

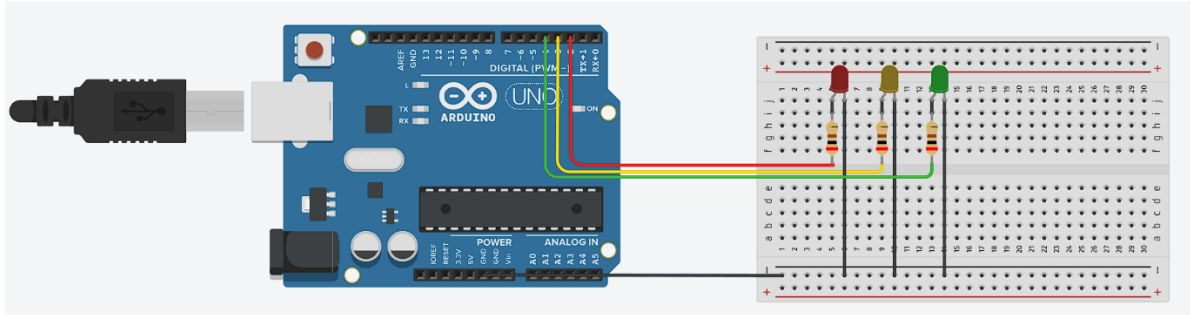
Documentation  
by Andrey Sysoev

## Traffic light (task.1)

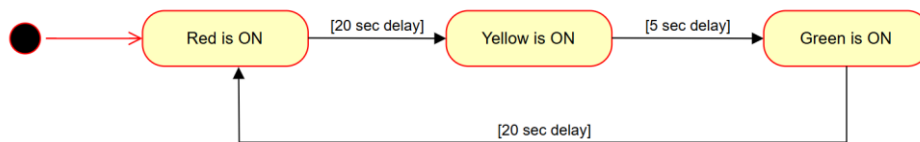
- Objective: Implement a basic traffic light system using Arduino.

The traffic light cycles through RED, YELLOW, and GREEN states with fixed durations.

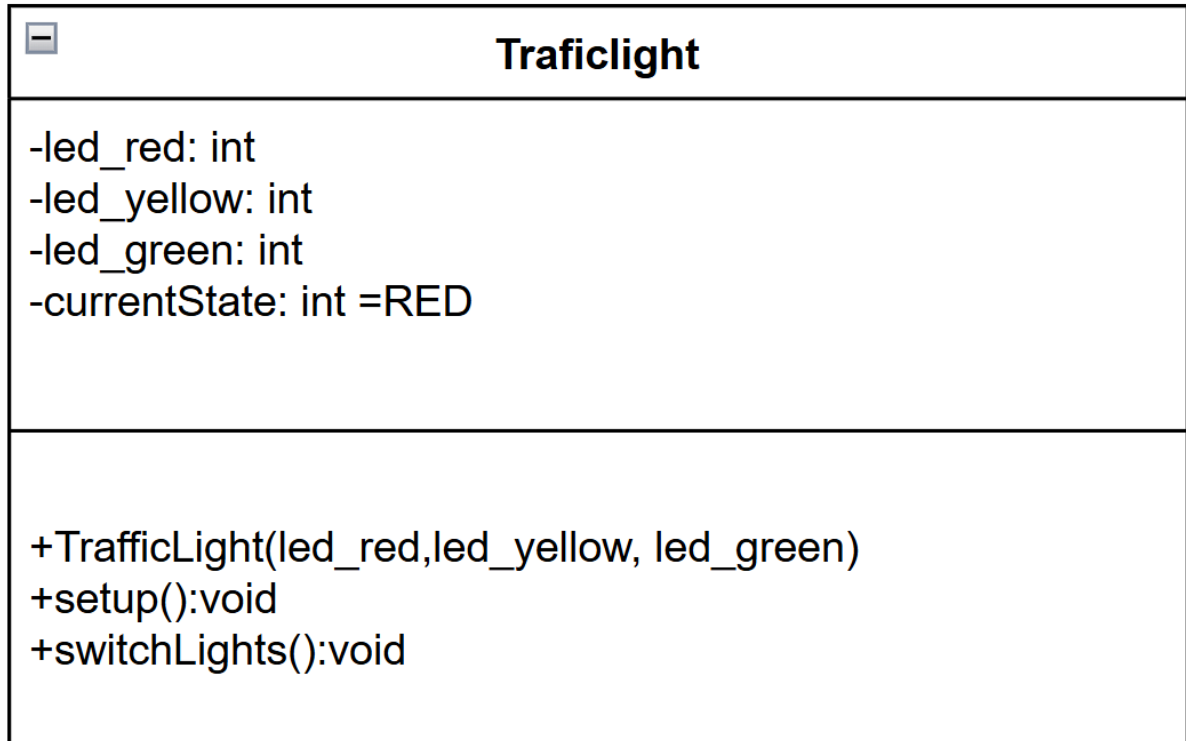
- Circuit model includes: Arduino board, 3 LEDs, 3 220 ohm resistors, breadboard, wires.



