```
"""
File: LEDCube.py
Class: ME3350
Team: Ethan Barlow, Ryan DaVisio, Dan Middleton
Date: TBD
================================================================================
===============
SETUP
================================================================================
===============
"""
#================================================================================
===============
#GPIO Data:
import RPi.GPIO as GPIO
import time
from time import sleep
import math
import random
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(2,GPIO.OUT)
GPIO.setup(3,GPIO.OUT)
GPIO.setup(4,GPIO.OUT)
GPIO.setup(14,GPIO.OUT)
GPIO.setup(15,GPIO.OUT)
GPIO.setup(18,GPIO.OUT)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(27,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(24,GPIO.OUT)
GPIO.setup(10,GPIO.OUT)
GPIO.setup(9,GPIO.OUT)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(8,GPIO.OUT)
GPIO.setup(7,GPIO.OUT)
GPIO.setup(16,GPIO.OUT)
GPIO.setup(26,GPIO.OUT)
GPIO.setup(20,GPIO.OUT)
GPIO.setup(21,GPIO.OUT)
#================================================================================
===============
def on(x,y,z): #Turns on an LED if given its coordinates.

#--------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
```

```
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------------
-----------
    #Actual Code:
    if z==1: #LED17 (Transistor 1)
        GPIO.output(16,True)
    elif z==2: #LED18 (Transistor 2)
        GPIO.output(26,True)
    elif z==3: #LED19 (Transistor 3)
        GPIO.output(20,True)
    elif z==4: #LED20 (Transistor 4)
        GPIO.output(21,True)
    if y==1:
        if x==1: #LED1
            GPIO.output(2,True)
        elif x==2: #LED2
            GPIO.output(3,True)
        elif x==3: #LED3
            GPIO.output(4,True)
        elif x==4: #LED4
            GPIO.output(14,True)
    elif y==2:
        if x==1: #LED5
            GPIO.output(15,True)
        elif x==2: #LED6
```

```
            GPIO.output(18,True)
        elif x==3: #LED7
            GPIO.output(17,True)
        elif x==4: #LED8
            GPIO.output(27,True)
    elif y==3:
        if x==1: #LED9
            GPIO.output(22,True)
        elif x==2: #LED10
            GPIO.output(23,True)
        elif x==3: #LED11
            GPIO.output(24,True)
        elif x==4: #LED12
            GPIO.output(10,True)
    elif y==4:
        if x==1: #LED13
            GPIO.output(9,True)
        elif x==2: #LED14
            GPIO.output(25,True)
        elif x==3: #LED15
            GPIO.output(8,True)
        elif x==4: #LED16
            GPIO.output(7,True)
#================================================================================
==============
def off(x,y,z): #Turns off an LED if given its coordinates.

#--------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
```

```
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------------------
-----------
    #Actual Code:
    if z==1: #LED17 (Transistor 1)
        GPIO.output(16,True)
    elif z==2: #LED18 (Transistor 2)
        GPIO.output(26,True)
    elif z==3: #LED19 (Transistor 3)
        GPIO.output(20,True)
    elif z==4: #LED20 (Transistor 4)
        GPIO.output(21,True)
    if y==1:
        if x==1: #LED1
            GPIO.output(2,False)
        elif x==2: #LED2
            GPIO.output(3,False)
        elif x==3: #LED3
            GPIO.output(4,False)
        elif x==4: #LED4
            GPIO.output(14,False)
    elif y==2:
        if x==1: #LED5
            GPIO.output(15,False)
        elif x==2: #LED6
            GPIO.output(18,False)
        elif x==3: #LED7
            GPIO.output(17,False)
        elif x==4: #LED8
            GPIO.output(27,False)
    elif y==3:
        if x==1: #LED9
            GPIO.output(22,False)
        elif x==2: #LED10
            GPIO.output(23,False)
        elif x==3: #LED11
            GPIO.output(24,False)
        elif x==4: #LED12
            GPIO.output(10,False)
    elif y==4:
        if x==1: #LED13
            GPIO.output(9,False)
        elif x==2: #LED14
```

```
            GPIO.output(25,False)
        elif x==3: #LED15
            GPIO.output(8,False)
        elif x==4: #LED16
            GPIO.output(7,False)
#================================================================================
==============
def alloff(): #Note: This doesn't work in every circumstance due to electrical
impossibility.

