Coding and Solutioning

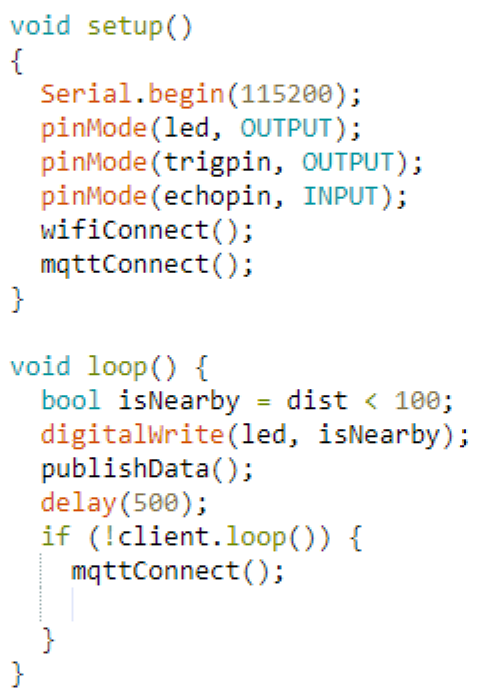
Utilization of Algorithms, Dynamic Programming, Optimization

Date	19 November 2022
Team ID	PNT2022TMID51098
Project Name	Real-Time River Water Quality Monitoring and Control System
Maximum Marks	2 Marks

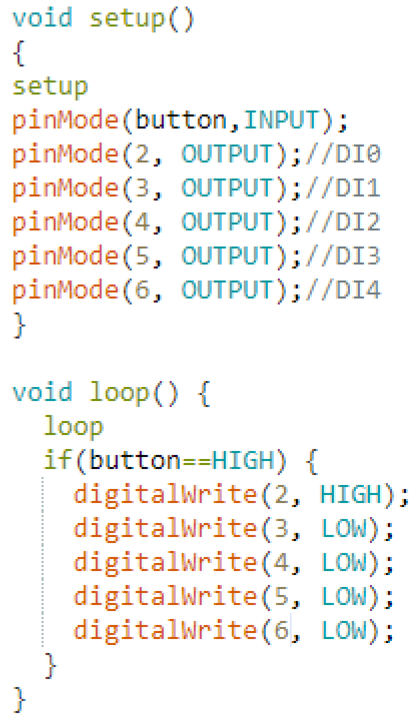
Utilization of Algorithms

1. Data from the sensor nodes are being gathered by us.
2. The Node-RED platform's IBM cloud connection configuration needs to be set up.
3. Next, it can link the Node-RED platform with IBM Watson IoT.
4. The information is then sent to the IBM Watson IoT platform.
5. The app that we need must be designed and developed.
6. Also link the app to Node-RED.
7. As a result, our mobile app can simply display the water's pH and turbidity levels in real time.
8. A motor controller was required if we wanted to close the specific dam.
9. As a result, we developed a motor controller for a mobile app.
10. Results from the controller are displayed in app.

Dynamic Programming

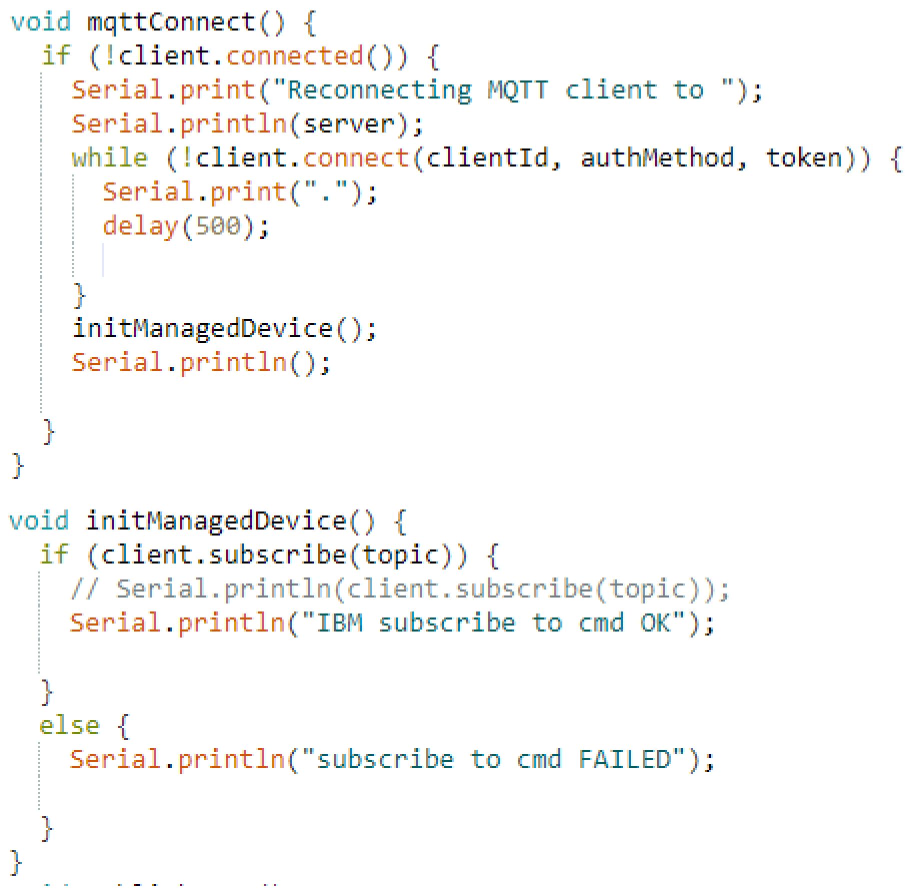


```
void setup()
{
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {
  bool isNearby = dist < 100;
  digitalWrite(led, isNearby);
  publishData();
  delay(500);
  if (!client.loop()) {
    mqttConnect();

  }
}
```



```
void setup()
{
setup
pinMode(button,INPUT);
pinMode(2, OUTPUT);//DI0
pinMode(3, OUTPUT);//DI1
pinMode(4, OUTPUT);//DI2
pinMode(5, OUTPUT);//DI3
pinMode(6, OUTPUT);//DI4
}

void loop() {
  loop
  if(button==HIGH) {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
  }
}
```

Optimization



```
void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to ");
    Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);

    }
    initManagedDevice();
    Serial.println();

  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    // Serial.println(client.subscribe(topic));
    Serial.println("IBM subscribe to cmd OK");

  }
  else {
    Serial.println("subscribe to cmd FAILED");

  }
}
```

```
void publishData()
{
  digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2;
  if(dist<100){
    String payload = "{\"Alert Distance is\":";
    payload += dist;
    payload += "}";
    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if(client.publish(publishTopic, (char*) payload.c_str())) {
      Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
      digitalWrite(led,HIGH);

    }

  }

  if(dist>101 && dist<111){
    String payload = "{\"Normal Distance\":";
    payload += dist;
    payload += "}";
    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
  }
}
```