- State machine diagram:



- Class diagram:



- Explanation of a code:

The traffic light class initialize with 3 pins and a starting state (RED):

```
class TrafficLight {  
  private:  
    int led_red;  
    int led_yellow;  
    int led_green;  
    int currentState=RED;
```

Traffic light cycles between 3 states red- yellow – green, with timing red(20 sec) yellow(5 sec) green(20sec):

```
void switchLights() {  
  switch (currentState) {  
    case RED:  
      digitalWrite(led_red, HIGH);  
      digitalWrite(led_yellow, LOW);  
      digitalWrite(led_green, LOW);  
      delay(20000);  
      currentState = YELLOW;  
      break;  
  
    case YELLOW:  
      digitalWrite(led_red, LOW);  
      digitalWrite(led_yellow, HIGH);  
      digitalWrite(led_green, LOW);  
      delay(5000);  
      currentState = GREEN;  
      break;  
  
    case GREEN:  
      digitalWrite(led_red, LOW);  
      digitalWrite(led_yellow, LOW);  
      digitalWrite(led_green, HIGH);  
      delay(20000);  
      currentState = RED;  
      break;  
  }  
}
```

- Code:

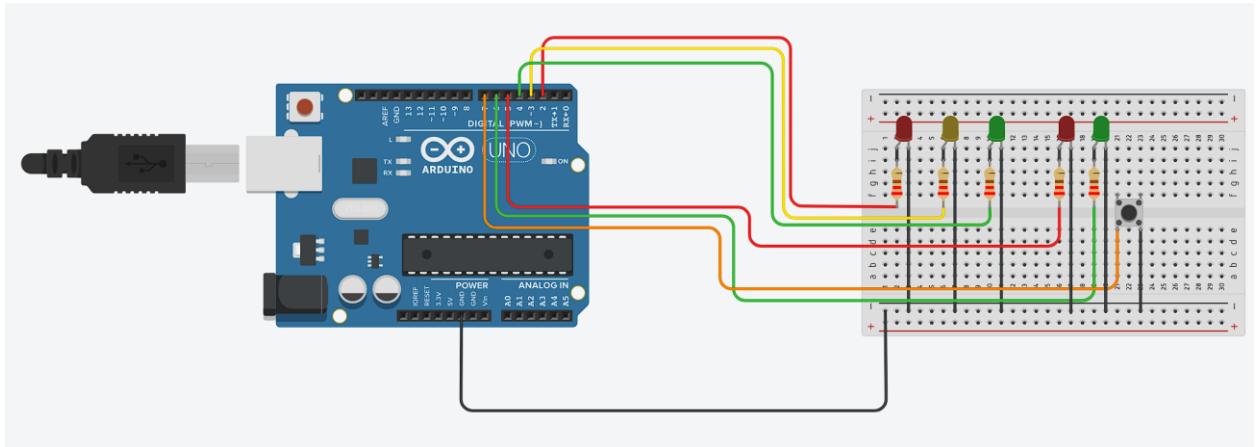
```

1 #define RED 100
2 #define YELLOW 200
3 #define GREEN 300
4 class TrafficLight {
5     private:
6         int led_red;
7         int led_yellow;
8         int led_green;
9         int currentState=RED;
10
11     public:
12         TrafficLight(int red, int yel, int gre) {
13             led_red = red;
14             led_yellow = yel;
15             led_green = gre;
16         }
17
18         void setup() {
19             pinMode(led_red, OUTPUT);
20             pinMode(led_yellow, OUTPUT);
21             pinMode(led_green, OUTPUT);
