ALGORITHM AND PSEUDOCODE

Problem Statement 1: Temperature Monitoring System

Objective: Design a temperature monitoring system that reads temperature data from a sensor and triggers an alarm if the temperature exceeds a predefined threshold.

Requirements:

- Read temperature data from a temperature sensor at regular intervals.
- Compare the read temperature with a predefined threshold.
- If the temperature exceeds the threshold, activate an alarm (e.g., LED or buzzer).
- Include functionality to reset the alarm.

Answer:

```
1. start
2. declare a temporary variable, temp, to activate and reset alarm
3. set a threshold value
4. read temperature value
5. if temperature is greater than threshold,
   set temp=1 to activate alarm
   set temp=0 to reset the alarm
6. repeat the steps
7. end
```

Problem Statement 2: Motor Control System

Objective: Implement a motor control system that adjusts the speed of a DC motor based on user input.

Requirements:

- Use a potentiometer to read user input for desired motor speed.
- Control the motor speed using PWM (Pulse Width Modulation).
- Display the current speed on an LCD.

Answer:

```
1. start
2. read user input for desired speed
```

```
3. if input=current speed, then exit the loop
4. else, use PWM to adjust the speed
   calculate current speed
   Display current speed
5. End
```

Problem Statement 3: LED Blinking Pattern

Objective: Create an embedded system that controls an array of LEDs to blink in a specific pattern based on user-defined settings.

Requirements:

- Allow users to define blink patterns (e.g., fast, slow).
- Implement different patterns using timers and interrupts.
- Provide feedback through an LCD or serial monitor.

Answer:

```
1. Start
2. set timers and interrupts
3. read user input for blink patterns
4. implement pattern based on user input
5. output feedback
6. end
```

Problem Statement 4: Data Logger

Objective: Develop a data logger that collects sensor data over time and stores it in non-volatile memory.

Requirements:

- Read data from sensors (e.g., temperature, humidity) at specified intervals.
- Store collected data in EEPROM or flash memory.
- Implement functionality to retrieve and display logged data

Answer:

```
1. start
2. set a timer to read data
3. read the sensor data at specified intervals
```

4. store data in EEPROM
5. display logged data when required
6. repeat the steps
7. end

Simple Calculator

Problem Statement: Write a program that functions as a simple calculator. It should be able to perform addition, subtraction, multiplication, and division based on user input.
Requirements:
1. Prompt the user to enter two numbers.
2. Ask the user to select an operation (addition, subtraction, multiplication, division).
3. Perform the selected operation and display the result.
4. Handle division by zero appropriately.

Answer:

```
enter first number, a
enter second number, b
declare res to store result
switch case to select operation(+,-,*,/)
1. for '+': res=a+b
2. for '-': res=a-b
3. for '*': res=a*b
4. for '/': res=a/b
display res
end
```

Factorial Calculation

Problem Statement: Write a program to calculate the factorial of a given non-negative integer.
Requirements:
1. Prompt the user to enter a non-negative integer.
2. Calculate the factorial using a loop.
3. Display the factorial of the number.

Answer:

```
1. Start
2. Read non-negative integer, n
3. Set Factorial, f=1
4. for(i=1;i<=n;i++)
5. f=f*i
```

6. Print f
7. End

Factorial Calculation using recursion

Answer:

```
int i=1,f=1;
Read user input for n
fact(int n)
{
while(i<=n)
f=f*i;
i++;
}
```

Problem Statement: Smart Irrigation System

Objective: Design a smart irrigation system that automatically waters plants based on soil moisture levels and environmental conditions. The system should monitor soil moisture and activate the water pump when the moisture level falls below a predefined threshold.

Requirements:

1. Inputs:

2. Outputs:

3. Conditions:

- The pump should only activate if the soil moisture is below the threshold and it is daytime (e.g., between 6 AM and 6 PM).
- If the soil moisture is adequate, the system should display a message indicating that watering is not needed.
- Activate the water pump when the soil moisture is below the threshold.
- Display the current soil moisture level and whether the pump is activated or not.
- Soil moisture sensor reading (percentage).
- User-defined threshold for soil moisture (percentage).
- Time of day (to prevent watering during rain or at night).

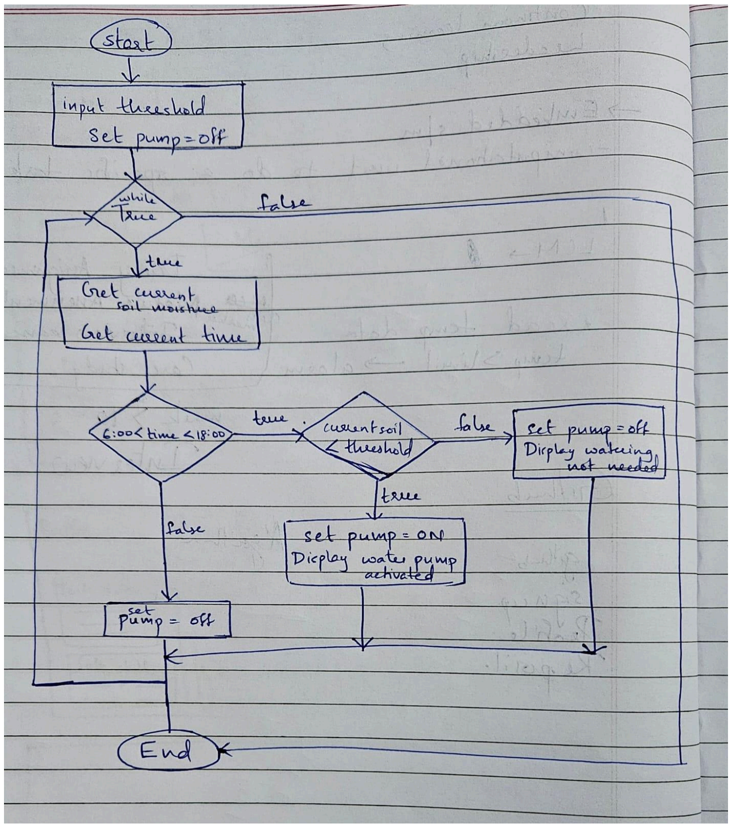
Deliverables:

- Write pseudocode that outlines the algorithm for the smart irrigation system.
- Create a flowchart that visually represents the logic of your pseudocode.

Answer:

Pseudocode:

```
1. Start
2. input the threshold value=threshold
3. Set pump=off
4. loop to monitor soil moisture and time
   Get current soil moisture= currsoil
   Get current time= time
5. If time is between 6.00 && 18.00
        If currsoil<threshold
            Set pump=on
            Display water pump activated
        Else
            Set pump=off
            Display Waterting not needed
   Else
        pump=off
6. End
```


start
input threshold
Set pump=off
while True
false
true
Get current soil moisture
Get current time
6:00< time <18:00
true
current soil < threshold
false
set pump = off
Display watering not needed
true
set pump = ON
Display water pump activated
false
set pump = off
End