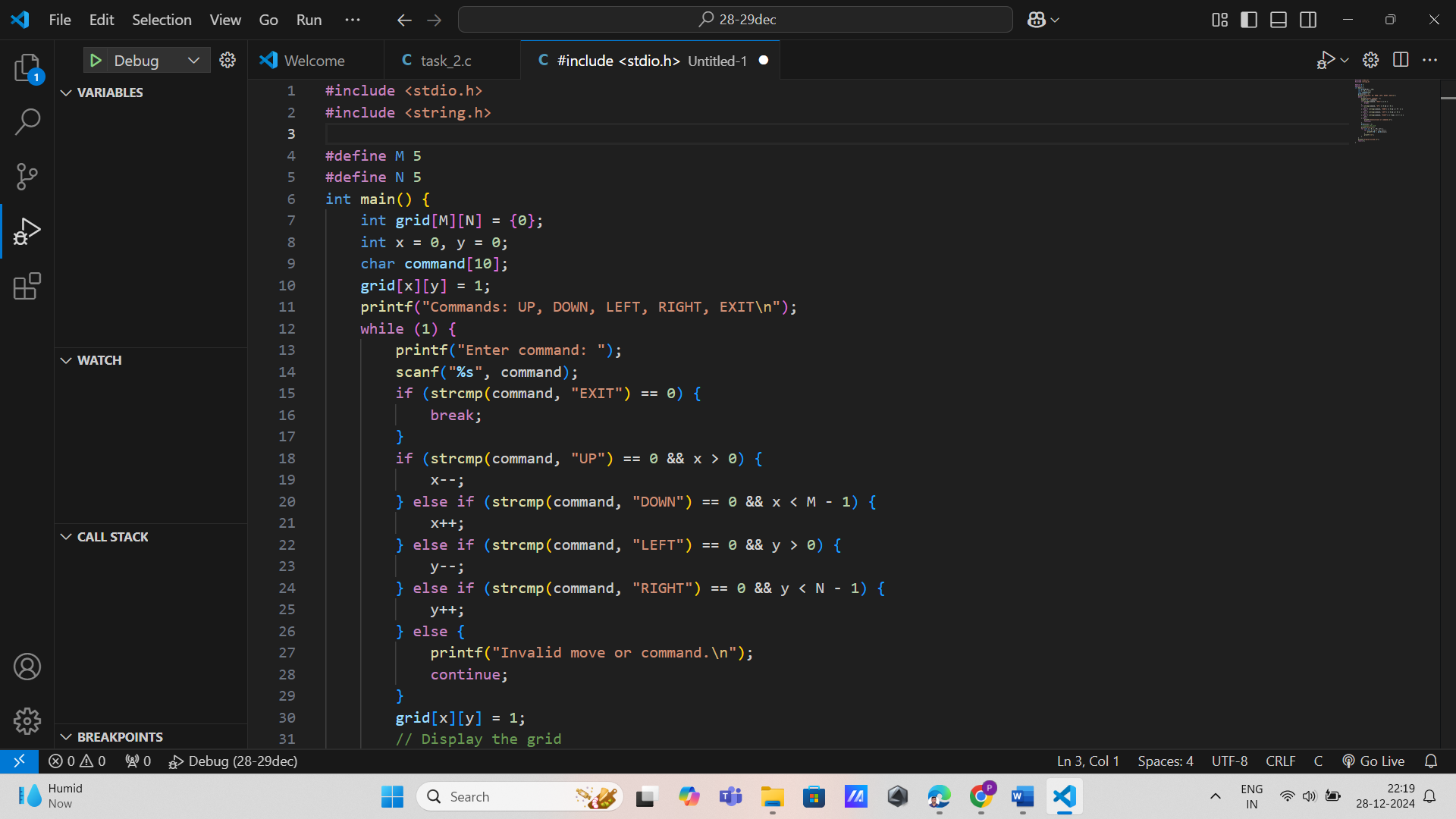
\* Requirements:

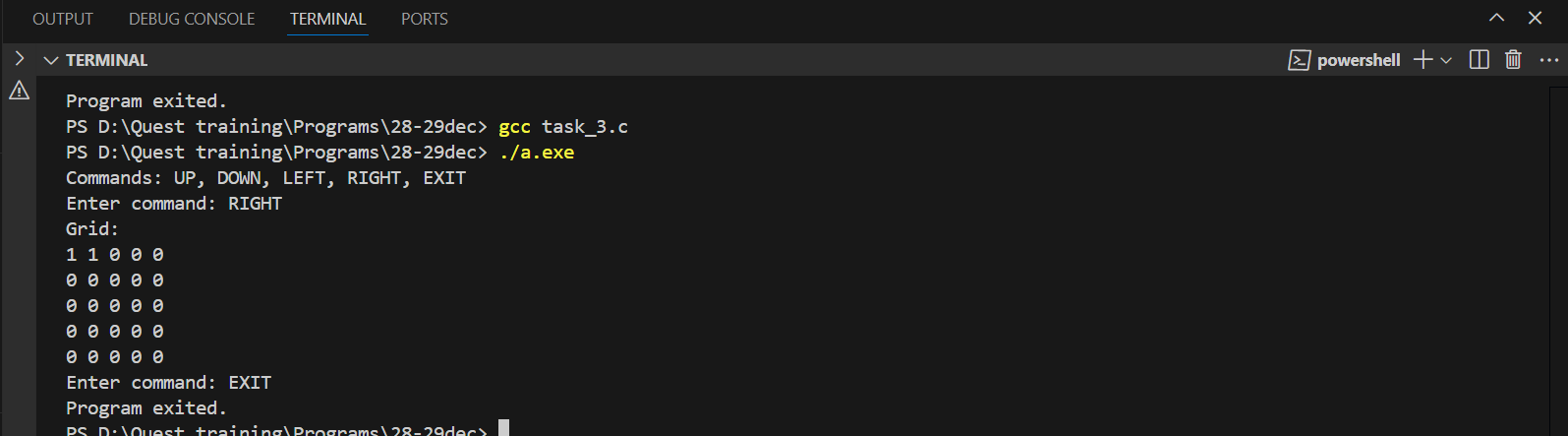
\* Use a 2D array to store visited positions (1 for visited, 0 otherwise).

