**Write the Pseudocode and Flowchart for the problem statements mentioned below:**

1. Smart Home Temperature Control

Problem Statement:

Design a temperature control system for a smart home. The system should read the current temperature from a sensor every minute and compare it to a user-defined setpoint.

Requirements:

• If the current temperature is above the setpoint, activate the cooling system.

• If the current temperature is below the setpoint, activate the heating system.

• Display the current temperature and setpoint on an LCD screen.

• Include error handling for sensor failures.

**Pseudocode:**

setpoint = n

Loop every minute

Read temperature temp

If (sensor != 0)

If (temp > setpoint)

cooling\_system = 1

Else

heating\_system = 1

End if

Else

Write: sensor failure

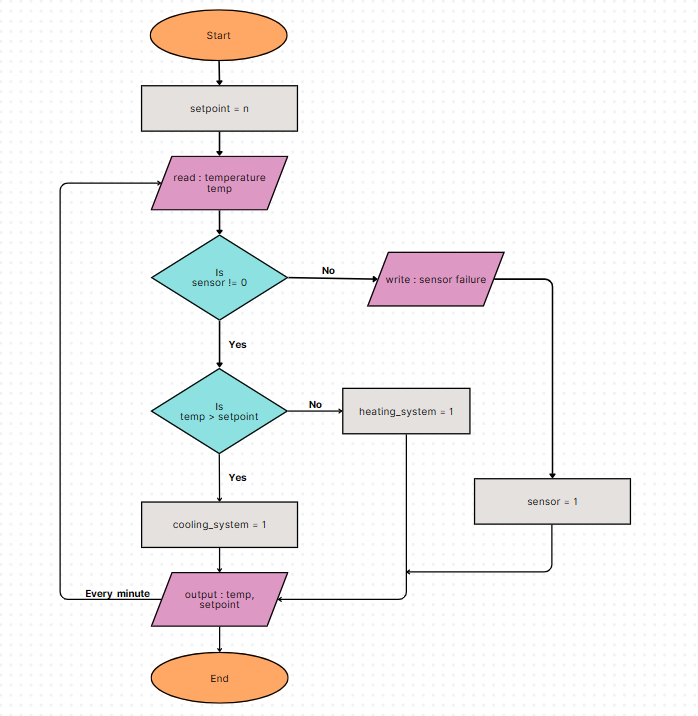
sensor = 1

End if

Return temp, setpoint

End loop

**Flowchart:**



2. Automated Plant Watering System

Problem Statement:

Create an automated watering system for plants that checks soil moisture levels and waters the plants accordingly.

Requirements:

• Read soil moisture level from a sensor every hour.

• If moisture level is below a defined threshold, activate the water pump for a specified duration.

• Log the watering events with timestamps to an SD card.

• Provide feedback through an LED indicator (e.g., LED ON when watering).

**Pseudocode:**

threshold = n

duration = t

Loop every hour

Read moisture m

If (m < threshold)

while (time <= duration)

pump\_state = 1

led = 1

Else

pump\_state = 0

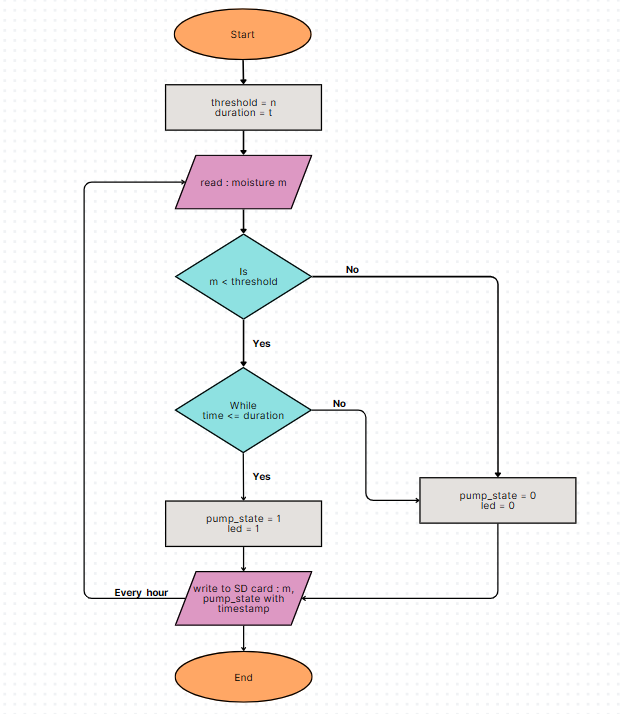
led = 0

End if

Log m, pump\_state with timestamp to SD card

End loop

**Flowchart:**



3. Motion Detection Alarm System

Problem Statement:

Develop a security alarm system that detects motion using a PIR sensor.

