PYTHON PROJECT
FIBONACCI CLOCK

ABOUT PROJECT

The goal is to create a machine that displays the time using only the numbers available from the Fibonacci sequence. This will be achieved using a Python script to pull current time from the computer clock and display a light sequence based on the figures. A Raspberry Pi will be initially used as the base computer due to size and convenience.

ABOUT PYTHON

Python is a high-level, general-purpose programming language. To create Python, the use of a interpreter is required, which is like a basic text editor that can mimic the running of a script.

Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, an open source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

ABOUT FIBONACCI

In mathematics, the Fibonacci numbers, commonly denoted Fn form a sequence, called the Fibonacci sequence, such that each number is the sum of the two preceding ones, starting from 0 and 1.

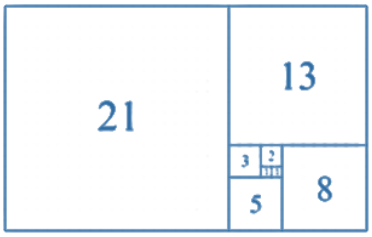
2 + 3 = 5

3 + 5 = 8

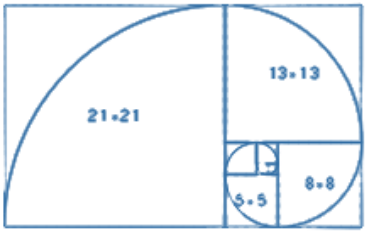
5 + 8 = 13

8 + 13 = 21

A tiling with squares whose side lengths are successive Fibonacci numbers: 1, 1, 2, 3, 5, 8, 13 and 21



The Fibonacci spiral: an approximation of the golden spiral created by drawing circular arcs connecting the opposite corners of squares in the Fibonacci tiling



FIBONACCI CODE Part One

Firstly the time is retrieved and converted to usable figures –

```
while 1:
 t = datetime.datetime.now()
 hr = t.hour
 mn = t.minute

 if (hr==00) or (hr==12):
     color_of_11 = red
     hr = 12
 else:
     color_of_11 = white
 mn5 = mn%5
 if (mn5 == 0):
    mn = mn/5
 else:
    mn = mn-mn5
    mn = mn/5
 if(hr > 12):
    hr = hr-12
 d1 = hr
 d2 = mn
 hr_factors = []
 min_factors = []

 hour_factors(d1)
 minute_factors(d2)
```

FIBONACCI CODE Part Two

Now the fun part, the number allocating -

```
num_list = [5,3,2,1,1]

def hour_factors(d1):

    j = 0
    while(d1 != 0):
        if(d1 >= num_list[j]):
            d1 = d1-num_list[j]
            hr_factors.append(num_list[j])
        j+=1

def minute_factors(d2):

         j = 0
         while(d2 != 0):
            if(d2 >= num_list[j]):
                d2 = d2-num_list[j]
                min_factors.append(num_list[j])

            j+=1
```

Basically, the num_list is your sequence and 'j' is which number in that sequence is being tested. In the case of hour_factors, it'll compare the hour to each number in sequence until no remainder is left each time appending the found numbers to a new list – hr.factors

FIBONACCI CODE Part Three

All that remains is to open the required GIPO pins and associate them to each requirement –

```
import RPi.GPIO as GPIO # Import Raspberry Pi GPIO library
GPIO.setwarnings(False) # Ignore warning for now
GPIO.setmode(GPIO.BOARD) # Use physical pin numbering

GPIO.setup(7, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(11, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(12, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(13, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(15, GPIO.OUT, initial=GPIO.LOW)
```

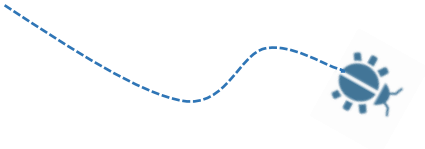
….. so on

```
#BULB_1

def bulb1Red():
    GPIO.output(7, GPIO.HIGH)
    GPIO.output(11, GPIO.LOW)
    GPIO.output(12, GPIO.LOW)

def bulb1Green():
    GPIO.output(7, GPIO.LOW)
    GPIO.output(11, GPIO.HIGH)
    GPIO.output(12, GPIO.LOW)

def bulb1Blue():
    GPIO.output(7, GPIO.LOW)
    GPIO.output(11, GPIO.LOW)
    GPIO.output(12, GPIO.HIGH)

def bulb1White():
    GPIO.output(7, GPIO.HIGH)
    GPIO.output(11, GPIO.HIGH)
    GPIO.output(12, GPIO.HIGH)
```

