```
#Ashley Conejo 100869378


# Include the library files
import I2C_LCD_driver
import RPi.GPIO as GPIO
from time import sleep

# Enter column pins
C1 = 5
C2 = 6
C3 = 13
C4 = 19

# Enter row pins
R1 = 12
R2 = 16
R3 = 20
R4 = 21

# Enter buzzer pin
buzzer = 17

# Enter green LED pin
greenLED = 27

# Enter red LED pin
redLED = 22

# Create an object for the LCD
lcd = I2C_LCD_driver.lcd()
# Starting text
lcd.lcd_display_string("System loading", 1, 1)
for a in range(0, 16):
    lcd.lcd_display_string(".", 2, a)
    sleep(0.1)
lcd.lcd_clear()

# The GPIO pin of the column of the key that is currently
# being held down or -1 if no key is pressed
keypadPressed = -1

# Enter your PIN
```

```
secretCode = "A55C"
input = ""

# Setup GPIO
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(buzzer, GPIO.OUT)
GPIO.setup(greenLED, GPIO.OUT)
GPIO.setup(redLED, GPIO.OUT)
GPIO.output(greenLED, GPIO.LOW)
GPIO.output(redLED, GPIO.LOW)

# Set column pins as output pins
GPIO.setup(C1, GPIO.OUT)
GPIO.setup(C2, GPIO.OUT)
GPIO.setup(C3, GPIO.OUT)
GPIO.setup(C4, GPIO.OUT)

# Set row pins as input pins
GPIO.setup(R1, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(R2, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(R3, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
GPIO.setup(R4, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

# This callback registers the key that was pressed if no other key is currently pressed
def keypadCallback(channel):
    global keypadPressed
    if keypadPressed == -1:
        keypadPressed = channel

# Detect the rising edges
GPIO.add_event_detect(R1, GPIO.RISING, callback=keypadCallback)
GPIO.add_event_detect(R2, GPIO.RISING, callback=keypadCallback)
GPIO.add_event_detect(R3, GPIO.RISING, callback=keypadCallback)
GPIO.add_event_detect(R4, GPIO.RISING, callback=keypadCallback)

# Sets all rows to a specific state.
def setAllRows(state):
    GPIO.output(C1, state)
    GPIO.output(C2, state)
    GPIO.output(C3, state)
    GPIO.output(C4, state)

# Check or clear PIN
```

```
def commands():
    global input
    pressed = False
    GPIO.output(C1, GPIO.HIGH)

    # Clear PIN
    if GPIO.input(R1) == 1:
        print("Input reset!")
        lcd.lcd_clear()
        lcd.lcd_display_string("Clear", 1, 5)
        sleep(1)
        pressed = True
    GPIO.output(C1, GPIO.HIGH)
    # Check PIN
    if not pressed and GPIO.input(R2) == 1:
        if input == secretCode:
            print("Valid Key.")
            lcd.lcd_clear()
            lcd.lcd_display_string("Successful", 1, 3)
            GPIO.output(greenLED, GPIO.HIGH)
            GPIO.output(buzzer, GPIO.HIGH)
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
            sleep(0.3)
            GPIO.output(greenLED, GPIO.LOW)
            GPIO.output(buzzer, GPIO.HIGH)
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
            sleep(0.3)
            GPIO.output(greenLED, GPIO.HIGH)
            GPIO.output(buzzer, GPIO.HIGH)
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
        else:
            print("Invalid key - Try Again.")
            lcd.lcd_clear()
            lcd.lcd_display_string("Invalid PIN!", 1, 3)
            GPIO.output(redLED, GPIO.HIGH)
            GPIO.output(buzzer, GPIO.HIGH)
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
            sleep(0.3)
            GPIO.output(redLED, GPIO.LOW)
            GPIO.output(buzzer, GPIO.HIGH)
```

```
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
            sleep(0.3)
            GPIO.output(redLED, GPIO.HIGH)
            GPIO.output(buzzer, GPIO.HIGH)
            sleep(0.3)
            GPIO.output(buzzer, GPIO.LOW)
        pressed = True
    GPIO.output(C1, GPIO.LOW)
    if pressed:
        input = ""
    return pressed

# Reads the columns and appends the value, that corresponds to the button, to a variable
def read(column, characters):
    global input
    GPIO.output(column, GPIO.HIGH)
    if GPIO.input(R1) == 1:
        input = input + characters[0]
        print(input)
        lcd.lcd_display_string(str(input), 2, 0)
    if GPIO.input(R2) == 1:
        input = input + characters[1]
        print(input)
        lcd.lcd_display_string(str(input), 2, 0)
    if GPIO.input(R3) == 1:
        input = input + characters[2]
        print(input)
        lcd.lcd_display_string(str(input), 2, 0)
    if GPIO.input(R4) == 1:
        input = input + characters[3]
        print(input)
        lcd.lcd_display_string(str(input), 2, 0)
    GPIO.output(column, GPIO.LOW)

try:
    while True:
        lcd.lcd_display_string("Enter your PIN:", 1, 0)

        # If a button was previously pressed,
        # check whether the user has released it yet
        if keypadPressed != -1:
            setAllRows(GPIO.HIGH)
            if GPIO.input(keypadPressed) == 0:
```

```
            keypadPressed = -1
        else:
            sleep(0.1)
    # Otherwise, just read the input
    else:
        if not commands():
            read(C1, ["D", "C", "B", "A"])
            read(C2, ["#", "9", "6", "3"])
            read(C3, ["0", "8", "5", "2"])
            read(C4, ["*", "7", "4", "1"])
            sleep(0.1)
        else:
            sleep(0.1)
except KeyboardInterrupt:
    print("Stopped!")
```