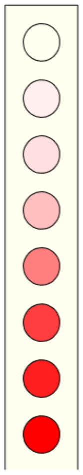
Starter Tree

Write a Python program so you can control a starter tree with your cell phone over a WiFi network.

- A NeoPixel acts as the starter tree (0 to 8 lights on)
- The Pico updates a web page so that you can see the status of the LEDs on your cell phone in real time.
- Your cell phone can send commands to the PICO to
  - Clear the lights (new race) and
  - Start a race (start turning on the lights)



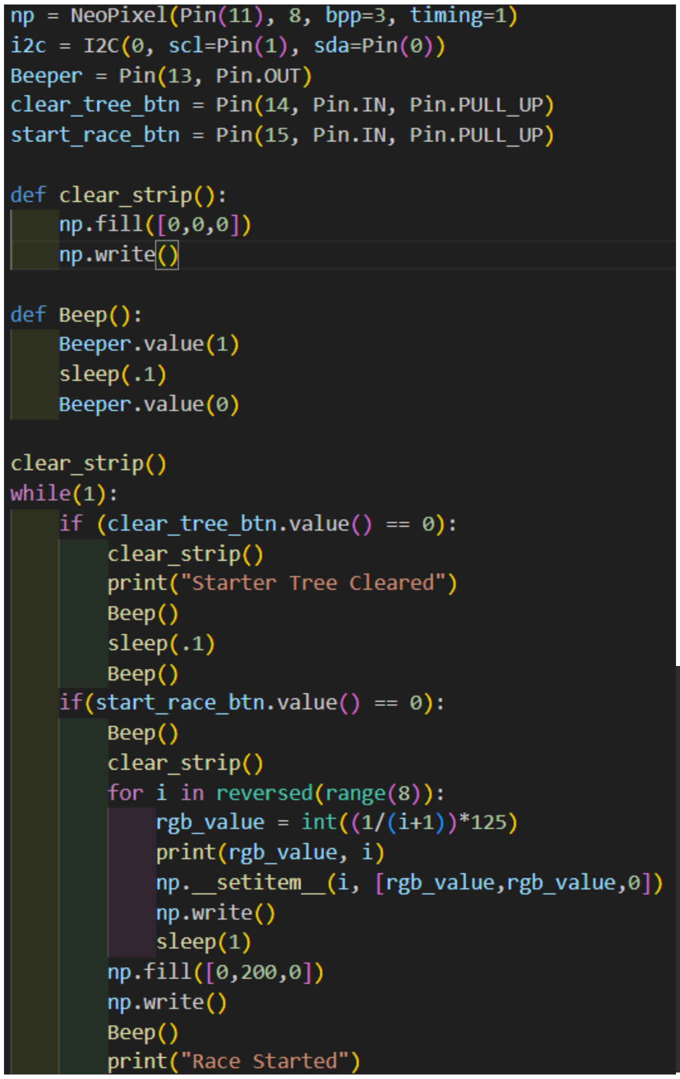
1) NeoPixel: Write a Pyton program so that the Pico controls the lights of a starter tree with push buttons:

- GP14: Clear the LEDs
- GP15: Start a race

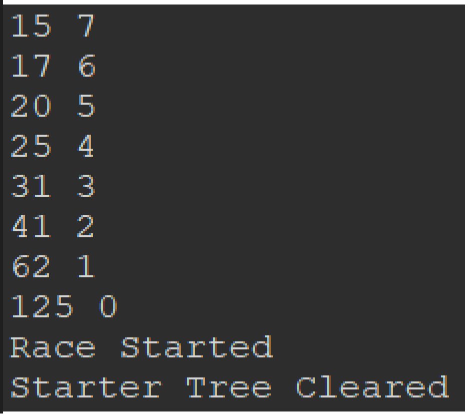
When a race is started,

- The LEDs turn on one at a time,
- Each LED turns on once every second
- Once all eight LEDs are on, they stay on (race has begun)