22         }
23
24         void switchLights() {
25             switch (currentState) {
26                 case RED:
27                     digitalWrite(led_red, HIGH);
28                     digitalWrite(led_yellow, LOW);
29                     digitalWrite(led_green, LOW);
30                     delay(20000);
31                     currentState = YELLOW;
32                     break;
33
34                 case YELLOW:
35                     digitalWrite(led_red, LOW);
36                     digitalWrite(led_yellow, HIGH);
37                     digitalWrite(led_green, LOW);
38                     delay(5000);
39                     currentState = GREEN;
40                     break;
41
42                 case GREEN:
43                     digitalWrite(led_red, LOW);
44                     digitalWrite(led_yellow, LOW);
45                     digitalWrite(led_green, HIGH);
46                     delay(20000);
47                     currentState = RED;
48                     break;
49             }
50         }
51 };
52
53 TrafficLight trafficLight(2, 3, 4);
54
55 void setup() {
56     trafficLight.setup();
57 }
58
59 void loop() {
60     trafficLight.switchLights();
61 }

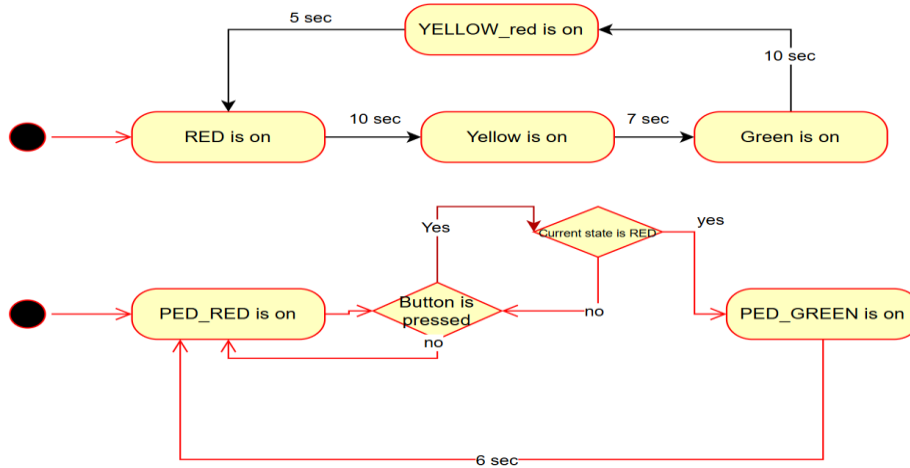
```

## Traffic light with pedestrian light (task.2)

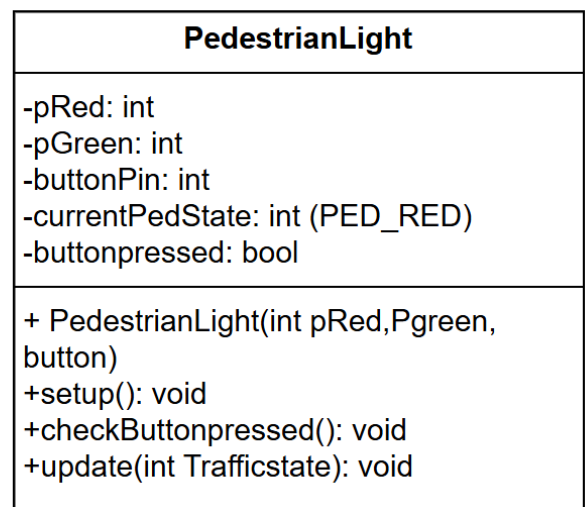
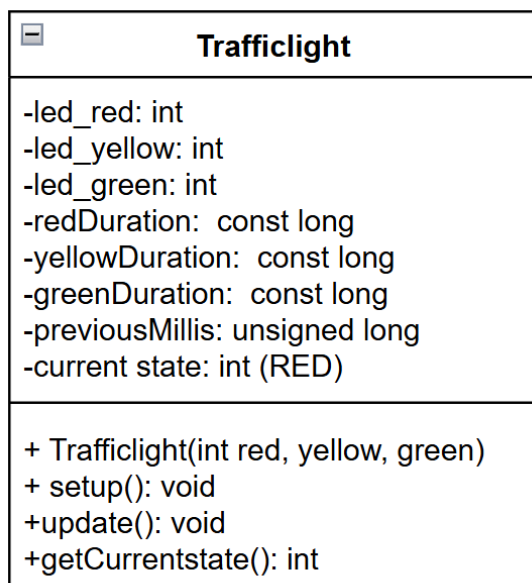
- **Objective:** extend traffic light system by adding pedestrian light, by pushing button, light for pedestrian sidewalk should be turned green only if current traffic light is red.
- **Circuit model includes:** Arduino board, 5 LEDs, 5 220 ohm resistors, breadboard, wires.