#--------------------------------------------------------------------------------
----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#--------------------------------------------------------------------------------
-----------
    #Reset GPIO (turn everything off)
    GPIO.output(2,False)
    GPIO.output(3,False)
    GPIO.output(4,False)
    GPIO.output(14,False)
    GPIO.output(15,False)
    GPIO.output(18,False)
```

```
    GPIO.output(17,False)
    GPIO.output(27,False)
    GPIO.output(22,False)
    GPIO.output(23,False)
    GPIO.output(24,False)
    GPIO.output(10,False)
    GPIO.output(9,False)
    GPIO.output(25,False)
    GPIO.output(8,False)
    GPIO.output(7,False)
    GPIO.output(16,False)
    GPIO.output(26,False)
    GPIO.output(20,False)
    GPIO.output(21,False)
#================================================================================
==============
def allon():

#--------------------------------------------------------------------------------
----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#--------------------------------------------------------------------------------
```

```
-----------
    #Turn everything on:
    GPIO.output(2,True)
    GPIO.output(3,True)
    GPIO.output(4,True)
    GPIO.output(14,True)
    GPIO.output(15,True)
    GPIO.output(18,True)
    GPIO.output(17,True)
    GPIO.output(27,True)
    GPIO.output(22,True)
    GPIO.output(23,True)
    GPIO.output(24,True)
    GPIO.output(10,True)
    GPIO.output(9,True)
    GPIO.output(25,True)
    GPIO.output(8,True)
    GPIO.output(7,True)
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
#================================================================================
===============
def xyplaneon(z): #Turns on an xy plane of LED's if given a z value.

#--------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
```

```
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    if z==1: #LED17 (Transistor 1)
        GPIO.output(16,True)
    elif z==2: #LED18 (Transistor 2)
        GPIO.output(26,True)
    elif z==3: #LED19 (Transistor 3)
        GPIO.output(20,True)
    elif z==4: #LED20 (Transistor 4)
        GPIO.output(21,True)
    GPIO.output(2,True)
    GPIO.output(3,True)
    GPIO.output(4,True)
    GPIO.output(14,True)
    GPIO.output(15,True)
    GPIO.output(18,True)
    GPIO.output(17,True)
    GPIO.output(27,True)
    GPIO.output(22,True)
    GPIO.output(23,True)
    GPIO.output(24,True)
    GPIO.output(10,True)
    GPIO.output(9,True)
    GPIO.output(25,True)
    GPIO.output(8,True)
    GPIO.output(7,True)
#=============================================================================
===============
def xyplaneoff(z): #Turns off an xy plane of LED's if given a z value.

#-------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
```

```
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    if z==1: #LED17 (Transistor 1)
        GPIO.output(16,False)
    elif z==2: #LED18 (Transistor 2)
        GPIO.output(26,False)
    elif z==3: #LED19 (Transistor 3)
        GPIO.output(20,False)
    elif z==4: #LED20 (Transistor 4)
        GPIO.output(21,False)
#===============================================================================
===============
def xzplaneon(y): #Turns on an xz plane of LED's if given a y value..

#-------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
```

```
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
    if y==1:
        GPIO.output(2,True)
        GPIO.output(3,True)
        GPIO.output(4,True)
        GPIO.output(14,True)
    elif y==2:
        GPIO.output(15,True)
        GPIO.output(18,True)
        GPIO.output(17,True)
        GPIO.output(27,True)
    elif y==3:
        GPIO.output(22,True)
        GPIO.output(23,True)
        GPIO.output(24,True)
        GPIO.output(10,True)
    else:
        GPIO.output(9,True)
        GPIO.output(25,True)
        GPIO.output(8,True)
        GPIO.output(7,True)
#===============================================================================
===============
def xzplaneoff(y): #Turns off an xz plane if given a y value.

#-------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
```

```
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
    if y==1:
        GPIO.output(2,False)
        GPIO.output(3,False)
        GPIO.output(4,False)
        GPIO.output(14,False)
    elif y==2:
        GPIO.output(15,False)
        GPIO.output(18,False)
        GPIO.output(17,False)
        GPIO.output(27,False)
    elif y==3:
        GPIO.output(22,False)
        GPIO.output(23,False)
        GPIO.output(24,False)
        GPIO.output(10,False)
    else:
```

```
        GPIO.output(9,False)
        GPIO.output(25,False)
        GPIO.output(8,False)
        GPIO.output(7,False)
#================================================================================
==============
def yzplaneon(x): #Turns on an yz plane of LED's if given an x value.

#--------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#--------------------------------------------------------------------------------
-----------
    #Actual Code:
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
    if x==1:
        GPIO.output(2,True)
        GPIO.output(15,True)
        GPIO.output(22,True)
```

```
        GPIO.output(9,True)
    elif x==2:
        GPIO.output(3,True)
        GPIO.output(18,True)
        GPIO.output(23,True)
        GPIO.output(25,True)
    elif x==3:
        GPIO.output(4,True)
        GPIO.output(17,True)
        GPIO.output(24,True)
        GPIO.output(8,True)
    else:
        GPIO.output(14,True)
        GPIO.output(27,True)
        GPIO.output(10,True)
        GPIO.output(7,True)
#================================================================================
===============
def yzplaneoff(x): #Turns off a yz plane if given a x value.