Verify your code works.



```
np = NeoPixel(Pin(11), 8, bpp=3, timing=1)
i2c = I2C(0, scl=Pin(1), sda=Pin(0))
Beeper = Pin(13, Pin.OUT)
clear_tree_btn = Pin(14, Pin.IN, Pin.PULL_UP)
start_race_btn = Pin(15, Pin.IN, Pin.PULL_UP)

def clear_strip():
    np.fill([0,0,0])
    np.write()

def Beep():
    Beeper.value(1)
    sleep(.1)
    Beeper.value(0)

clear_strip()
while(1):
    if (clear_tree_btn.value() == 0):
        clear_strip()
        print("Starter Tree Cleared")
        Beep()
        sleep(.1)
        Beep()
    if(start_race_btn.value() == 0):
        Beep()
        clear_strip()
        for i in reversed(range(8)):
            rgb_value = int((1/(i+1))*125)
            print(rgb_value, i)
            np.__setitem__(i, [rgb_value,rgb_value,0])
            np.write()
            sleep(1)
        np.fill([0,200,0])
        np.write()
        Beep()
        print("Race Started")
```



```
15 7
17 6
20 5
25 4
31 3
41 2
62 1
125 0
Race Started
Starter Tree Cleared
```

hw_11_1.py

I added a gradient feature so as the starter tree gets closer to 8, the light gets brighter (pictured above). Video demo in *HW11-1.mp4*.

2) Pico to Cell Phone over WiFi: Set up a web page so that you can see the status of the starter tree on your cell phone. It's your choice if you set up the PIC as a host or a client.

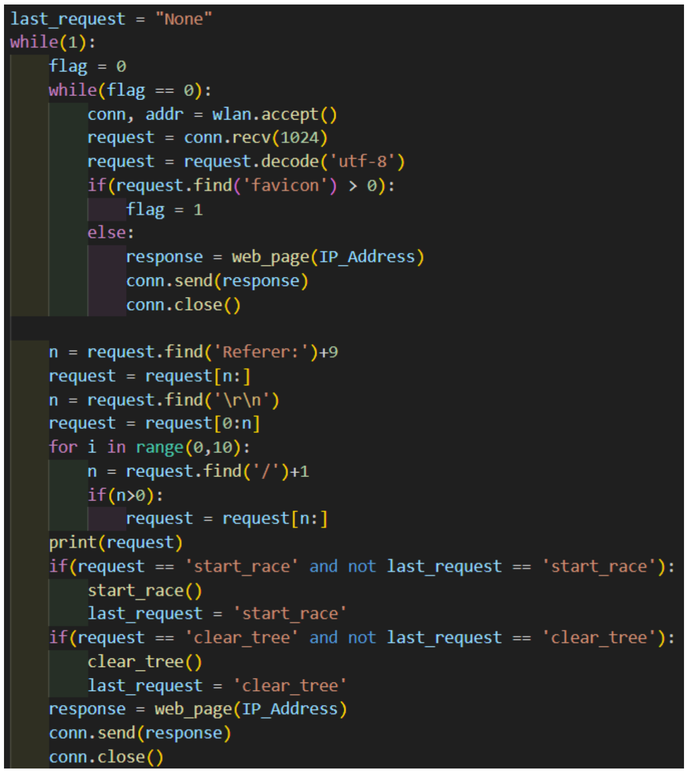
Verify that you can see the status of the starter tree in real time on your cell phone.

3) Cell Phone to Pico over WiFi: Add to the previous design a way for you to use your cell phone and the WiFi network to