\* Declare grid dimensions using const variables.

\* Use a while loop to update the robot’s position based on input directions (e.g., UP, DOWN, LEFT, RIGHT).

\* Use if statements to ensure the robot stays within bounds.





4. Sensor Data Aggregation (3D Array)

\* Problem Statement: Store and analyze data from multiple sensors placed in a 3D grid (e.g., environmental sensors in a greenhouse).

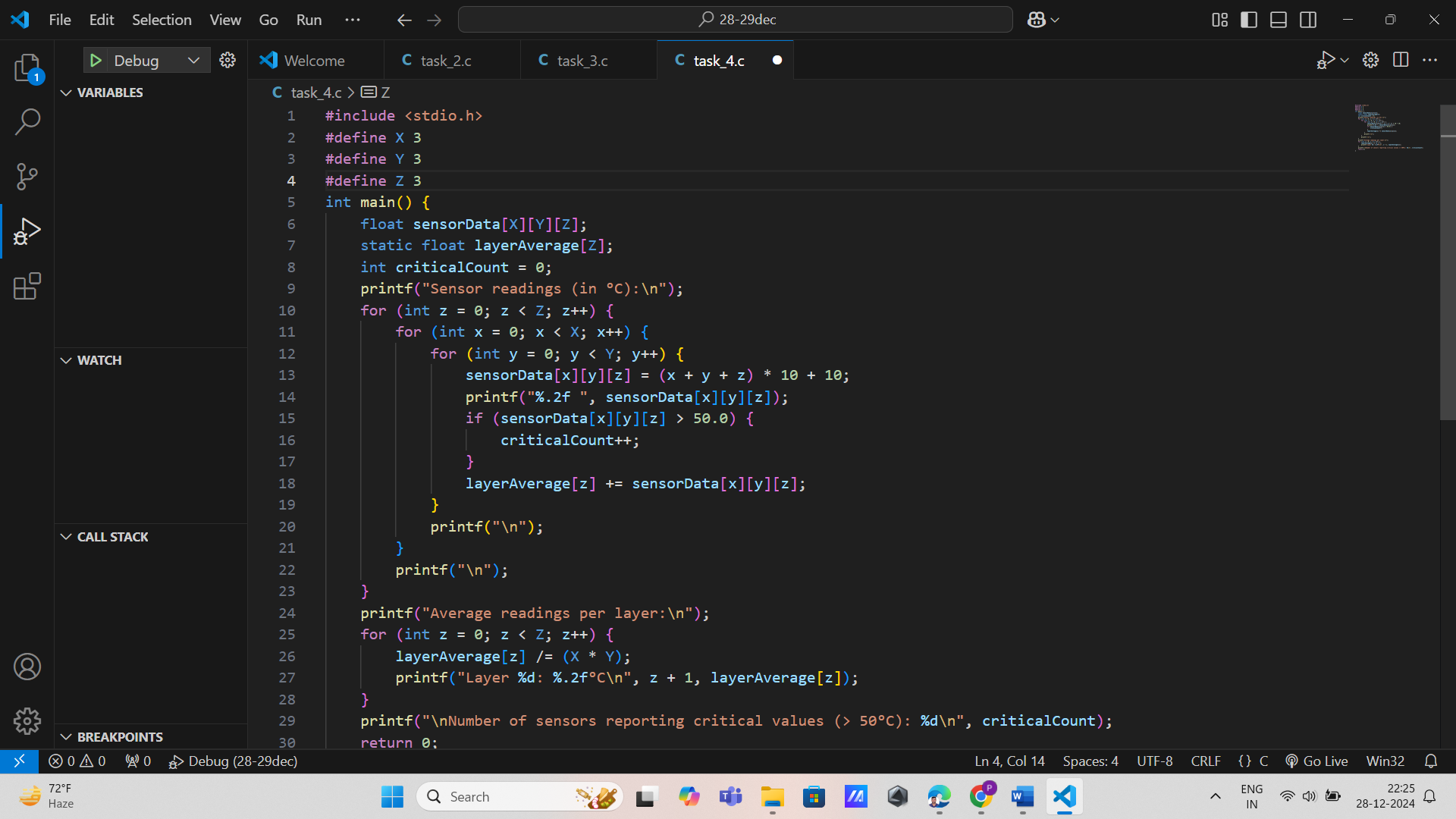
\* Requirements:

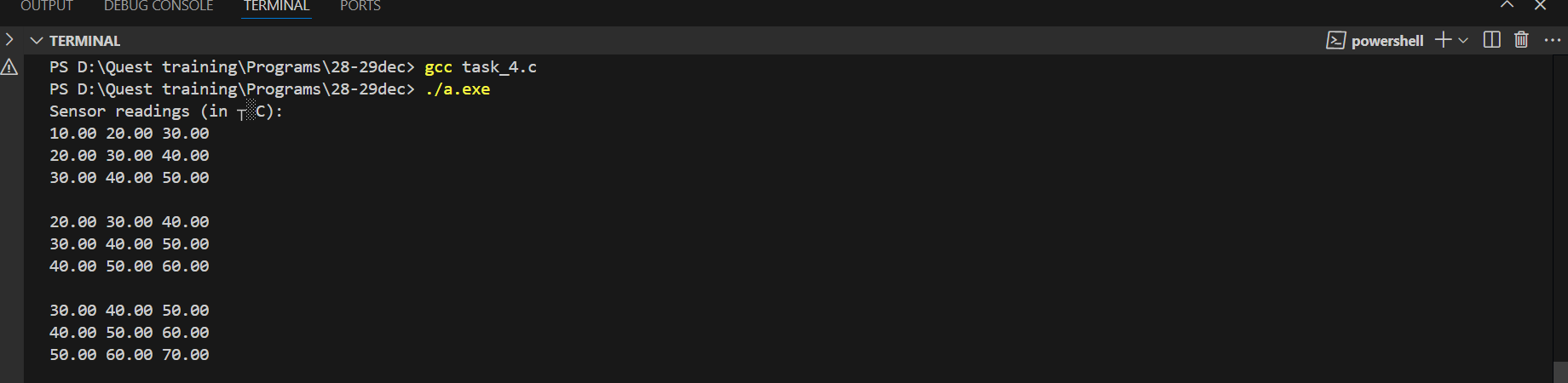
\* Use a 3D array of size [X][Y][Z] to store data, where dimensions are defined using const variables.