#--------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)
```

```
#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
    if x==1:
        GPIO.output(2,False)
        GPIO.output(15,False)
        GPIO.output(22,False)
        GPIO.output(9,False)
    elif x==2:
        GPIO.output(3,False)
        GPIO.output(18,False)
        GPIO.output(23,False)
        GPIO.output(25,False)
    elif x==3:
        GPIO.output(4,False)
        GPIO.output(17,False)
        GPIO.output(24,False)
        GPIO.output(8,False)
    else:
        GPIO.output(14,False)
        GPIO.output(27,False)
        GPIO.output(10,False)
        GPIO.output(7,False)
#===============================================================================
===============
def alltrans(): #Turns on all transistors.

#-------------------------------------------------------------------------------
-----------
    #GPIO Data:
    import RPi.GPIO as GPIO
    import time
    from time import sleep
    import math
    import random
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(2,GPIO.OUT)
    GPIO.setup(3,GPIO.OUT)
    GPIO.setup(4,GPIO.OUT)
    GPIO.setup(14,GPIO.OUT)
    GPIO.setup(15,GPIO.OUT)
    GPIO.setup(18,GPIO.OUT)
    GPIO.setup(17,GPIO.OUT)
    GPIO.setup(27,GPIO.OUT)
```

```
    GPIO.setup(22,GPIO.OUT)
    GPIO.setup(23,GPIO.OUT)
    GPIO.setup(24,GPIO.OUT)
    GPIO.setup(10,GPIO.OUT)
    GPIO.setup(9,GPIO.OUT)
    GPIO.setup(25,GPIO.OUT)
    GPIO.setup(8,GPIO.OUT)
    GPIO.setup(7,GPIO.OUT)
    GPIO.setup(16,GPIO.OUT)
    GPIO.setup(26,GPIO.OUT)
    GPIO.setup(20,GPIO.OUT)
    GPIO.setup(21,GPIO.OUT)

#-------------------------------------------------------------------------------
-----------
    #Actual Code:
    GPIO.output(16,True)
    GPIO.output(26,True)
    GPIO.output(20,True)
    GPIO.output(21,True)
#===============================================================================
===============
"""
================================================================================
===============
PATTERNS
================================================================================
===============
"""
#===============================================================================
===============
#Pattern: Planes
#Author: Ethan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    alloff()
    #xy planes:
    xyplaneon(1)
    time.sleep(.5)
    xyplaneon(2)
    time.sleep(.5)
    xyplaneon(3)
    time.sleep(.5)
    xyplaneon(4)
    time.sleep(.5)
    xyplaneoff(4)
    time.sleep(.5)
    xyplaneoff(3)
    time.sleep(.5)
    xyplaneoff(2)
    time.sleep(.5)
```

```
xyplaneoff(1)
time.sleep(.5)
alloff()
#oscillate/flash planes:
for i in range(10):
    xyplaneon(2)
    xyplaneon(4)
    time.sleep(.1)
    alloff()
    xyplaneon(1)
    xyplaneon(3)
    time.sleep(.1)
    alloff()
#xz planes:
xzplaneon(1)
time.sleep(.5)
xzplaneon(2)
time.sleep(.5)
xzplaneon(3)
time.sleep(.5)
xzplaneon(4)
time.sleep(.5)
xzplaneoff(4)
time.sleep(.5)
xzplaneoff(3)
time.sleep(.5)
xzplaneoff(2)
time.sleep(.5)
xzplaneoff(1)
time.sleep(.5)
alloff()
#oscillate/flash planes:
for i in range(10):
    xzplaneon(2)
    xzplaneon(4)
    time.sleep(.1)
    alloff()
    xzplaneon(1)
    xzplaneon(3)
    time.sleep(.1)
    alloff()
#yz planes:
yzplaneon(1)
time.sleep(.5)
yzplaneon(2)
time.sleep(.5)
yzplaneon(3)
time.sleep(.5)
yzplaneon(4)
time.sleep(.5)
```

```
    yzplaneoff(4)
    time.sleep(.5)
    yzplaneoff(3)
    time.sleep(.5)
    yzplaneoff(2)
    time.sleep(.5)
    yzplaneoff(1)