Requirements:

• Continuously monitor motion detection status.

• If motion is detected for more than 5 seconds, trigger an alarm (buzzer).

• Send a notification to a mobile device via UART communication.

• Include a reset mechanism to deactivate the alarm.

**Pseudocode:**

motion\_timer = 0

Loop

Read motion m

while (m == 1)

Increment motion\_timer

If (motion\_timer > 5)

buzzer = 1

Send notification to mobile

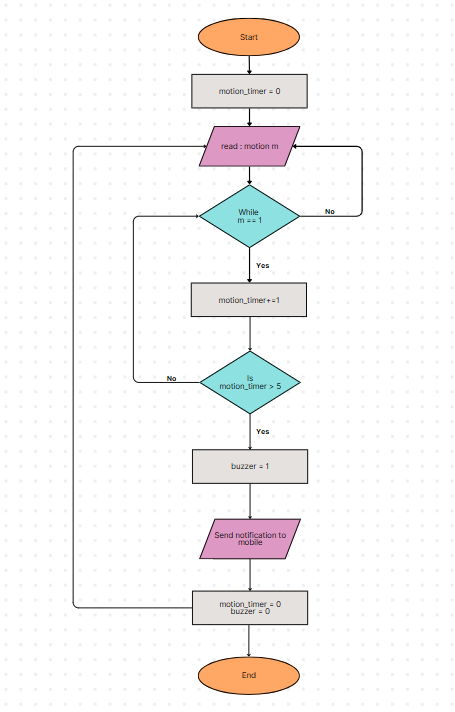
motion\_timer = 0

buzzer = 0

End if

End loop

**Flowchart:**



4. Heart Rate Monitor

Problem Statement:

Implement a heart rate monitoring application that reads data from a heart rate sensor.

Requirements:

• Sample heart rate data every second and calculate the average heart rate over one minute.

• If the heart rate exceeds 100 beats per minute, trigger an alert (buzzer).

• Display current heart rate and average heart rate on an LCD screen.

• Log heart rate data to an SD card for later analysis.

**Pseudocode:**

Loop every minute

total = 0

average = 0

Loop every second

Read heart\_rate r

total += r

End loop

average = total/60

If (average > 100)

buzzer = 1

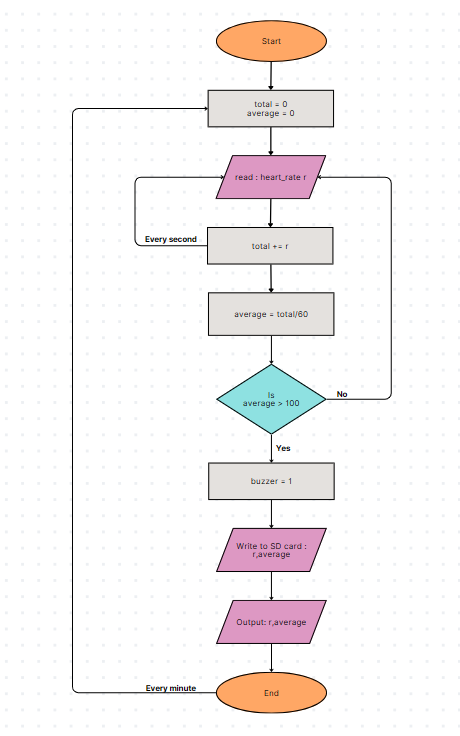
Log r, average to SD card

End if

Return r, average

End loop

**Flowchart:**



5. LED Control Based on Light Sensor

Problem Statement:

Create an embedded application that controls an LED based on ambient light levels detected by a light sensor.

Requirements:

• Read light intensity from the sensor every minute.

• If light intensity is below a certain threshold, turn ON the LED; otherwise, turn it OFF.

• Include a manual override switch that allows users to control the LED regardless of sensor input.

• Provide status feedback through another LED (e.g., blinking when in manual mode).

**Pseudocode:**

threshold = n

Loop every minute

Read light\_intensity i

If (mode != manual)