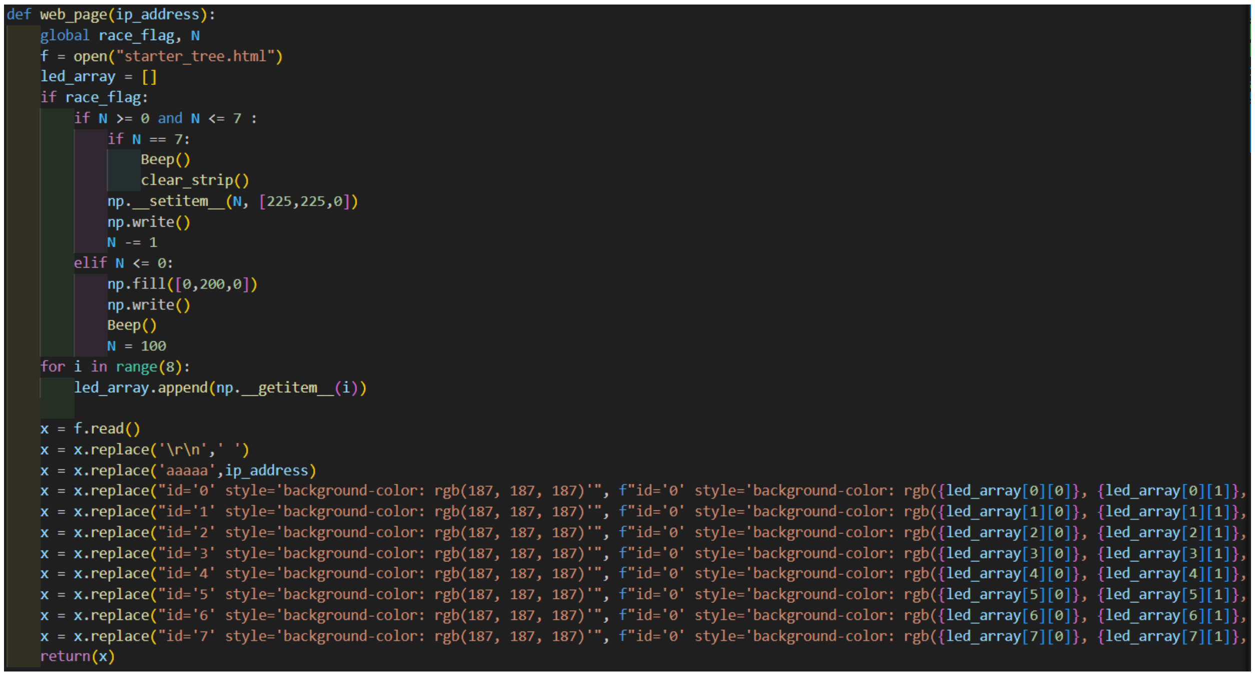
- Clear the starter tree (replacing button GP14), and
- Start a race (replacing button GP15)

Verify your code works.

I decided to jump the gun and combine parts 2 and 3 into *hw_11_2.py*:



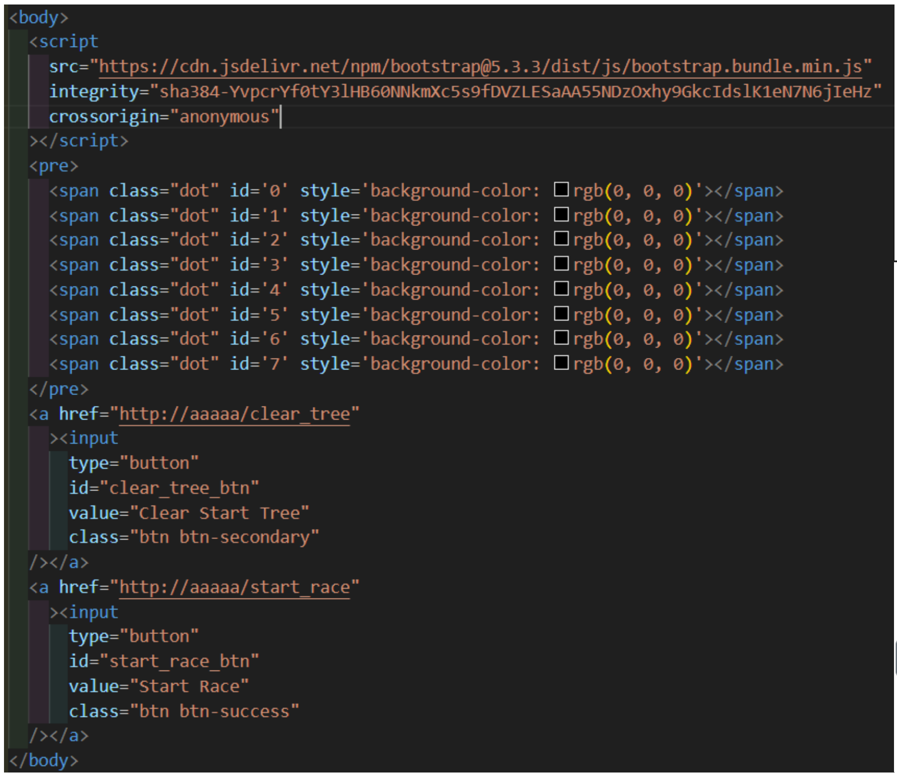
```python
last_request = "None"
while(1):
    flag = 0
    while(flag == 0):
        conn, addr = wlan.accept()
        request = conn.recv(1024)
        request = request.decode('utf-8')
        if(request.find('favicon') > 0):
            flag = 1
        else:
            response = web_page(IP_Address)
            conn.send(response)
            conn.close()

    n = request.find('Referer:')+9
    request = request[n:]
    n = request.find('\r\n')
    request = request[0:n]
    for i in range(0,10):
        n = request.find('/')+1
        if(n>0):
            request = request[n:]
    print(request)
    if(request == 'start_race' and not last_request == 'start_race'):
        start_race()
        last_request = 'start_race'
    if(request == 'clear_tree' and not last_request == 'clear_tree'):
        clear_tree()
        last_request = 'clear_tree'
    response = web_page(IP_Address)
    conn.send(response)
    conn.close()
```