    time.sleep(.5)
    alloff()
    #oscillate/flash planes:
    for i in range(10):
        yzplaneon(2)
        yzplaneon(4)
        time.sleep(.1)
        alloff()
        yzplaneon(1)
        yzplaneon(3)
        time.sleep(.1)
        alloff()
#================================================================================
===============
#Pattern: Cubes
#Author: Ethan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    alloff()
    #grow cube from (4,4,1)
    on(4,4,1)
    time.sleep(.5)
    on(4,3,1)
    on(3,3,1)
    on(3,4,1)
    on(4,3,2)
    on(3,3,2)
    on(3,4,2)
    on(4,4,2)
    time.sleep(.5)
    on(4,2,1)
    on(3,2,1)
    on(2,2,1)
    on(2,3,1)
    on(2,4,1)
    on(4,2,2)
    on(3,2,2)
    on(2,2,2)
    on(2,3,2)
    on(2,4,2)
    on(4,2,3)
    on(3,2,3)
    on(2,2,3)
    on(2,3,3)
```

```
on(2,4,3)
time.sleep(.5)
allon()
time.sleep(.5)
alloff()
#grow cube from (1,1,4)
on(1,1,4)
time.sleep(.5)
on(1,2,4)
on(2,2,4)
on(2,1,4)
on(1,1,3)
on(1,2,3)
on(2,2,3)
on(2,1,3)
time.sleep(.5)
on(1,1,2)
on(1,2,2)
on(1,3,2)
on(2,1,2)
on(2,2,2)
on(2,3,2)
on(3,1,2)
on(3,2,2)
on(3,3,2)
on(3,1,3)
on(3,2,3)
on(3,3,3)
on(2,3,3)
on(1,3,3)
on(3,1,4)
on(3,2,4)
on(3,3,4)
on(2,3,4)
on(1,3,4)
time.sleep(.5)
allon()
time.sleep(.5)
alloff()
for i in range(4):
    allon()
    time.sleep(.1)
    alloff()
    on(2,2,2)
    on(3,2,2)
    on(2,3,2)
    on(3,3,2)
    on(2,2,3)
    on(3,2,3)
    on(2,3,3)
```

```
        on(3,3,3)
        time.sleep(.1)
    alloff()
#================================================================================
==============
#Pattern: Lines
#Author: Ethan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    alloff()
    alltrans()
    #actual code:
    on(1,1,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(2,1,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(3,1,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(4,1,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(4,2,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(3,2,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(2,2,1)
    time.sleep(.1)
    alloff()

    alltrans()
    on(1,2,1)
    time.sleep(.1)
    alloff()
```

```
alltrans()
on(1,3,1)
time.sleep(.1)
alloff()

alltrans()
on(2,3,1)
time.sleep(.1)
alloff()

alltrans()
on(3,3,1)
time.sleep(.1)
alloff()

alltrans()
on(4,3,1)
time.sleep(.1)
alloff()

alltrans()
on(4,4,1)
time.sleep(.1)
alloff()

alltrans()
on(3,4,1)
time.sleep(.1)
alloff()

alltrans()
on(2,4,1)
time.sleep(.1)
alloff()

alltrans()
on(1,4,1)
time.sleep(.1)
alloff()

#next part of pattern:
alltrans()
on(1,4,1)
time.sleep(.1)
alloff()

alltrans()
on(2,4,1)
time.sleep(.1)
alloff()
```

```
alltrans()
on(3,4,1)
time.sleep(.1)
alloff()

alltrans()
on(4,4,1)
time.sleep(.1)
alloff()

alltrans()
on(4,3,1)
time.sleep(.1)
alloff()

alltrans()
on(4,2,1)
time.sleep(.1)
alloff()

alltrans()
on(4,1,1)
time.sleep(.1)
alloff()

alltrans()
on(3,1,1)
time.sleep(.1)
alloff()

alltrans()
on(2,1,1)
time.sleep(.1)
alloff()

alltrans()
on(1,1,1)
time.sleep(.1)
alloff()

alltrans()
on(1,2,1)
time.sleep(.1)
alloff()

alltrans()
on(1,3,1)
time.sleep(.1)
alloff()
```

```
alltrans()
on(2,3,1)
time.sleep(.1)
alloff()

alltrans()
on(3,3,1)
time.sleep(.1)
alloff()

alltrans()
on(3,2,1)
time.sleep(.1)
alloff()

alltrans()
on(2,2,1)
time.sleep(.1)
alloff()

alltrans()
on(2,2,1)
time.sleep(.1)

alltrans()
on(3,2,1)
time.sleep(.1)

alltrans()
on(3,3,1)
time.sleep(.1)

alltrans()
on(2,3,1)
time.sleep(.1)

GPIO.output(2,False)
GPIO.output(3,True)