\* Use nested for loops to populate the array with sensor readings.

\* Use if statements to find and count sensors reporting critical values (e.g., temperature > 50°C).

\* Use static variables to store aggregated results (e.g., average readings per layer).





5. Image Processing (2D Array)

\* Problem Statement: Perform edge detection on a grayscale image represented as a 2D array.

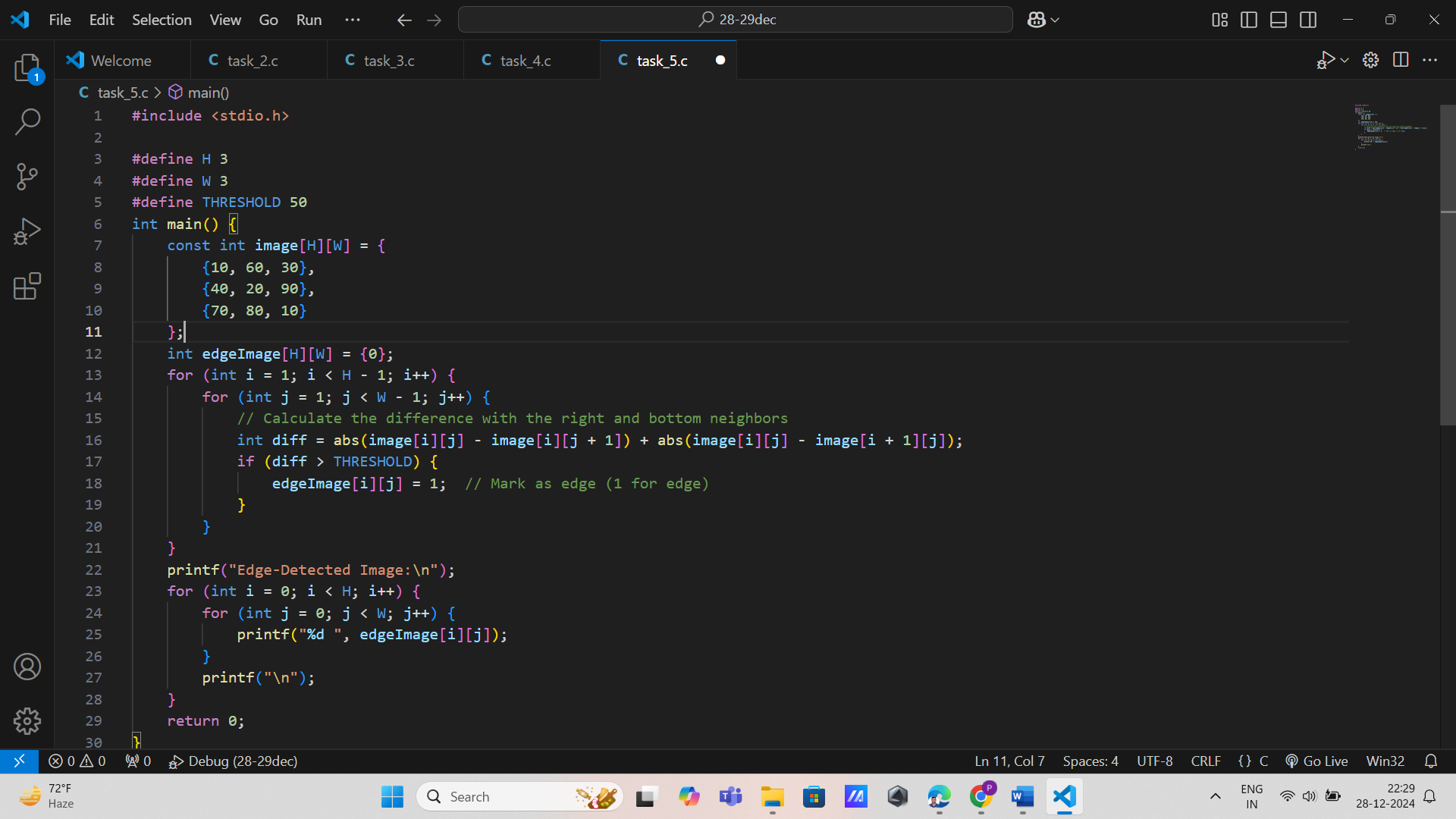
\* Requirements:

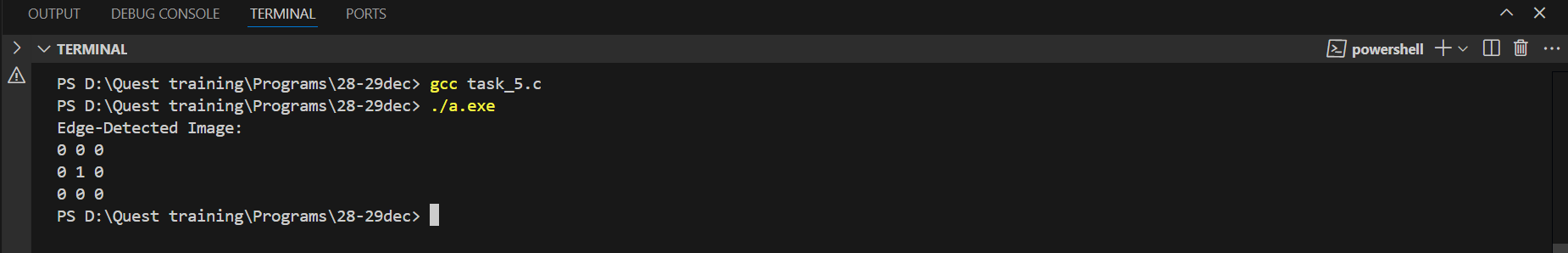
\* Use a 2D array of size [H][W] to store pixel intensity values (defined using const variables).