If (i < threshold)

led\_1 = 1

Else

led\_1 = 0

End if

Else

led\_2 = blinking

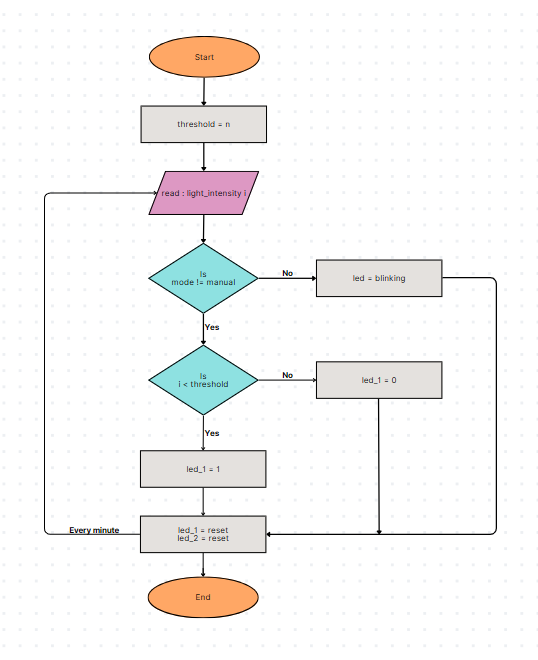
End if

led\_1 = reset

led\_2 = reset

End loop

**Flowchart:**



6. Digital Stopwatch

Problem Statement:

Design a digital stopwatch application that can start, stop, and reset using button inputs.

Requirements:

• Use buttons for Start, Stop, and Reset functionalities.

• Display elapsed time on an LCD screen in hours, minutes, and seconds format.

• Include functionality to pause and resume timing without resetting.

• Log start and stop times to an SD card when stopped.

**Pseudocode:**

Initialize start, stop, reset, elapsed\_time = 0

Initialize start\_time, stop\_time

Loop

If (start == 1)

Read start\_time

If (stop == 0)

Elapsed\_time = start\_time – stop\_time

Display elapsed\_time in hr:min:s

End if

If (stop == 1)

Read stop\_time

Elapsed\_time = start\_time – stop\_time

Display elapsed\_time in hr:min:s

End if

End if

log start\_time, stop\_time to SD card

Display elapsed\_time hr:min:sec

If (reset == 1)

elapsed\_time = 0

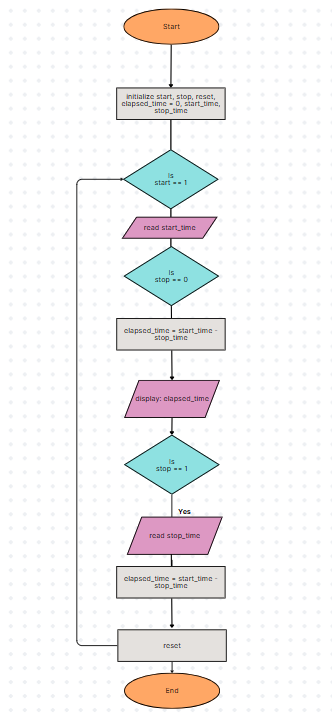
start\_time = 0

stop\_time = 0

End if

End loop

**Flowchart:**



7. Temperature Logging System

Problem Statement:

Implement a temperature logging system that records temperature data at regular intervals.

Requirements:

• Read temperature from a sensor every 10 minutes.

• Store each reading along with its timestamp in an array or log file.

• Provide functionality to retrieve and display historical data upon request.

• Include error handling for sensor read failures.

**Pseudocode:**

Loop every 10 minutes

Read temperature t

Store t, time\_stamp

If (sensor != 0)

If (retrieve == 1)

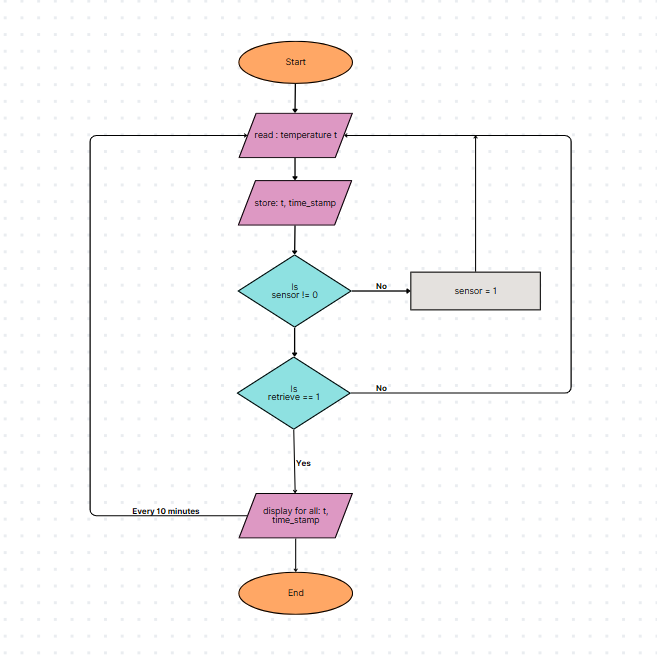
Display for all t, time\_stamp

Else

sensor = 1

End loop

**Pseudocode:**



8. Bluetooth Controlled Robot

Problem Statement:

Create an embedded application for controlling a robot via Bluetooth commands.