- **State machine diagram:**



- **Class diagram:**



Explanation of a code:

Traffic light cycles through RED, YELLOW, GREEN, YELLOW\_RED states:

```
switch (currentState) {
  case RED:
    if (currentMillis - previousMillis >= redDuration) {
      previousMillis = currentMillis;
      currentState = YELLOW;
      digitalWrite(led_yellow, HIGH);
      delay(2000);
      digitalWrite(led_red, LOW);
    }
    break;

  case YELLOW:
    if (currentMillis - previousMillis >= yellowDuration) {
      previousMillis = currentMillis;
      currentState = GREEN;
      digitalWrite(led_yellow, LOW);
      digitalWrite(led_green, HIGH);
    }
    break;

  case GREEN:
    if (currentMillis - previousMillis >= greenDuration) {
      previousMillis = currentMillis;
      currentState = YELLOW_RED;
      digitalWrite(led_green, LOW);
      digitalWrite(led_yellow, HIGH);
    }
    break;

  case YELLOW_RED:
    if (currentMillis - previousMillis >= yellowDuration) {
      previousMillis = currentMillis;
      currentState = RED;
      digitalWrite(led_yellow, LOW);
      digitalWrite(led_red, HIGH);
    }
    break;
}
```

Time between states cycle defines by millis() function.

A variable previousMillis is used to store time after last state happened.

```
unsigned long previousMillis = 0;
```

Arduino has own clock, by using millis() we call every loop iteration to get current time.

```
unsigned long currentMillis = millis();
```

If a time between current and stored (for example red state duration should be 10 sec) is greater or equal, it resets and moves to next state.

```
if (currentMillis - previousMillis >= redDuration) {
  previousMillis = currentMillis;
  currentState = YELLOW;
```

For button is used Boolean function, if it has been pressed button changes from false to true.

```
bool buttonPressed = false;
void checkButtonPress() {
  if (digitalRead(buttonPin) == LOW) {
    buttonPressed = true;
  }
}
```

If button is pressed and current state of traffic light is red, pedestrian light turns green for 6 sec.

```
switch (currentPedState) {
  case PED_RED:
    if (buttonPressed && trafficState == RED) {
      currentPedState = PED_GREEN;
      buttonPressed = false;
      digitalWrite(pRed, LOW);
      digitalWrite(pGreen, HIGH);
    }
    break;

  case PED_GREEN:
    delay(6000);
    currentPedState = PED_RED;
    digitalWrite(pGreen, LOW);
    digitalWrite(pRed, HIGH);
    break;
}
```

To get current state of traffic light.

```
int getCurrentState() {
  return currentState;
}
```