time.sleep(.1)

GPIO.output(3,False)
GPIO.output(4,True)
time.sleep(.1)

GPIO.output(4,False)
GPIO.output(14,True)
time.sleep(.1)

GPIO.output(14,False)
```

```
GPIO.output(27,True)
time.sleep(.1)

GPIO.output(27,False)
GPIO.output(10,True)
time.sleep(.1)

GPIO.output(10,False)
GPIO.output(7,True)
time.sleep(.1)

GPIO.output(7,False)
GPIO.output(8,True)
time.sleep(.1)

GPIO.output(8,False)
GPIO.output(25,True)
time.sleep(.1)

GPIO.output(25,False)
GPIO.output(9,True)
time.sleep(.1)

GPIO.output(9,False)
GPIO.output(22,True)
time.sleep(.1)

GPIO.output(22,False)
GPIO.output(15,True)
time.sleep(.1)

GPIO.output(15,False)
GPIO.output(18,True)
time.sleep(.1)

GPIO.output(18,False)
GPIO.output(17,True)
time.sleep(.1)

GPIO.output(17,False)
GPIO.output(24,True)
time.sleep(.1)

GPIO.output(24,False)
GPIO.output(23,True)
time.sleep(.1)

GPIO.output(23,False)
GPIO.output(24,True)
time.sleep(.1)
```

```
GPIO.output(24,False)
GPIO.output(17,True)
time.sleep(.1)

GPIO.output(17,False)
GPIO.output(18,True)
time.sleep(.1)

GPIO.output(18,False)
GPIO.output(15,True)
time.sleep(.1)

GPIO.output(15,False)
GPIO.output(22,True)
time.sleep(.1)

GPIO.output(22,False)
GPIO.output(9,True)
time.sleep(.1)

GPIO.output(9,False)
GPIO.output(25,True)
time.sleep(.1)

GPIO.output(25,False)
GPIO.output(8,True)
time.sleep(.1)

GPIO.output(8,False)
GPIO.output(7,True)
time.sleep(.1)

GPIO.output(7,False)
GPIO.output(10,True)
time.sleep(.1)

GPIO.output(10,False)
GPIO.output(27,True)
time.sleep(.1)

GPIO.output(27,False)
GPIO.output(14,True)
time.sleep(.1)

GPIO.output(14,False)
GPIO.output(4,True)
time.sleep(.1)
#LED 3 off, LED 2 on
GPIO.output(4,False)
```

```
GPIO.output(3,True)
time.sleep(.1)
#LED 2 off, LED 1 on
GPIO.output(3,False)
GPIO.output(2,True)
time.sleep(.1)
#LED 1 off
GPIO.output(2,False)
#all transistors off
GPIO.output(16,False)
GPIO.output(26,False)
GPIO.output(20,False)
GPIO.output(21,False)
alloff()

#next part:
allon()
off(3,4,1)
time.sleep(.1)
off(2,2,1)
time.sleep(.1)
off(1,4,1)
time.sleep(.1)
off(4,1,1)
time.sleep(.1)
off(1,1,1)
time.sleep(.1)
off(4,4,1)
time.sleep(.1)
off(2,4,1)
time.sleep(.1)
off(4,3,1)
time.sleep(.1)
off(1,3,1)
time.sleep(.1)
off(3,2,1)
time.sleep(.1)
off(2,1,1)
time.sleep(.1)
off(4,2,1)
time.sleep(.1)
off(1,2,1)
time.sleep(.1)
off(2,3,1)
time.sleep(.1)
off(3,1,1)
time.sleep(.1)
off(3,3,1)
time.sleep(.1)
alloff()
```

```
    allon()
    time.sleep(.1)
    alloff()
#================================================================================
==============
#Pattern: Spiral
#Author: Ethan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    #layer 1:
    on(1,1,1)
    time.sleep(.1)
    on(2,1,1)
    time.sleep(.1)
    on(3,1,1)
    time.sleep(.1)
    on(4,1,1)
    time.sleep(.1)
    on(4,2,1)
    time.sleep(.1)
    on(4,3,1)
    time.sleep(.1)
    on(4,4,1)
    time.sleep(.1)
    on(3,4,1)
    time.sleep(.1)
    on(2,4,1)
    time.sleep(.1)
    on(1,4,1)
    time.sleep(.1)
    on(1,3,1)
    time.sleep(.1)
    on(1,2,1)
    time.sleep(.1)
    on(2,2,1)
    time.sleep(.1)
    on(3,2,1)
    time.sleep(.1)
    on(3,3,1)
    time.sleep(.1)
    on(2,3,1)
    time.sleep(.1)
    alloff()
    #layer 2:
    on(1,1,2)
    time.sleep(.1)
    on(2,1,2)
    time.sleep(.1)
    on(3,1,2)
    time.sleep(.1)
    on(4,1,2)
```

```
time.sleep(.1)
on(4,2,2)
time.sleep(.1)
on(4,3,2)
time.sleep(.1)
on(4,4,2)
time.sleep(.1)
on(3,4,2)
time.sleep(.1)
on(2,4,2)
time.sleep(.1)
on(1,4,2)
time.sleep(.1)
on(1,3,2)
time.sleep(.1)
on(1,2,2)
time.sleep(.1)