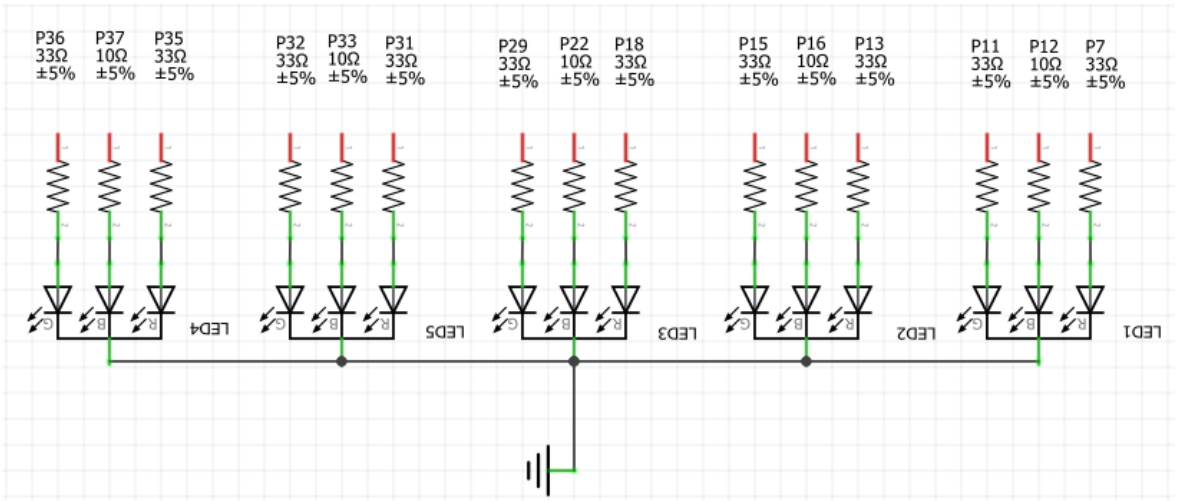
….. so on

```
def assign_colors():
 if (5 in hr_factors) and (5 in min_factors):
  bulb5Blue()
 elif (5 in hr_factors):
  bulb5Red()
 elif (5 in min_factors):
  bulb5Green()
 else:
  bulb5White()
```