- Code:

```

1  #define RED 100
2  #define YELLOW 200
3  #define GREEN 300
4  #define YELLOW_RED 400
5  #define PED_GREEN 500
6  #define PED_RED 600
7
8  class TrafficLight {
9      private:
10         int led_red, led_yellow, led_green;
11         const long redDuration = 10000;
12         const long yellowDuration = 5000;
13         const long greenDuration = 10000;
14         int currentState = RED;
15         unsigned long previousMillis = 0;
16
17     public:
18         TrafficLight(int red, int yellow, int green) {
19             led_red = red;
20             led_yellow = yellow;
21             led_green = green;
22         }
23
24         void setup() {
25             pinMode(led_red, OUTPUT);
26             pinMode(led_yellow, OUTPUT);
27             pinMode(led_green, OUTPUT);
28             digitalWrite(led_red, HIGH);
29             digitalWrite(led_yellow, LOW);
30             digitalWrite(led_green, LOW);
31         }
32
33         void update() {
34             unsigned long currentMillis = millis();
35
36             switch (currentState) {
37                 case RED:
38                     if (currentMillis - previousMillis >= redDuration) {
39                         previousMillis = currentMillis;
40                         currentState = YELLOW;
41                         digitalWrite(led_yellow, HIGH);
42                         delay(2000);
43                         digitalWrite(led_red, LOW);
44                     }
45                     break;
46
47                 case YELLOW:
48                     if (currentMillis - previousMillis >= yellowDuration)
49                         previousMillis = currentMillis;
50                         currentState = GREEN;
51                         digitalWrite(led_yellow, LOW);
52                         digitalWrite(led_green, HIGH);
53                     }
54                     break;
55
56                 case GREEN:
57                     if (currentMillis - previousMillis >= greenDuration) {
58                         previousMillis = currentMillis;
59                         currentState = YELLOW_RED;
60                         digitalWrite(led_green, LOW);
61                         digitalWrite(led_yellow, HIGH);
62                     }
63                     break;
64
65                 case YELLOW_RED:
66                     if (currentMillis - previousMillis >= yellowDuration)
67                         previousMillis = currentMillis;
68                         currentState = RED;
69                         digitalWrite(led_yellow, LOW);
70                         digitalWrite(led_red, HIGH);
71                     }
72                     break;
73             }