\* Use nested for loops to apply a basic filter (e.g., Sobel filter) on the matrix.

\* Use decision-making statements to identify and highlight edge pixels (threshold-based).

\* Store the output image in a static 2D array.





6. Traffic Light Controller (State Management with 2D Array)

\* Problem Statement: Manage the states of traffic lights at an intersection with four roads, each having three lights (red, yellow, green).

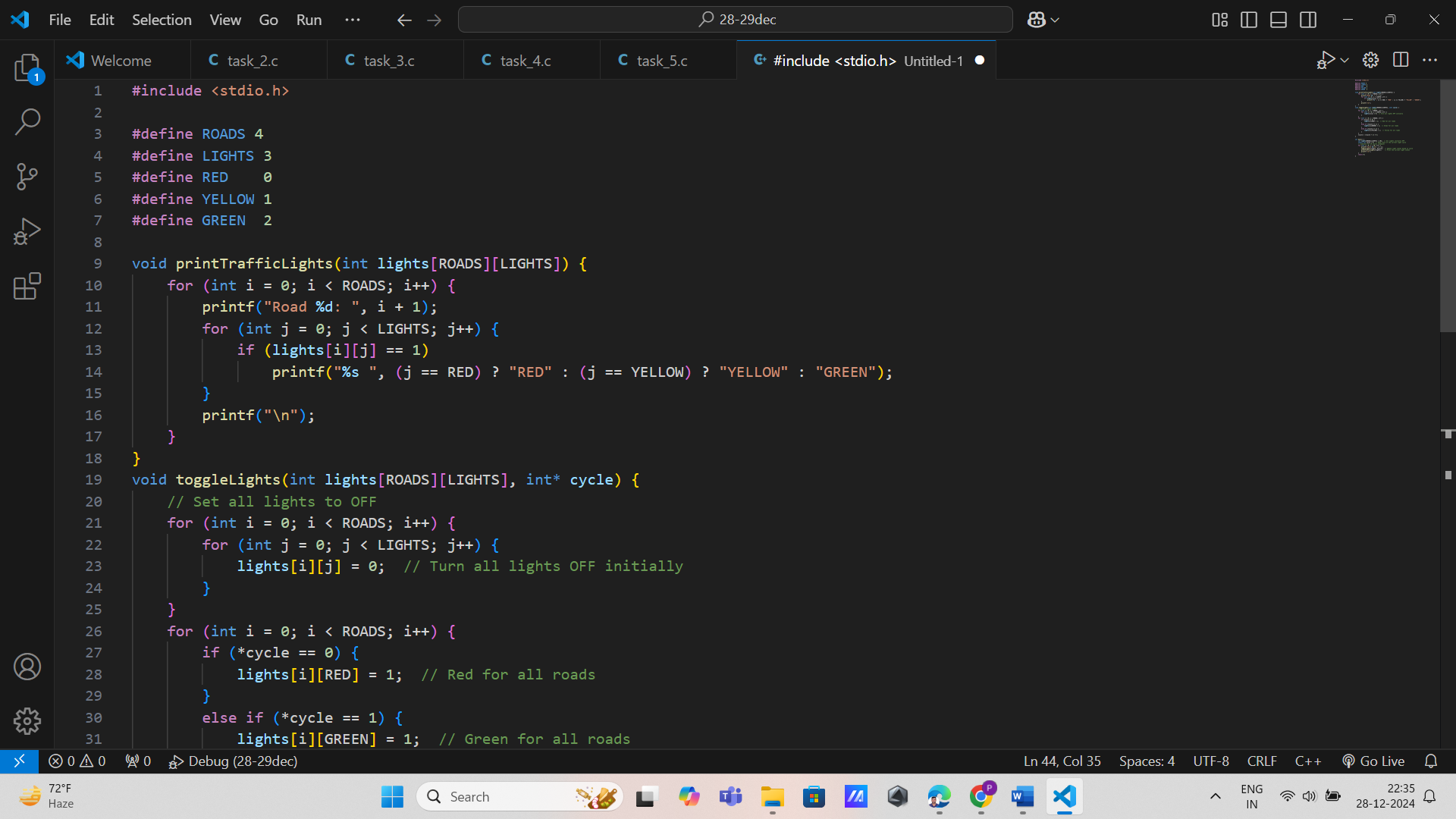
\* Requirements:

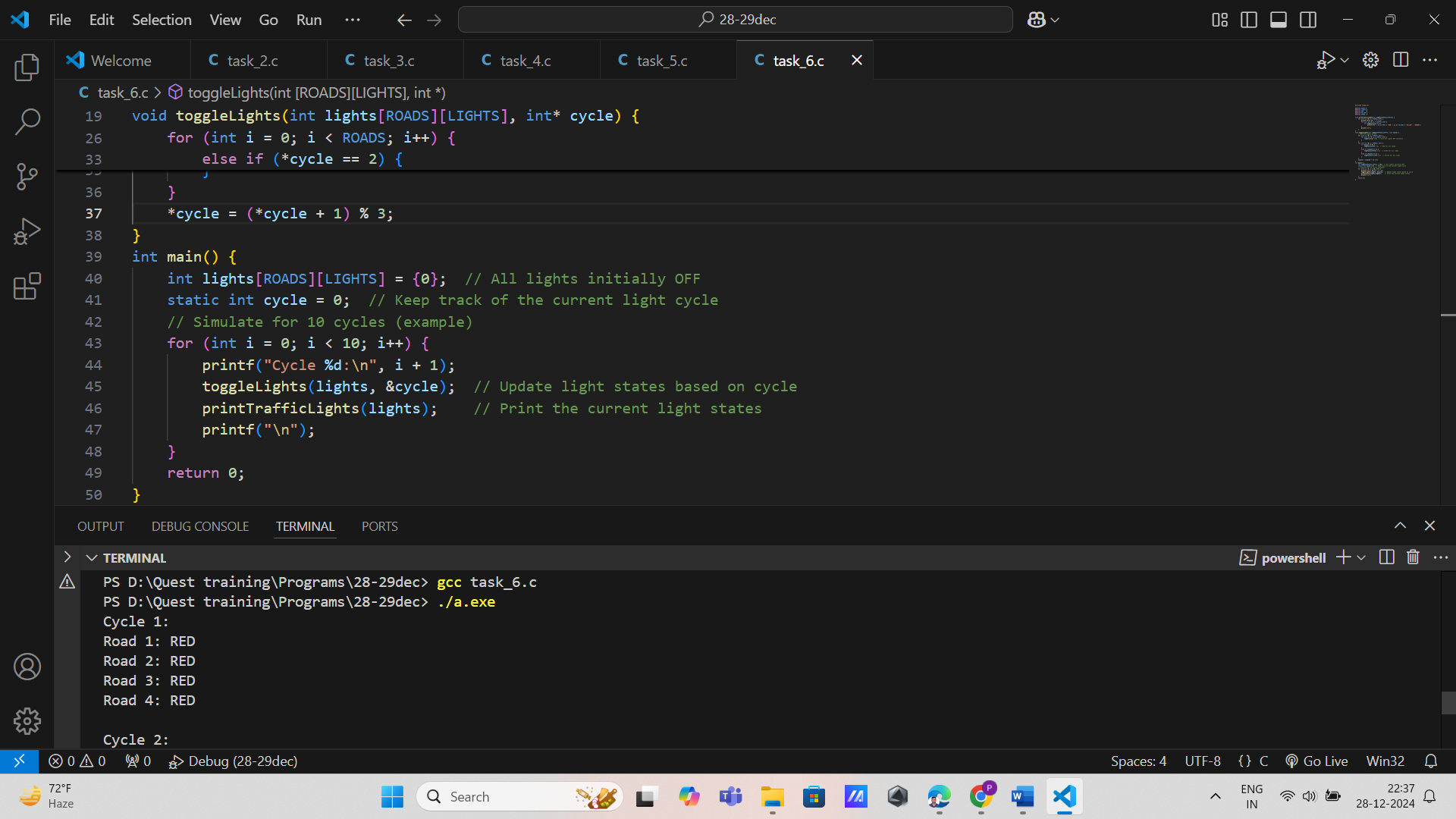
\* Use a 2D array of size [4][3] to store the state of each light (1 for ON, 0 for OFF).

\* Use nested for loops to toggle light states based on time intervals.

\* Use static variables to keep track of the current state cycle.

\* Use if statements to validate light transitions (e.g., green should not overlap with red).





7. 3D LED Cube Animation (3D Array)

\* Problem Statement: Simulate an animation on an LED cube of size 4x4x4.

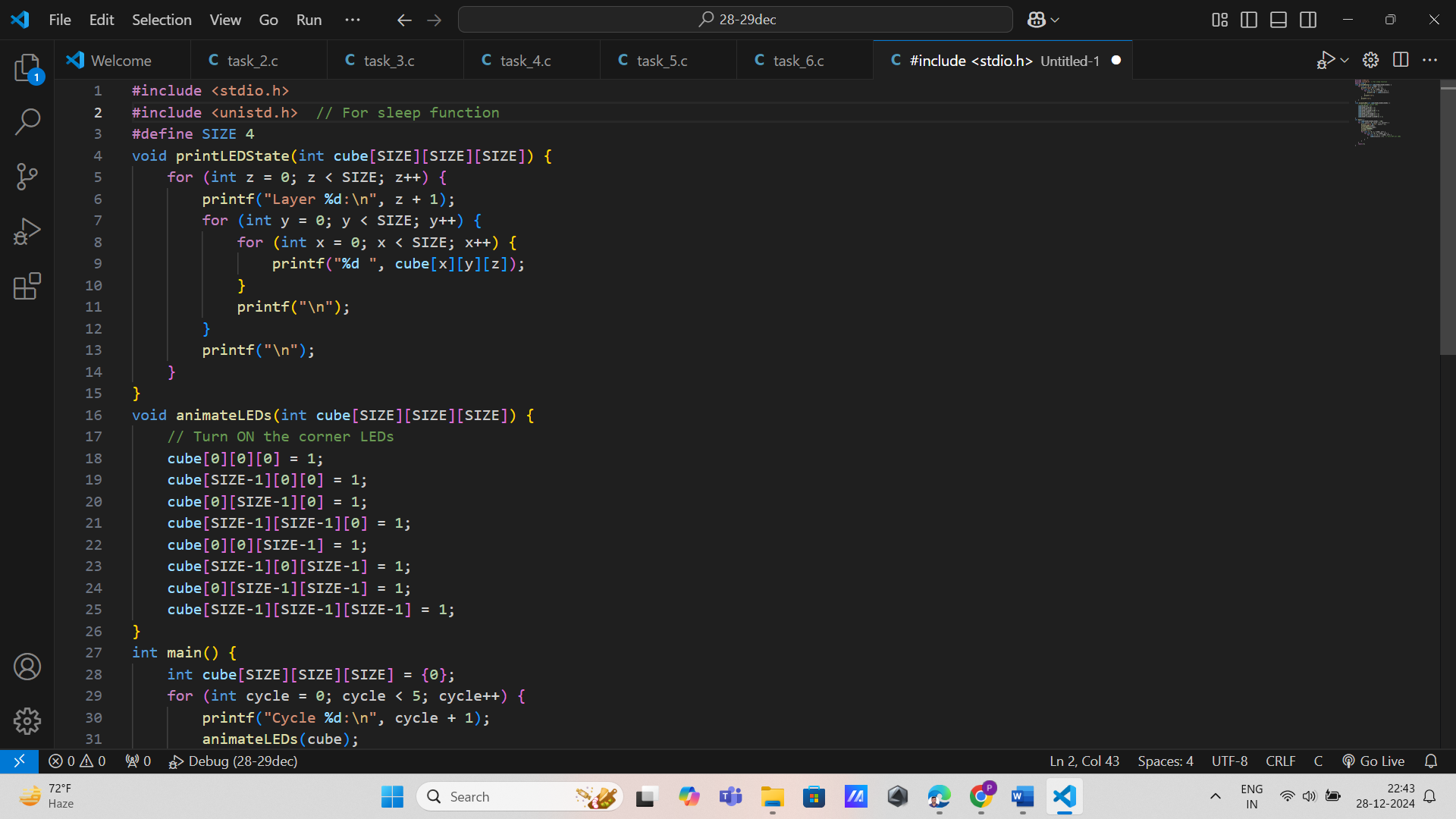
\* Requirements:

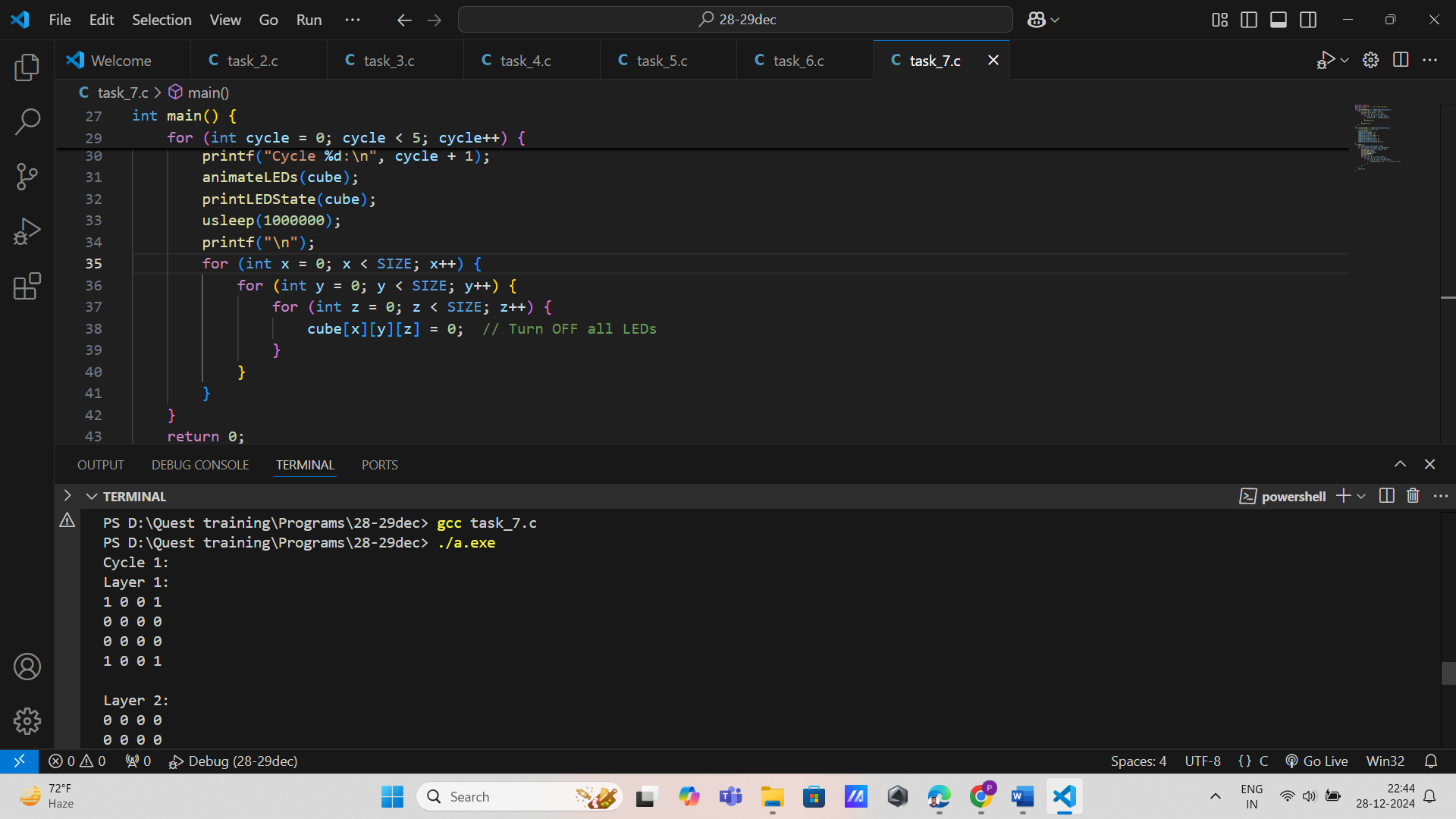
\* Use a 3D array to represent the LED cube's state.

\* Use nested for loops to turn ON/OFF LEDs in a predefined pattern.

\* Use static variables to store animation progress and frame counters.

\* Use if-else statements to create transitions between animation frames.





8. Warehouse Inventory Tracking (3D Array)

\* Problem Statement: Track inventory levels for multiple products stored in a 3D warehouse (e.g., rows, columns, and levels).

