Real-Time Temperature Monitoring Using Arrays

Overview

This program simulates a real-time temperature monitoring system that continuously records temperature readings using an array (list). It generates alerts if the temperature exceeds a predefined threshold.

Features

- **Real-time temperature simulation**: Random temperature readings are generated.
- **Fixed-size array storage**: Stores only the last 10 temperature readings.
- **Automatic data update**: Old readings are removed when new data is added.
- **Threshold-based alerts**: Generates an alert when the temperature exceeds 75°F.
- **Simulated real-time delay**: Data updates every 2 seconds.

Dependencies

This program requires Python's built-in modules:

- time for simulating real-time behavior.
- random for generating temperature readings.

Implementation

Constants

- MAX_READINGS = 10: Defines the maximum number of recent temperature readings stored.
- TEMP_THRESHOLD = 75: Sets the temperature threshold for generating alerts.

Functions

get_temperature()

Generates a random temperature reading between 60°F and 90°F.

add_temperature(temp)

Adds a new temperature reading to the list:

- If the list reaches MAX_READINGS, the oldest reading is removed before adding the new one.

check_alert(temp)

Checks if the new temperature exceeds TEMP_THRESHOLD and prints an alert message if it does.

Main Loop

The program continuously:

1. Generates a new temperature reading.
2. Updates the array with the latest reading.
3. Checks if the reading crosses the threshold and raises an alert if necessary.
4. Displays the latest stored readings.
5. Waits for 2 seconds before repeating the cycle.

Code Example

```
import time
import random

# Constants
MAX_READINGS = 10  # Array size to store recent readings
TEMP_THRESHOLD = 75  # Alert threshold temperature

# Initialize an empty list to store temperature readings
temperature_readings = []

def get_temperature():
    """Simulate getting a real-time temperature reading."""
    return random.randint(60, 90)  # Simulated temperature range

def add_temperature(temp):
```

```
    """Add a new temperature reading to the list, maintaining the size limit."""
    if len(temperature_readings) >= MAX_READINGS:
        temperature_readings.pop(0)  # Remove the oldest reading
    temperature_readings.append(temp)

def check_alert(temp):
    """Check if the temperature crosses the threshold."""
    if temp > TEMP_THRESHOLD:
        print(f"⚠ ALERT! High temperature detected: {temp}°F")

# Real-time monitoring loop (runs indefinitely, but can be stopped manually)
while True:
    current_temp = get_temperature()  # Simulate real-time data collection
    add_temperature(current_temp)  # Store the latest temperature
    check_alert(current_temp)  # Check for alerts

    print(f"Latest Temperatures: {temperature_readings}")  # Display stored readings

    time.sleep(2)  # Simulate a delay in real-time monitoring
```

Usage

- Run the script in a Python environment.
- The program will display the last 10 temperature readings and raise alerts when necessary.
- Stop the script manually (Ctrl+C) when needed.

Future Enhancements

- Store temperature data in a database.
- Add real sensor integration.

- Implement a graphical dashboard for real-time visualization.

Conclusion

This program effectively demonstrates the use of arrays (lists) in real-time data processing, with applications in sensor monitoring, IoT, and automation systems.