ASSIGNMENT-3 (B7-IA3E)

```
//code for trafficlight system//
```

```
import turtle  # Allows us to use turtles
turtle.setup(400, 600)  # Determine the window size
wn = turtle.Screen()  # Creates a playground for turtles
wn.title('traffic light using different turtles')  # Set the window title
wn.bgcolor('skyblue')  # Set the window background color
tess = turtle.Turtle()  # Create a turtle, assign to tess
alex = turtle.Turtle()  # Create alex
henry = turtle.Turtle()  # Create henry


def draw_housing():
    """ Draw a nice housing to hold the traffic lights"""
    tess.pensize(3)  # Change tess' pen width
    tess.color('black', 'white')  # Set tess' color
    tess.begin_fill()  # Tell tess to start filling the color
    tess.forward(80)  # Tell tess to move forward by 80 units
    tess.left(90)  # Tell tess to turn left by 90 degrees
    tess.forward(200)
    tess.circle(40, 180)  # Tell tess to draw a semi-circle
    tess.forward(200)
    tess.left(90)
    tess.end_fill()  # Tell tess to stop filling the color


draw_housing()


def circle(t, ht, colr):
    """Position turtle onto the place where the lights should be, and
    turn turtle into a big circle"""
    t.penup()  # This allows us to move a turtle without drawing a line
    t.forward(40)
    t.left(90)
    t.forward(ht)
    t.shape('circle')  # Set tutle's shape to circle
    t.shapesize(3)  # Set size of circle
    t.fillcolor(colr)  # Fill color in circle


circle(tess, 50, 'green')
circle(alex, 120, 'orange')
circle(henry, 190, 'red')
state_num = 0
```

```
def advance_state_machine():
    """A state machine for traffic light"""
    global state_num  # Tells Python not to create a new local variable for
state_num

    if state_num == 0:  # Transition from state 0 to state 1
        henry.color('darkgrey')
        alex.color('darkgrey')
        tess.color('green')
        wn.ontimer(advance_state_machine, 3000)  # set the timer to explode in 3
sec
        state_num = 1
    elif state_num == 1:  # Transition from state 1 to state 2
        henry.color('darkgrey')
        alex.color('orange')
        wn.ontimer(advance_state_machine, 1000)
        state_num = 2
    elif state_num == 2:  # Transition from state 2 to state 3
        tess.color('darkgrey')
        wn.ontimer(advance_state_machine, 1000)
        state_num = 3
    else:                 # Transition from state 3 to state 0
        henry.color('red')
        alex.color('darkgrey')
        wn.ontimer(advance_state_machine, 2000)
        state_num = 0


advance_state_machine()

wn.listen()  # Listen for events

wn.mainloop()  # Wait for user to close window
```

```
//code for blinking LED//
```

```
#!/usr/bin/env python
import RPi.GPIO as GPIO # RPi.GPIO can be referred as GPIO from now
import time

ledPin = 22    # pin22

def setup():
        GPIO.setmode(GPIO.BOARD)       # GPIO Numbering of Pins
        GPIO.setup(ledPin, GPIO.OUT)   # Set ledPin as output
        GPIO.output(ledPin, GPIO.LOW)  # Set ledPin to LOW to turn Off the LED
 def loop():
        while True:
                print 'LED on'
                GPIO.output(ledPin, GPIO.HIGH)   # LED On
                time.sleep(1.0)                  # wait 1 sec
                print 'LED off'
                GPIO.output(ledPin, GPIO.LOW)   # LED Off
                time.sleep(1.0)                 # wait 1 sec
def endprogram():

        GPIO.output(ledPin, GPIO.LOW)     # LED Off
        GPIO.cleanup()                    # Release resources

if __name__ == '__main__':         # Program starts from here
        setup()
        try:
                loop()
        except KeyboardInterrupt:  # When 'Ctrl+C' is pressed, the destroy() will
be  executed.
                endprogram()
```