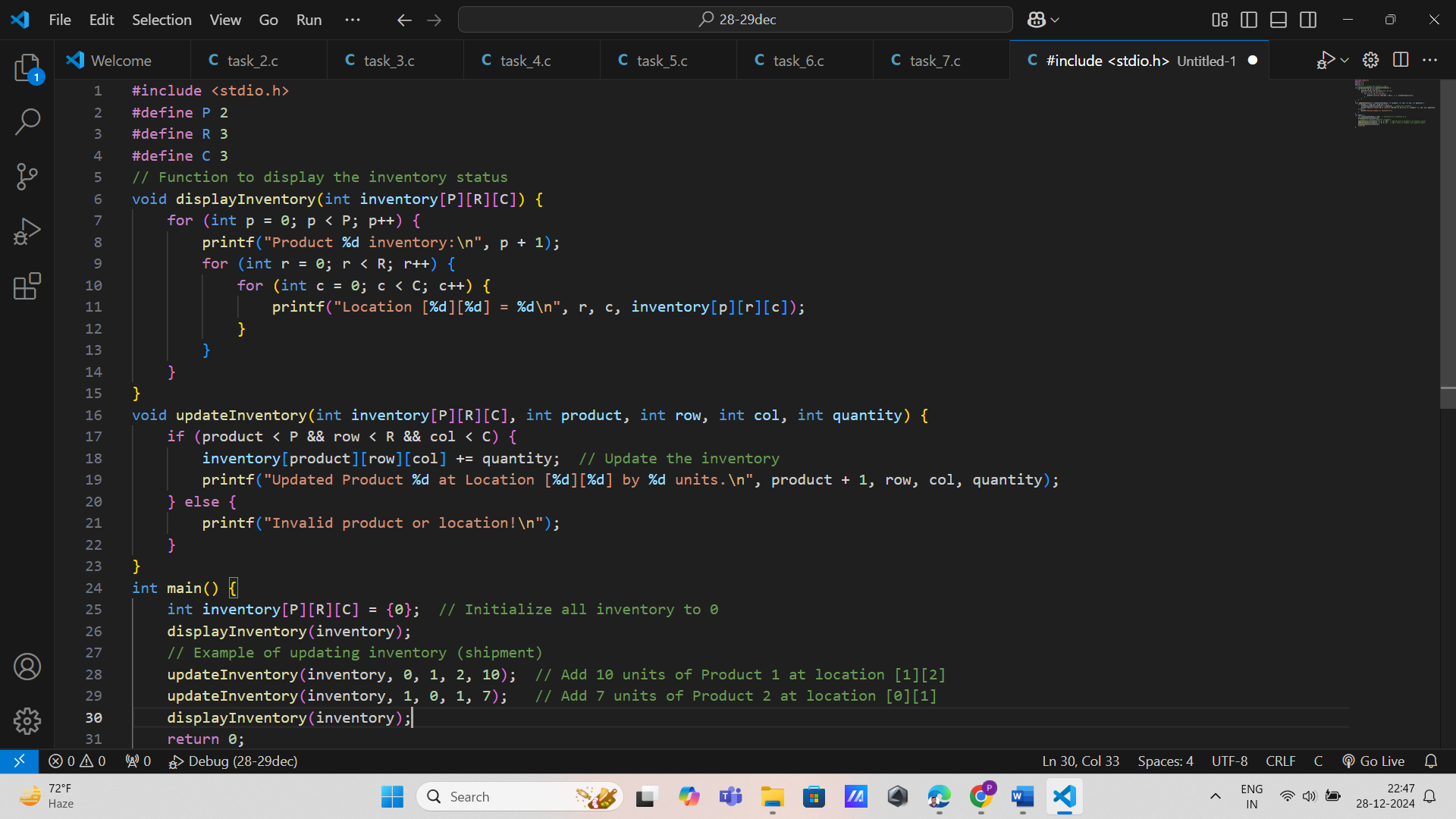
\* Requirements:

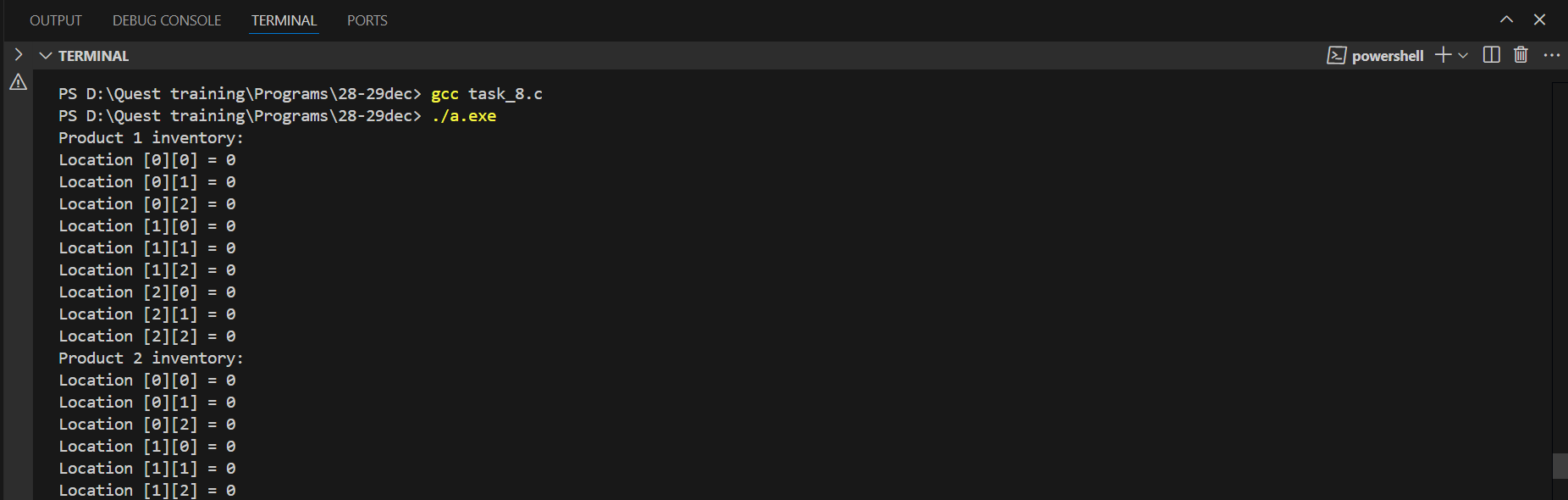
\* Use a 3D array of size [P][R][C] to represent the inventory of P products in a grid

\* Use nested for loops to update inventory levels based on shipments.

\* Use if statements to detect low-stock levels in any location.

\* Use a static variable to store total inventory counts for each product.





9. Signal Processing on a 3D Matrix

\* Problem Statement: Apply a basic signal filter to a 3D matrix representing sampled signals over time.