```



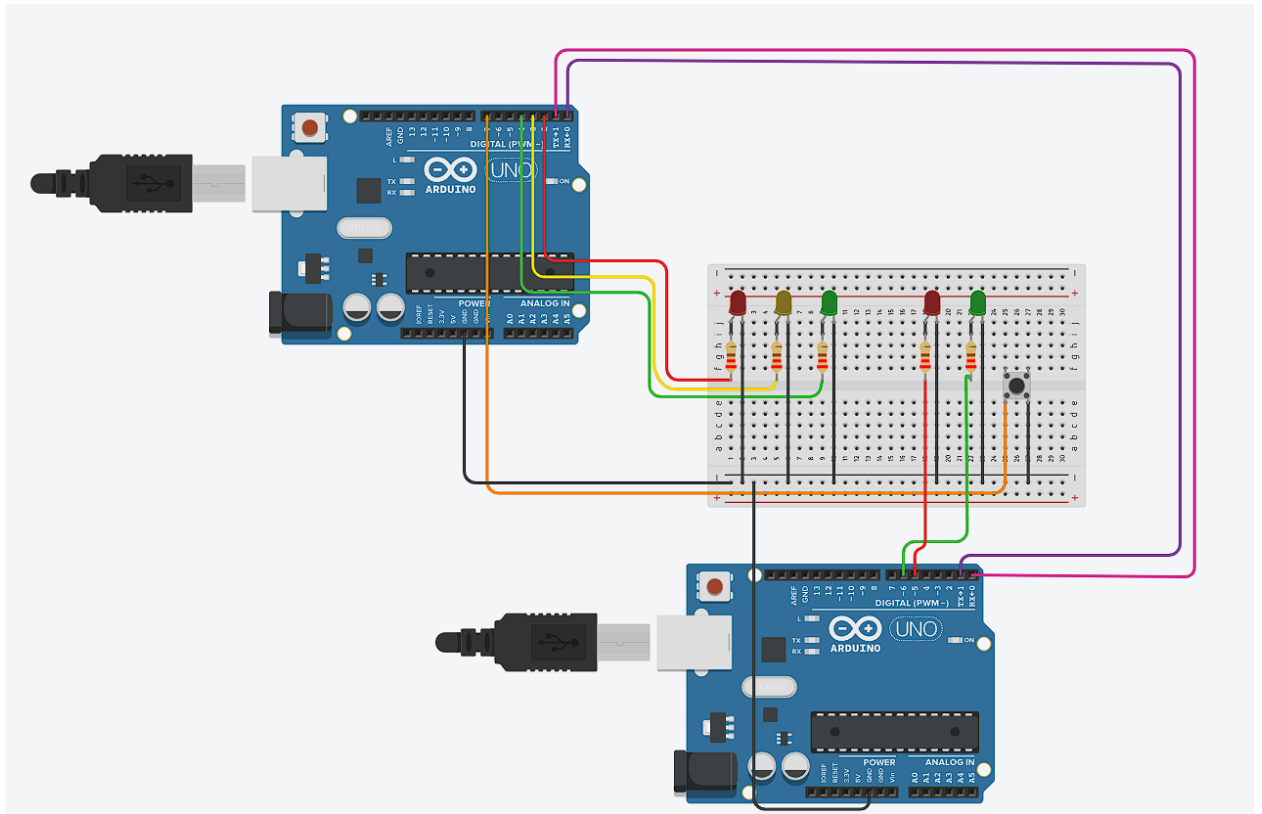
```

74     }
75
76     int getCurrentState() {
77         return currentState;
78     }
79 };
80
81 class PedestrianLight {
82 private:
83     int pRed, pGreen, buttonPin;
84     int currentPedState = PED_RED;
85     bool buttonPressed = false;
86
87 public:
88     PedestrianLight(int red, int green, int button) {
89         pRed = red;
90         pGreen = green;
91         buttonPin = button;
92     }
93
94     void setup() {
95         pinMode(pRed, OUTPUT);
96         pinMode(pGreen, OUTPUT);
97         pinMode(buttonPin, INPUT_PULLUP);
98         digitalWrite(pRed, HIGH);
99         digitalWrite(pGreen, LOW);
100     }
101
102     void checkButtonPress() {
103         if (digitalRead(buttonPin) == LOW) {
104             buttonPressed = true;
105         }
106     }
107
108     void update(int trafficState) {
109         switch (currentPedState) {
110             case PED_RED:
111                 if (buttonPressed && trafficState == RED) {
112                     currentPedState = PED_GREEN;
113                     buttonPressed = false;
114                     digitalWrite(pRed, LOW);
115                     digitalWrite(pGreen, HIGH);
116                 }
117                 break;
118
119             case PED_GREEN:
120                 delay(6000);
121                 currentPedState = PED_RED;
122                 digitalWrite(pGreen, LOW);
123                 digitalWrite(pRed, HIGH);
124                 break;
125         }
126     }
127 };
128
129 TrafficLight trafficLight(2, 3, 4);
130 PedestrianLight pedestrianLight(5, 6, 7);
131
132 void setup() {
133     trafficLight.setup();
134     pedestrianLight.setup();
135 }
136
137 void loop() {
138     trafficLight.update();
139     pedestrianLight.checkButtonPress();
140     pedestrianLight.update(trafficLight.getCurrentState());
141 }

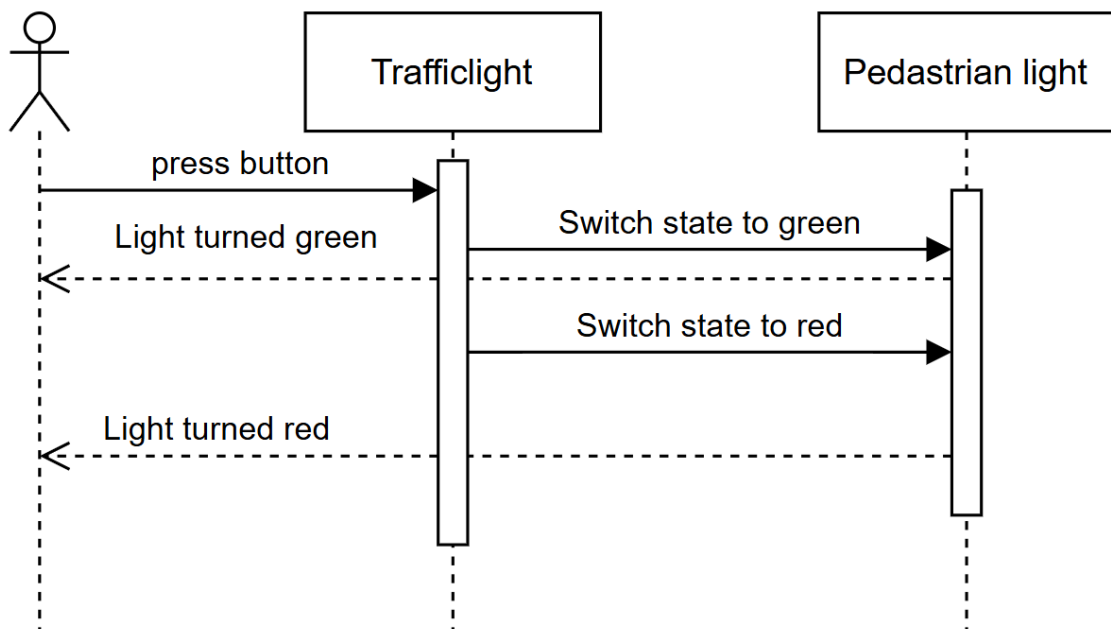
```