Requirements:

• Establish Bluetooth communication with a mobile device.

• Implement commands for moving forward, backward, left, and right.

• Include speed control functionality based on received commands.

• Provide feedback through LEDs indicating the current state (e.g., moving or stopped).

**Pseudocode:**

set connection = 1

If (moving == 1)

Read speed

Led = 1

If (action == forward)

Move forward

If (action == backward)

Move backward

If (action == left)

Move left

If (action == right)

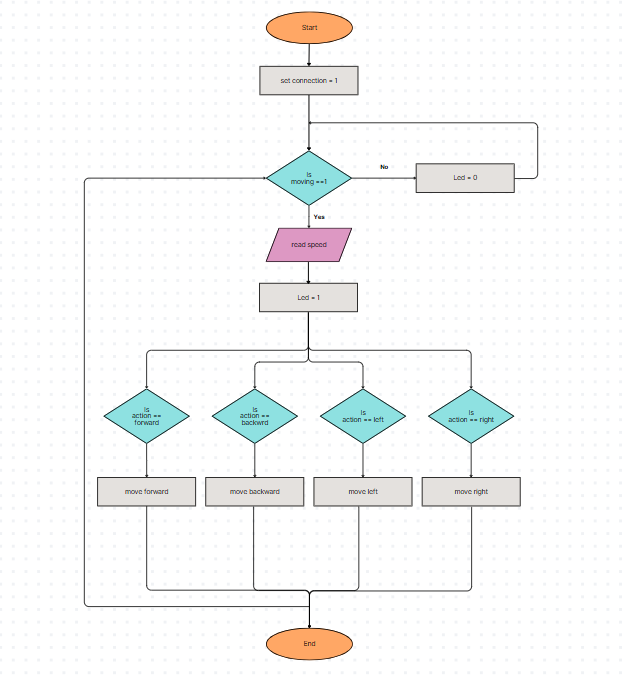
Move right

Else

Led = 0

End if

**Flowchart:**



9. Battery Monitoring System

Problem Statement:

Develop a battery monitoring system that checks battery voltage levels periodically and alerts if voltage drops below a safe threshold.

Requirements:

• Measure battery voltage every minute using an ADC (Analog-to-Digital Converter).

• If voltage falls below 11V, trigger an alert (buzzer) and log the event to memory.

• Display current voltage on an LCD screen continuously.

• Implement power-saving features to reduce energy consumption during idle periods.

**Pseudocode:**

Loop every minute

Read voltage v

If (v != 0)

If (v < 11)

buzzer = 1

log event to memory

End if

Display v

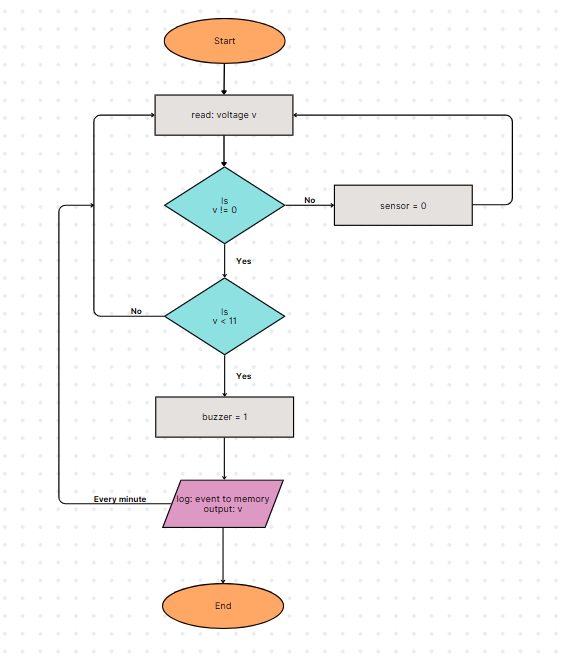
Else

sensor = 0

End if

End loop

**Flowchart:**



10. RFID-Based Access Control System

Problem Statement:

Design an access control system using RFID technology to grant or deny access based on scanned RFID tags.

Requirements:

• Continuously monitor for RFID tag scans using an RFID reader.

• Compare scanned tags against an authorized list stored in memory.

• Grant access by activating a relay if the tag is authorized; otherwise, deny access with an alert (buzzer).

• Log access attempts (successful and unsuccessful) with timestamps to an SD card.

**Pseudocode:**

Initialize authorized\_list

Loop

Read RFID\_tag

If (RFID\_tag in authorized\_list)

relay = 1

Else

buzzer = 1

End if

Log event with time\_stamp to SD card

End loop

**Flowchart:**