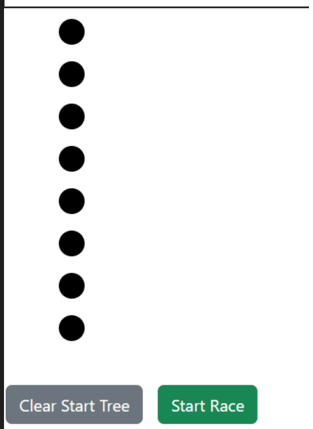
```python
def web_page(ip_address):
    global race_flag, N
    f = open("starter_tree.html")
    led_array = []
    if race_flag:
        if N >= 0 and N <= 7 :
            if N == 7:
                Beep()
                clear_strip()
            np.__setitem__(N, [225,225,0])
            np.write()
            N -= 1
        elif N <= 0:
            np.fill([0,200,0])
            np.write()
            Beep()
            N = 100
    for i in range(8):
        led_array.append(np.__getitem__(i))

    x = f.read()
    x = x.replace('\r\n',' ')
    x = x.replace('aaaaa',ip_address)
    x = x.replace("id='0' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[0][0]}, {led_array[0][1]},
    x = x.replace("id='1' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[1][0]}, {led_array[1][1]},
    x = x.replace("id='2' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[2][0]}, {led_array[2][1]},
    x = x.replace("id='3' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[3][0]}, {led_array[3][1]},
    x = x.replace("id='4' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[4][0]}, {led_array[4][1]},
    x = x.replace("id='5' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[5][0]}, {led_array[5][1]},
    x = x.replace("id='6' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[6][0]}, {led_array[6][1]},
    x = x.replace("id='7' style='background-color: rgb(187, 187, 187)'", f"id='0' style='background-color: rgb({led_array[7][0]}, {led_array[7][1]},
    return(x)
```

Note the updates to the main loop and *web_page(ip_address)*. I decided to integrate *start_race()* with *web_page()* for simplicity and timing sake, while keeping *clear_tree()* intact. I opted to update rgb values of background colors to show the changes in real time.

This is the body of *starter_tree.html*:



```html
<body>
  <script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
    crossorigin="anonymous"
  ></script>
  <pre>
    <span class="dot" id='0' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='1' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='2' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='3' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='4' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='5' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='6' style='background-color: rgb(0, 0, 0)'></span>
    <span class="dot" id='7' style='background-color: rgb(0, 0, 0)'></span>
  </pre>
  <a href="http://aaaaa/clear_tree"
    ><input
      type="button"
      id="clear_tree_btn"
      value="Clear Start Tree"
      class="btn btn-secondary"
  /></a>
  <a href="http://aaaaa/start_race"
    ><input
      type="button"
      id="start_race_btn"
      value="Start Race"
      class="btn btn-success"
  /></a>
</body>
```



I opted to use Bootstrap and a bit of inline styling to achieve the above website layout. In practice, the dots rendered in a line, which I can only think to blame the removal of the carriage returns and line feeds. I assumed that the **pre** tag would take care of that, but it still functions the same.

4) Demo your starter tree

- Video preferred

See HW11-2.mp4

I worked with Cole Rahne on this homework assignment.