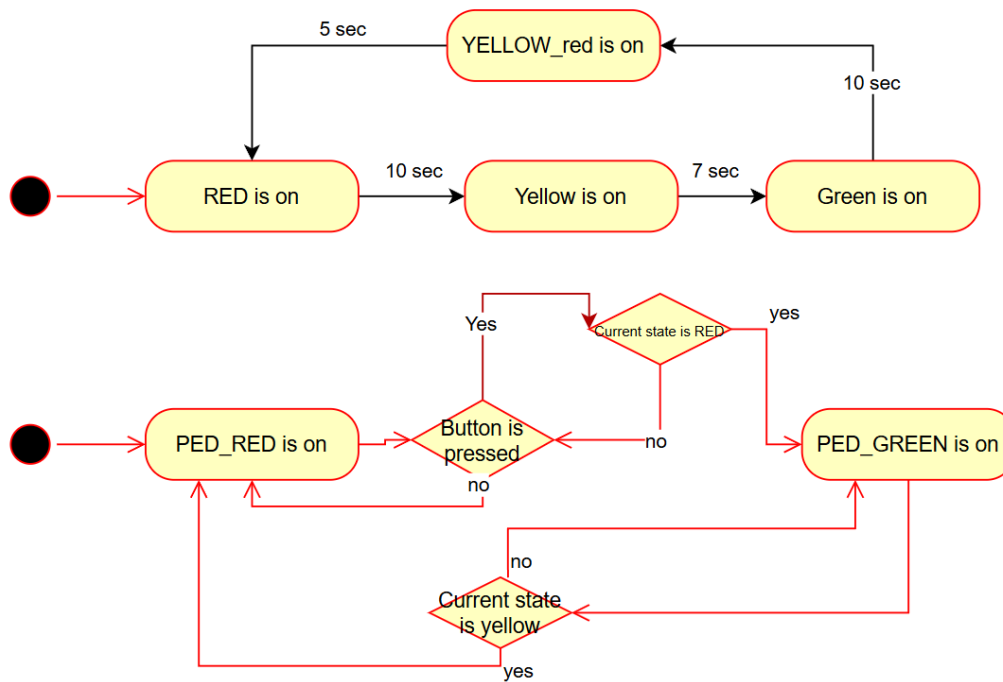
### Traffic and Pedestrian Light System with Serial Communication (Task 5)

- Objective: establish communication between two separate  $\mu\text{C}$  for the car traffic light and the pedestrian traffic light.
- Circuit model includes: 2 Arduino board, 5 LEDs, 5 220 ohm resistors, breadboard, wires.