on(2,2,2)
time.sleep(.1)
on(3,2,2)
time.sleep(.1)
on(3,3,2)
time.sleep(.1)
on(2,3,2)
time.sleep(.1)
alloff()
#layer 3:
on(1,1,3)
time.sleep(.1)
on(2,1,3)
time.sleep(.1)
on(3,1,3)
time.sleep(.1)
on(4,1,3)
time.sleep(.1)
on(4,2,3)
time.sleep(.1)
on(4,3,3)
time.sleep(.1)
on(4,4,3)
time.sleep(.1)
on(3,4,3)
time.sleep(.1)
on(2,4,3)
time.sleep(.1)
on(1,4,3)
time.sleep(.1)
on(1,3,3)
time.sleep(.1)
on(1,2,3)
```

```
time.sleep(.1)
on(2,2,3)
time.sleep(.1)
on(3,2,3)
time.sleep(.1)
on(3,3,3)
time.sleep(.1)
on(2,3,3)
time.sleep(.1)
alloff()
#layer 4:
on(1,1,4)
time.sleep(.1)
on(2,1,4)
time.sleep(.1)
on(3,1,4)
time.sleep(.1)
on(4,1,4)
time.sleep(.1)
on(4,2,4)
time.sleep(.1)
on(4,3,4)
time.sleep(.1)
on(4,4,4)
time.sleep(.1)
on(3,4,4)
time.sleep(.1)
on(2,4,4)
time.sleep(.1)
on(1,4,4)
time.sleep(.1)
on(1,3,4)
time.sleep(.1)
on(1,2,4)
time.sleep(.1)
on(2,2,4)
time.sleep(.1)
on(3,2,4)
time.sleep(.1)
on(3,3,4)
time.sleep(.1)
on(2,3,4)
time.sleep(.1)
alloff()
on(2,3,4)
time.sleep(.1)
alloff()
on(2,3,3)
time.sleep(.1)
alloff()
```

```
    on(2,3,2)
    time.sleep(.1)
    on(2,3,1)
    time.sleep(.1)
    xyplaneon(1)
    time.sleep(.1)
    xyplaneon(2)
    time.sleep(.1)
    xyplaneon(3)
    time.sleep(.1)
    xyplaneon(4)
    time.sleep(.1)
    alloff()
#================================================================================
==============
#Pattern: Teleport
#Author: Dan
for i in range(100):
    a=random.randint(1,4)
    b=random.randint(1,4)
    c=random.randint(1,4)
    d=random.randint(1,4)
    e=random.randint(1,4)
    f=random.randint(1,4)
    on(a,b,c)
    on(d,e,f)
    time.sleep(.1)
    alloff()
#================================================================================
==============
#Pattern: Pivot
#Author: Dan
#================================================================================
==============
#Pattern:
#Author: Dan



#================================================================================
==============
#Pattern: Rain
#Author: Ryan
for i in range(50): #For loop here to let us minimize this pattern in Spyder.

    a=random.randint(1,4) #rain runs 50 time
    b=random.randint(1,4) #a and b decide the coord for rain

    on(a,b,4)
    time.sleep(.075)  #drop starts, turns of after time delay
```

```
    xyplaneoff(4)

    on(a,b,3)
    time.sleep(.075)   #second appearance,  turns of after time delay
    xyplaneoff(3)

    on(a,b,2)
    time.sleep(.075)  #thrid appearance,  turns of after time delay
    xyplaneoff(2)

    on(a,b,1)
    time.sleep(.075)   #bottom appearance,  turns of after time delay
    xyplaneoff(1)

    alloff()
#================================================================================
===============
#Pattern: Chess
#Author: Ryan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    xyplaneon(1)     #bottom plane on

    GPIO.output(15,False)
    GPIO.output(18,False)
    GPIO.output(17,False)
    GPIO.output(27,False)   #turns off middle rows of xy plane
    GPIO.output(22,False)
    GPIO.output(23,False)
    GPIO.output(24,False)
    GPIO.output(10,False)

    time.sleep(.1)
    on(1,2,1)       #pawn(1,1) moves forward
    off(1,1,1)

    time.sleep(.5)
    on(3,3,1)          #pawn(3,4) moves forward
    off(3,4,1)

    time.sleep(.5)
    on(2,3,1)         #pawn(2,4) moves forward
    off(2,4,1)

    time.sleep(.5)
    off(1,2,1)

    time.sleep(.5)   #pawn(1,2) takes pawn at (2,3)
    off(2,3,1)       #blinks to show take
    time.sleep(.5)
    on(2,3,1)
```

```
time.sleep(.5)