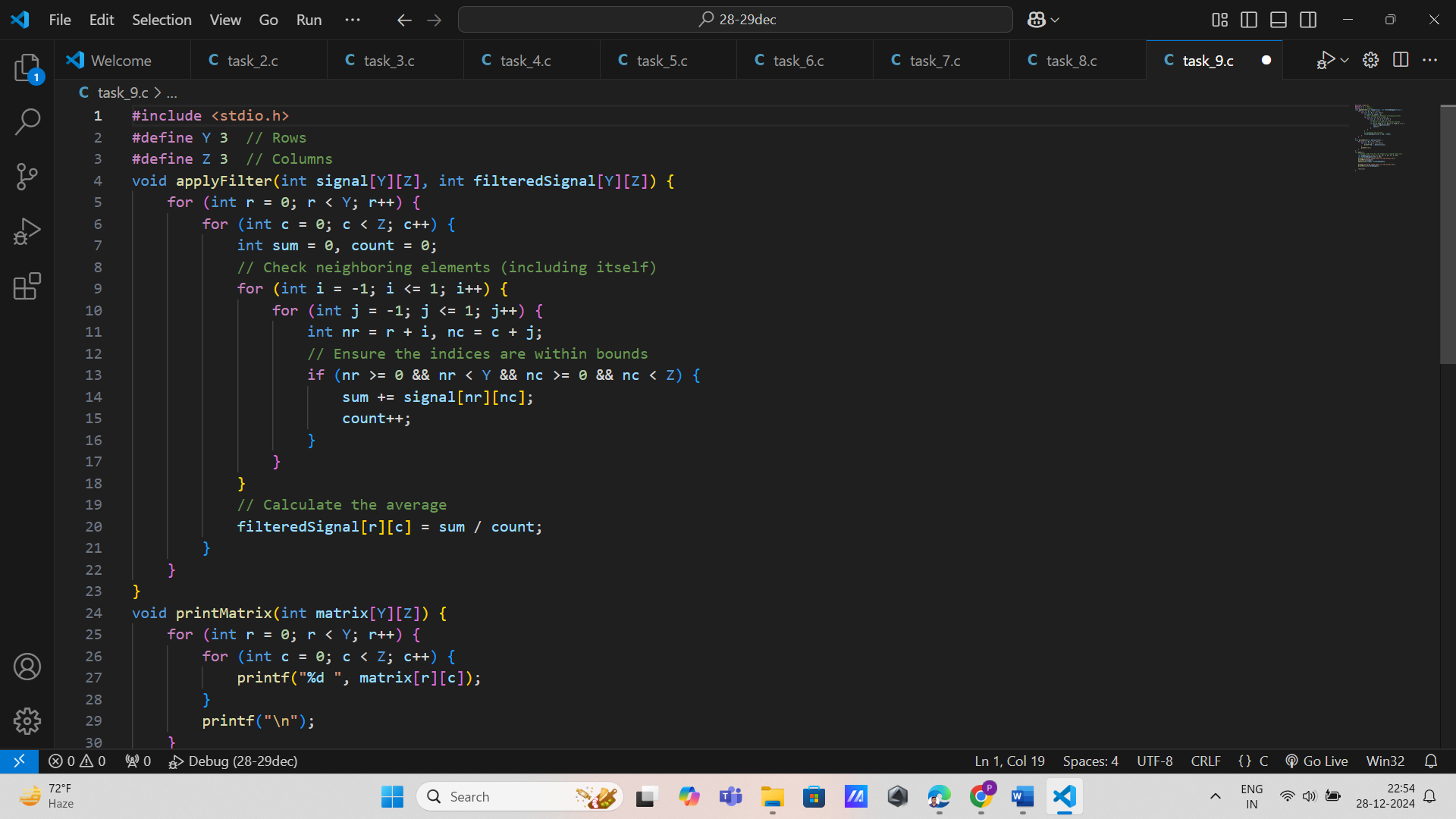
\* Requirements:

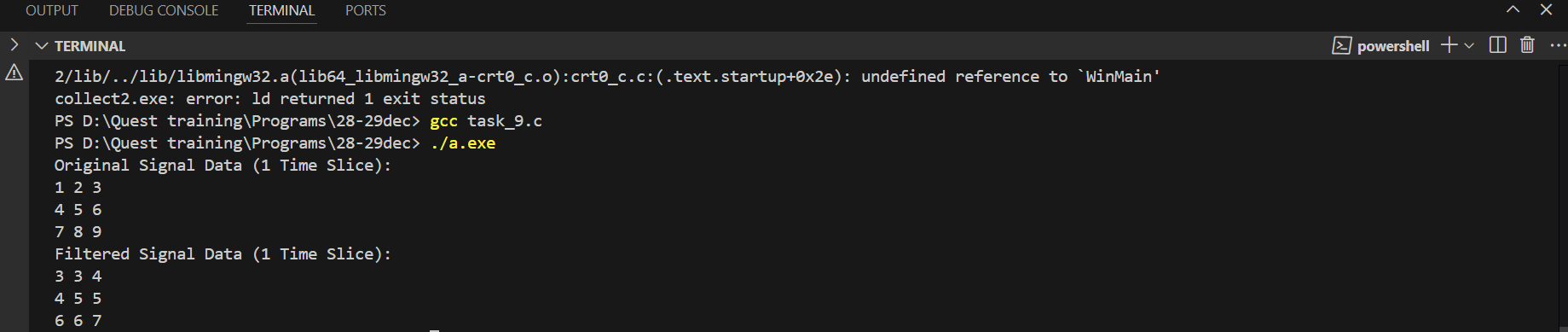
\* Use a 3D array of size [X][Y][Z] to store signal data.

\* Use nested for loops to apply a filter that smoothens the signal values.

\* Use if statements to handle boundary conditions while processing the matrix.

\* Store the filtered results in a static 3D array.





10. Weather Data Analysis (3D Array)

\* Problem Statement: Analyze weather data recorded over multiple locations and days, with hourly samples for each day.

\* Requirements:

\* Use a 3D array of size [D][L][H] to store temperature readings (D days, L locations, H hours per day).

\* Use nested for loops to calculate the average daily temperature for each location.

\* Use if statements to find the location and day with the highest temperature.

\* Use static variables to store results for each location.