- Diagrams:





- Code:

```

1  #define RED 100
2  #define YELLOW 200
3  #define GREEN 300
4  #define YELLOW_RED 400
5
6  class TrafficLight {
7  private:
8      int led_red, led_yellow, led_green, buttonPin;
9      const long redDuration = 10000;
10     const long yellowDuration = 5000;
11     const long greenDuration = 10000;
12
13     unsigned long previousMillis = 0;
14     bool buttonPressed = false;
15     int currentState = RED;
16
17 public:
18     TrafficLight(int red, int yellow, int green, int button) {
19         led_red = red;
20         led_yellow = yellow;
21         led_green = green;
22         buttonPin = button;
23     }
24
25     void setup() {
26         pinMode(led_red, OUTPUT);
27         pinMode(led_yellow, OUTPUT);
28         pinMode(led_green, OUTPUT);
29         pinMode(buttonPin, INPUT_PULLUP);
30         digitalWrite(led_red, HIGH);
31         digitalWrite(led_yellow, LOW);
32         digitalWrite(led_green, LOW);
33     }
34
35     void checkButtonPress() {
36         if (digitalRead(buttonPin) == LOW) {
37             buttonPressed = true;
38         }
39     }
40
41     void update() {
42         unsigned long currentMillis = millis();

```

```

44     switch (currentState) {
45         case RED:
46             if (buttonPressed) {
47                 Serial.write(100);
48                 buttonPressed = false;
49             }
50             digitalWrite(led_yellow, LOW);
51             digitalWrite(led_red, HIGH);
52
53             if (currentMillis - previousMillis >= redDuration) {
54                 previousMillis = currentMillis;
55                 currentState = YELLOW;
56
57             }
58             break;
59
60         case YELLOW:
61             digitalWrite(led_yellow, HIGH);
62             Serial.write(200);
63             delay(2000);
64             digitalWrite(led_red, LOW);
65
66             if (currentMillis - previousMillis >= yellowDuration)
67                 previousMillis = currentMillis;
68                 currentState = GREEN;
69
70             }
71             break;
72
73         case GREEN:
74             digitalWrite(led_yellow, LOW);
75             digitalWrite(led_green, HIGH);
76
77             if (currentMillis - previousMillis >= greenDuration) {
78                 previousMillis = currentMillis;
79                 currentState = YELLOW_RED;
80
81             }
82             break;
83
84         case YELLOW_RED:
85             digitalWrite(led_green, LOW);
86             digitalWrite(led_yellow, HIGH);
87
88             if (currentMillis - previousMillis >= yellowDuration) {
89                 previousMillis = currentMillis;
90                 currentState = RED;
91             }
92             break;
93             break;
94     }
95 };
96
97 TrafficLight trafficLight(2, 3, 4, 7);
98
99 void setup() {
100     Serial.begin(9600);
101     trafficLight.setup();
102 }
103
104 void loop() {
105     trafficLight.checkButtonPress();
106     trafficLight.update();
107 }

```

```

1  #define PED_GREEN 500
2  #define PED_RED 600
3  #define RED 100
4  #define YELLOW 200
5
6  class PedestrianLight {
7      private:
8          int pRed, pGreen;
9          int currentPedState = PED_RED;
10
11     public:
12         PedestrianLight(int red, int green) {
13             pRed = red;
14             pGreen = green;
15         }
16
17         void setup() {
18             pinMode(pRed, OUTPUT);
19             pinMode(pGreen, OUTPUT);
20             digitalWrite(pRed, HIGH);
21             digitalWrite(pGreen, LOW);
22         }
23
24         void update() {
25             if (Serial.available() > 0) {
26                 int trafficState = Serial.read();
27
28                 switch (currentPedState) {
29                     case PED_RED:
30                         if (trafficState == RED) {
31                             currentPedState = PED_GREEN;
32                             digitalWrite(pRed, LOW);
33                             digitalWrite(pGreen, HIGH);
34                         }
35                         break;
36
37                     case PED_GREEN:
38                         if (trafficState == YELLOW) {
39                             currentPedState = PED_RED;
40                             digitalWrite(pGreen, LOW);
41                             digitalWrite(pRed, HIGH);
42                         }
43                         break;
44                 }
45             }
46         }
47     };
48
49     PedestrianLight pedestrianLight(5, 6);
50
51     void setup() {
52         Serial.begin(9600);
53         pedestrianLight.setup();
54     }
55
56     void loop() {
57         pedestrianLight.update();
58     }

```