off(2,3,1)
time.sleep(.5)
on(2,3,1)

time.sleep(.5)
off(1,4,1)

time.sleep(.5)  #pawn taken at (2,3)
off(2,3,1)
time.sleep(.5)
on(2,3,1)
time.sleep(.5)
off(2,3,1)
time.sleep(.5)
on(2,3,1)

time.sleep(.5) #pawn(3,1) moves to (3,2)
off(3,1,1)
on(3,2,1)

time.sleep(.5)
off(2,3,1)

time.sleep(.5)  #pawn taken at (3,2)
off(3,2,1)
time.sleep(.5)
on(3,2,1)
time.sleep(.5)
off(3,2,1)
time.sleep(.5)
on(3,2,1)

time.sleep(.5)
off(2,1,1)

time.sleep(.5)  #pawn taken at (3,2)
off(3,2,1)
time.sleep(.5)
on(3,2,1)
time.sleep(.5)
off(3,2,1)
time.sleep(.5)
on(3,2,1)

time.sleep(.5) #pawn(4,4) moves to (4,3)
off(4,4,1)
on(4,3,1)

time.sleep(.5) #pawn(4,1) moves to (4,2)
```

```
off(4,1,1)
on(4,2,1)

time.sleep(.5)
off(3,3,1)

time.sleep(.5)  #pawn taken at (4,2)
off(4,2,1)
time.sleep(.5)
on(4,2,1)
time.sleep(.5)
off(4,2,1)
time.sleep(.5)
on(4,2,1)

time.sleep(.5)
off(3,2,1)

time.sleep(.5)  #pawn taken at (4,3)
off(4,3,1)
time.sleep(.5)
on(4,3,1)
time.sleep(.5)
off(4,3,1)
time.sleep(.5)
on(4,3,1)

time.sleep(.5) #pawn(4,2) moves to (4,1)
off(4,2,1)
on(4,1,1)

time.sleep(.25)  #pawn(4,1) promote to queen
off(4,1,1)       #quicker flash for queen promotion
time.sleep(.25)
on(4,1,1)
time.sleep(.25)
off(4,1,1)
time.sleep(.25)
on(4,1,1)

time.sleep(.5)  #pawn(4,3) move to (4,4)
off(4,3,1)
on(4,4,1)

time.sleep(.25)  #pawn(4,4) promote to queen
off(4,4,1)
time.sleep(.25)
on(4,4,1)
time.sleep(.25)
off(4,4,1)
```

```
time.sleep(.25)
on(4,4,1)

time.sleep(.5)
off(4,1,1)

time.sleep(.5) #queen(4,1) takes queen(4,4)
off(4,4,1)      #flash quickens to explosion to indicate end of game
time.sleep(.45)
on(4,4,1)
time.sleep(.4)
off(4,4,1)
time.sleep(.35)
on(4,4,1)
time.sleep(.3)
off(4,4,1)
time.sleep(.25)
on(4,4,1)
time.sleep(.2)
off(4,4,1)
time.sleep(.15)
on(4,4,1)

time.sleep(.5)   #cubes use to make explosion
on(4,3,1)
on(3,3,1)
on(3,4,1)
on(4,3,2)
on(3,3,2)
on(3,4,2)
on(4,4,2)
time.sleep(.5)
on(4,2,1)
on(3,2,1)
on(2,2,1)
on(2,3,1)
on(2,4,1)
on(4,2,2)
on(3,2,2)
on(2,2,2)
on(2,3,2)
on(2,4,2)
on(4,2,3)
on(3,2,3)
on(2,2,3)
on(2,3,3)
on(2,4,3)
time.sleep(.5)
allon()
time.sleep(.5) #all off after event
```

```
    alloff()
#================================================================================
==============
#Pattern: Frogger/ Back and forth
#Author: Ryan
for i in range(1): #For loop here to let us minimize this pattern in Spyder.
    time.sleep(.15)
    for i in range(3): #layer one start with dot on either side staggerered
        on(1,1,1)
        on(2,4,1)
        on(3,1,1)
        on(4,4,1)

        time.sleep(.15)

        off(1,1,1)
        off(2,4,1)
        off(3,1,1)
        off(4,4,1)   #first move

        on(1,2,1)
        on(2,3,1)
        on(3,2,1)
        on(4,3,1)

        time.sleep(.15)

        off(1,2,1)
        off(2,3,1)
        off(3,2,1)
        off(4,3,1)   #second move

        on(1,3,1)
        on(2,2,1)
        on(3,3,1)
        on(4,2,1)

        time.sleep(.15)

        off(1,3,1)
        off(2,2,1)
        off(3,3,1)
        off(4,2,1)   #dots have move to opposite side

        on(1,4,1)
        on(2,1,1)
        on(3,4,1)
        on(4,1,1)

        time.sleep(.15)
```

```
    off(1,4,1)
    off(2,1,1)
    off(3,4,1)
    off(4,1,1)  #first move back

    on(1,3,1)
    on(2,2,1)
    on(3,3,1)
    on(4,2,1)

    time.sleep(.15)

    off(1,3,1)
    off(2,2,1)
    off(3,3,1)
    off(4,2,1)  #second move back

    on(1,2,1)
    on(2,3,1)
    on(3,2,1)
    on(4,3,1)

    time.sleep(.15)

    off(1,2,1)
    off(2,3,1)
    off(3,2,1)
