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.

ANS::

```
1.int a,b,c
2.Take desired operation Op
3.Switch(Op):
        3.1 Case(+) :
        3.1.1 c=a+b
        3.1.2 print "Sum=%d",c
        3.2 Case(-) :
        3.2.1 c=a-b
        3.2.2 print "Sub=%d",c
        3.3 Case(*) :
        3.3.1 c=a*b
        3.3.2 print "Mul=%d",c
        3.4 Case(/) :
        3.4.1 If b!=0
        3.4.1.1 c=a/b
        3.4.1.2 print "Div=%d",c
        3.4.2 print "Division by 0 not allowed"
        3.5 default :
        3.5.1 print "Enter a valid input"
```

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.

ANS::

```
1.Take the input number n
2.if n>0
        2.1 fact=1
        2.2 for (i=n ; i>=1 ; i--)
        2.2.1 fact *=i;
        2.3 print fact
```

```
3.else
  3.1 print "enter non negative value"
```

Same Factorial using recursion

ANS::

```
1.Take n
2.if n>0
        3.int fact(n)
        3.1 if n==1
        3.1.1 return 1;
        3.2 else
        3.2.1 return n*fact(n-1);
```

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.

ANS::

```
1.Set Threshold Temp
2.Read Temp from the Temp sensor
3.Compare Temp with Threshold Temp
  3.1 if Temp > Threshold Temp
        3.1.1 Activate Alarm
        3.1.2 Reset Alarm
        3.1.1.1 Yes
        3.1.1.2 No
  3.2 else
        3.2.1 Continue looping from Step 2 every 10 seconds
```

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.

ANS::

1.Take input speed from the user using a potentiometer
2.Check the current state of DC motor
  2.1 If required increase or decrease the motor speed using input from PWM
3.Display the speed on LCD
4.Repeat

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.

ANS::
1.Set the LED's
2.Set the timer
3.Take the desired pattern as input from user
4.According to the input adjust the Blinking interval timer
5.Display the Current pattern on LCD
6.Repeat

Problem Statement 5: 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

ANS::

1.Set an Interval for data logging
2.Read the data into variables at the specified interval
3.Store the data in EEPROM or flash memory
4.Display the data when required
5.Repeat

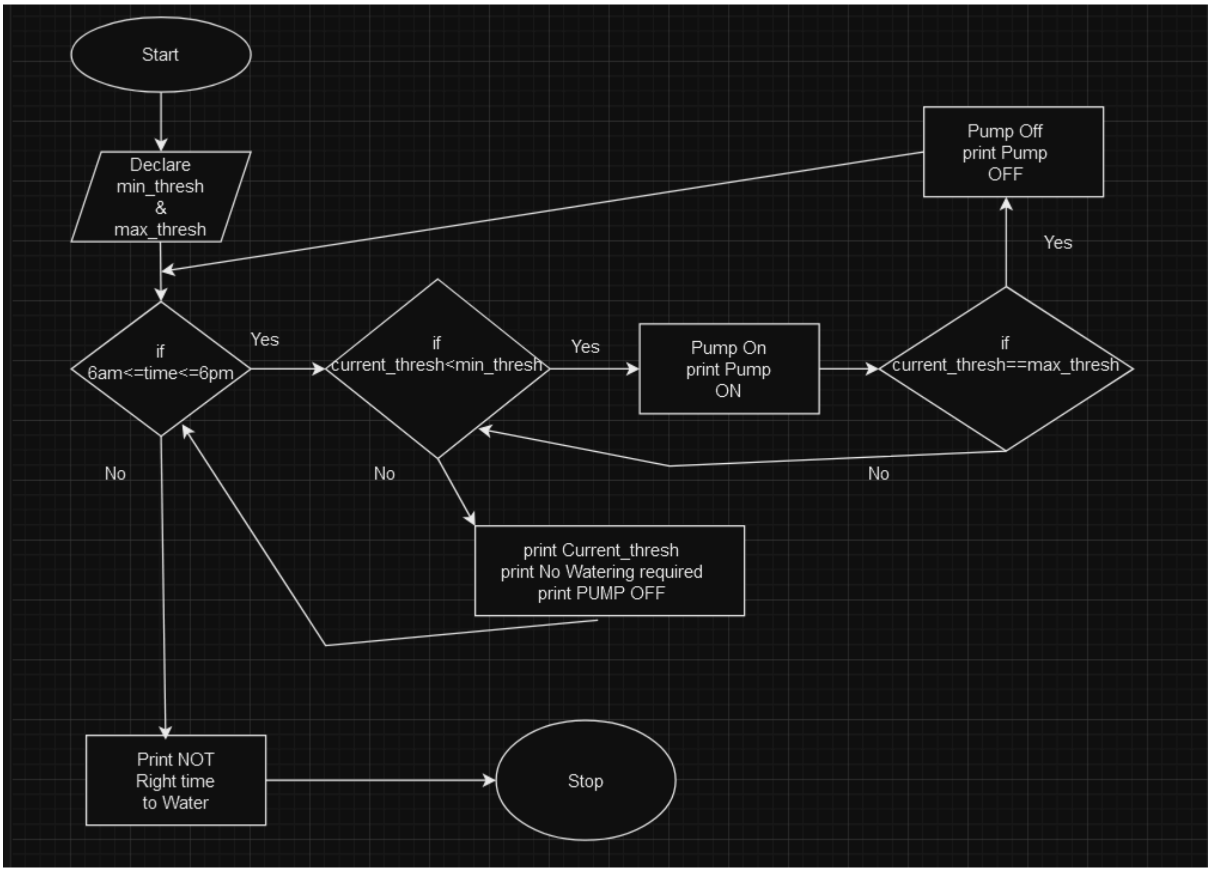
Problem Statement: Smart Irrigation System
Objective: Design a smart irrigation system that automatically waters plants based on soil moisture levels and environmental conditions. The system should monitor soil moisture and activate the water pump when the moisture level falls below a predefined threshold.

Requirements:
Inputs:
Outputs:
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.

ANS::

```
1.Take maximum and minimum threshold from user as max_thresh min_thresh
2.if 6:00AM<=time<=6:00PM
        2.1 if current_thresh < min_thresh
          2.1.1 print current_thresh
          2.1.2 turn pump on
          2.1.3 print "Pump activated"
            2.1.4 if current_thresh == max_thresh
                2.1.4.1 print current_thresh
                2.1.4.2 turn off pump off
                2.1.4.3 print "Pump Deactivated"
        2.2 else
            2.2.1 print current_thresh
           2.2.2 print "No Watering Required"
           2.2.3 print "Pump not Activated"
           2.2.4 Repeat from step 2
3.else
        3.1 print "Not right time to water"
```


Start
Declare
min_thresh
&
max_thresh
if
6am<=time<=6pm
Yes
if
current_thresh<min_thresh
Yes
Pump On
print Pump
ON
if
current_thresh==max_thresh
Yes
Pump Off
print Pump
OFF
No
No
No
print Current_thresh
print No Watering required
print PUMP OFF
Print NOT
Right time
to Water
Stop