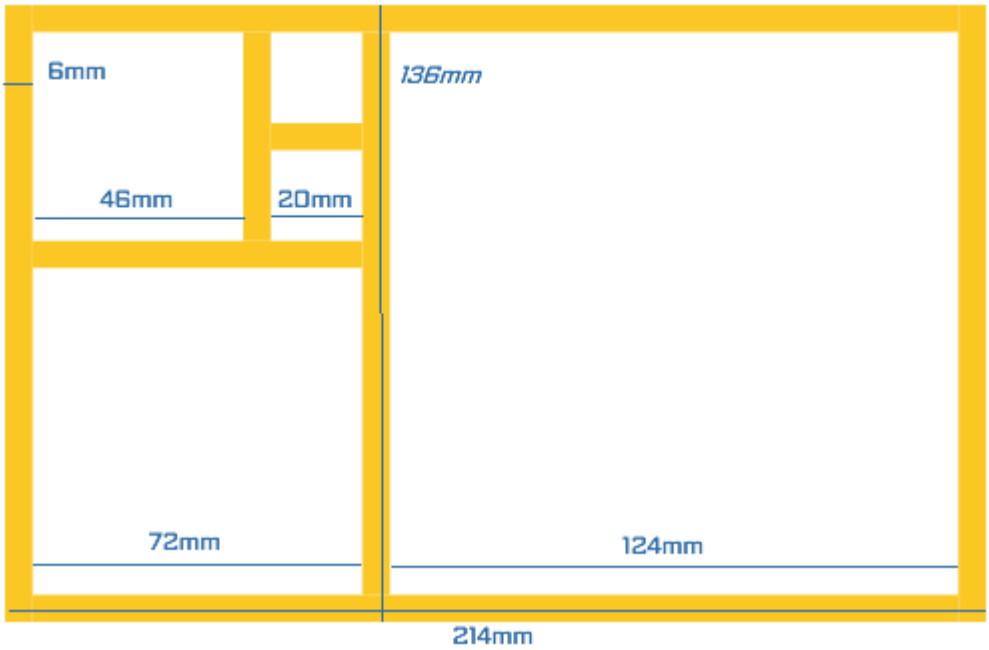
….. so on



ELECTRONICS



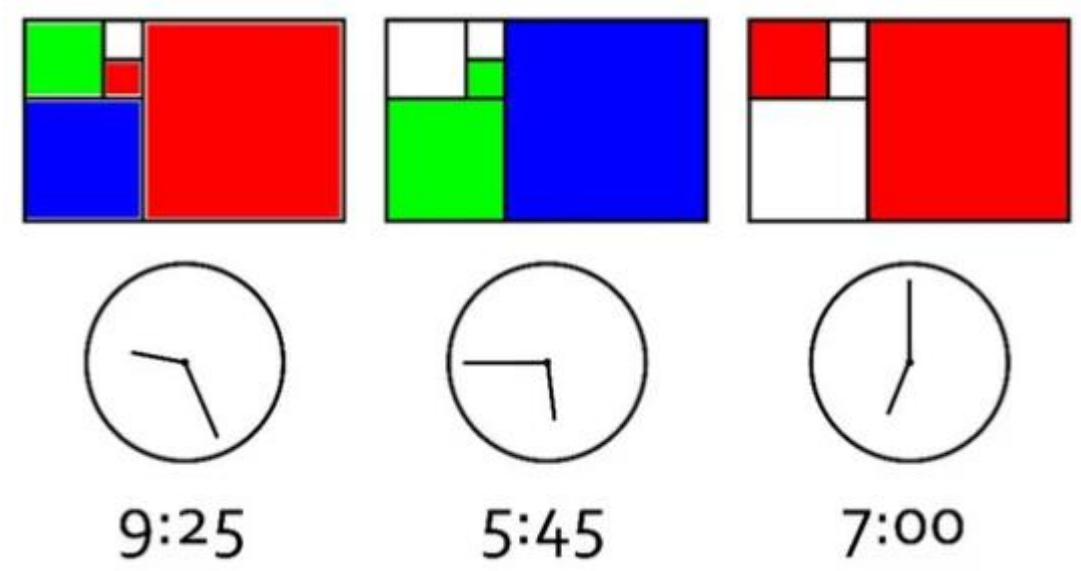
DIMENSIONS



HOW TO READ A FIBONACCI CLOCK

The screen of the clock is made up of five squares whose side lengths match the first five Fibonacci numbers: 1, 1, 2, 3 and 5. The hours are displayed using red and the minutes using green. When a square is used to display both the hours and minutes it turns blue. White squares are ignored. To tell time on the Fibonacci clock you need to do some math.

To read the hour, simply add up the corresponding values of the red and blue squares. To read the minutes, do the same with the green and blue squares. The minutes are displayed in 5 minute increments (0 to 12) so you have to multiply your result by 5 to get the actual number.



MICROPYTHON - PYBOARD

Additional: I did try using a PyBoard instead of the Raspberry Pi which uses MicroPython instead of Python. Alas due to the poor time keeping of the hardware, this was abandoned.

```
from pyb import RTC

P1 = Pin('X1', Pin.OUT_PP)
P2 = Pin('X2', Pin.OUT_PP)
P3 = Pin('X3', Pin.OUT_PP)
```

And so on…

```
def bulb1Red():
    P1.high()
    P2.low()
    P3.low()
```

And so on…

```
rtc = RTC()
rtc.datetime((2020, 2, 10, 1, 21, 34, 13, 67)
while 1:
 hr = rtc.datetime()[4]
 mn = rtc.datetime()[5]
```