    off(4,3,1)  #dots have returned to original position

    on(1,1,1)
    on(2,4,1)
    on(3,1,1)
    on(4,4,1)

    time.sleep(.15)  #cycle repeats three time

    off(1,1,1)
    off(2,4,1)
    off(3,1,1)
    off(4,4,1)

for i in range(3):

    on(4,1,2)
    on(1,2,2)   #all tensitors on, dots form towers on second layer
    on(4,3,2)   #first position
    on(1,4,2)

    time.sleep(.15)
```

```
off(4,1,2)
off(1,2,2)
off(4,3,2)
off(1,4,2)    #first move

on(3,1,2)
on(2,2,2)
on(3,3,2)
on(2,4,2)

time.sleep(.15)

off(3,1,2)
off(2,2,2)
off(3,3,2)
off(2,4,2)  #second move

on(2,1,2)
on(3,2,2)
on(2,3,2)
on(3,4,2)

time.sleep(.15)

off(2,1,2)
off(3,2,2)
off(2,3,2)
off(3,4,2)  #towers on opposite side

on(1,1,2)
on(4,2,2)
on(1,3,2)
on(4,4,2)

time.sleep(.15)

off(1,1,2)
off(4,2,2)
off(1,3,2)
off(4,4,2) #first move back

on(2,1,2)
on(3,2,2)
on(2,3,2)
on(3,4,2)

time.sleep(.15)

off(2,1,2)
```

```
    off(3,2,2)
    off(2,3,2)
    off(3,4,2) #second move back

    on(3,1,2)
    on(2,2,2)
    on(3,3,2)
    on(2,4,2)

    time.sleep(.15)

    off(3,1,2)
    off(2,2,2)
    off(3,3,2)
    off(2,4,2)  #towers in orignal position

    on(4,1,2)
    on(1,2,2)
    on(4,3,2)
    on(1,4,2)

    time.sleep(.15)  #repeats three times

    off(4,1,2)
    off(1,2,2)
    off(4,3,2)
    off(1,4,2)

for i in range(3):
    on(1,1,3)
    on(2,4,3) #next layer, towers grow,original position
    on(3,1,3)
    on(4,4,1)

    time.sleep(.15)

    off(1,1,3)
    off(2,4,3)
    off(3,1,3)
    off(4,4,3) #first move

    on(1,2,3)
    on(2,3,3)
    on(3,2,3)
    on(4,3,3)

    time.sleep(.15)

    off(1,2,3)
    off(2,3,3)
```

```
off(3,2,3)
off(4,3,3)  #second move

on(1,3,3)
on(2,2,3)
on(3,3,3)
on(4,2,3)

time.sleep(.15)

off(1,3,3)
off(2,2,3)
off(3,3,3)
off(4,2,3)  #towers on opposite side

on(1,4,3)
on(2,1,3)
on(3,4,3)
on(4,1,3)

time.sleep(.15)

off(1,4,3)
off(2,1,3)
off(3,4,3)
off(4,1,3) #first move back

on(1,3,3)
on(2,2,3)
on(3,3,3)
on(4,2,3)

time.sleep(.15)

off(1,3,3)
off(2,2,3)
off(3,3,3)
off(4,2,3)  # second move back

on(1,2,3)
on(2,3,3)
on(3,2,3)
on(4,3,3)

time.sleep(.15)

off(1,2,3)
off(2,3,3)
off(3,2,3)
off(4,3,3)  # towers return to original position
```

```
    on(1,1,3)
    on(2,4,3)
    on(3,1,3)
    on(4,4,3)

    time.sleep(.15) #repeats three time

    off(1,1,3)
    off(2,4,3)
    off(3,1,3)
    off(4,4,3)


for i in range(3):

    on(4,1,4)
    on(1,2,4)  #towers at full height, original position
    on(4,3,4)
    on(1,4,4)

    time.sleep(.15)

    off(4,1,4)
    off(1,2,4)
    off(4,3,4)
    off(1,4,4)  #first move

    on(3,1,4)
    on(2,2,4)
    on(3,3,4)
    on(2,4,4)

    time.sleep(.15)

    off(3,1,4)
    off(2,2,4)
    off(3,3,4)
    off(2,4,4)  #seconcd move

    on(2,1,4)
    on(3,2,4)
    on(2,3,4)
    on(3,4,4)

    time.sleep(.15)

    off(2,1,4)
    off(3,2,4)
    off(2,3,4)
```

```
    off(3,4,4)  #towers at opposite side

    on(1,1,4)
    on(4,2,4)
    on(1,3,4)
    on(4,4,4)

    time.sleep(.15)

    off(1,1,4)
    off(4,2,4)
    off(1,3,4)
    off(4,4,4)  #first move back

    on(2,1,4)
    on(3,2,4)
    on(2,3,4)
    on(3,4,4)

    time.sleep(.15)

    off(2,1,4)
    off(3,2,4)
    off(2,3,4)
    off(3,4,4) #second move back

    on(3,1,4)
    on(2,2,4)
    on(3,3,4)
    on(2,4,4)

    time.sleep(.15)

    off(3,1,4)
    off(2,2,4)
    off(3,3,4)
    off(2,4,4) #tower in original position

    on(4,1,4)
    on(1,2,4)
    on(4,3,4)
    on(1,4,4)

    time.sleep(.15)  #repeats three times

alloff()   #end of pattern
```