Faroles de Independencia con CircuitPython

Telf/Fax: 2257 1015
Greencore Solutions SRL, San José, C
cursos@greencore.co.cr | www.greencore.co.cr

Agenda

- Faroles de Independencia
- Opcional: ¿Como emular la tarjeta?
- Rifa
- ¿Que es CircuitPython?
- ¿Que puedo hacer con una CircuitPlayground Express?
- Luces LED: Tradicionales e Inteligentes
- Matemática fractal para nuestro farol

Faroles de Independencia

- México, Centroamérica y Suramérica celebran independencia en estas fechas
- En algunos países usamos faroles o linternas, donde cada ciudadano porta una en marchas norcturnas
- Desde pequeño lo que se usa son candelas. Mis papás me hicieron uno eléctrico con una bombilla, una batería, cables y cinta eléctrica. Sin interruptor
- Queremos evitar quemaduras

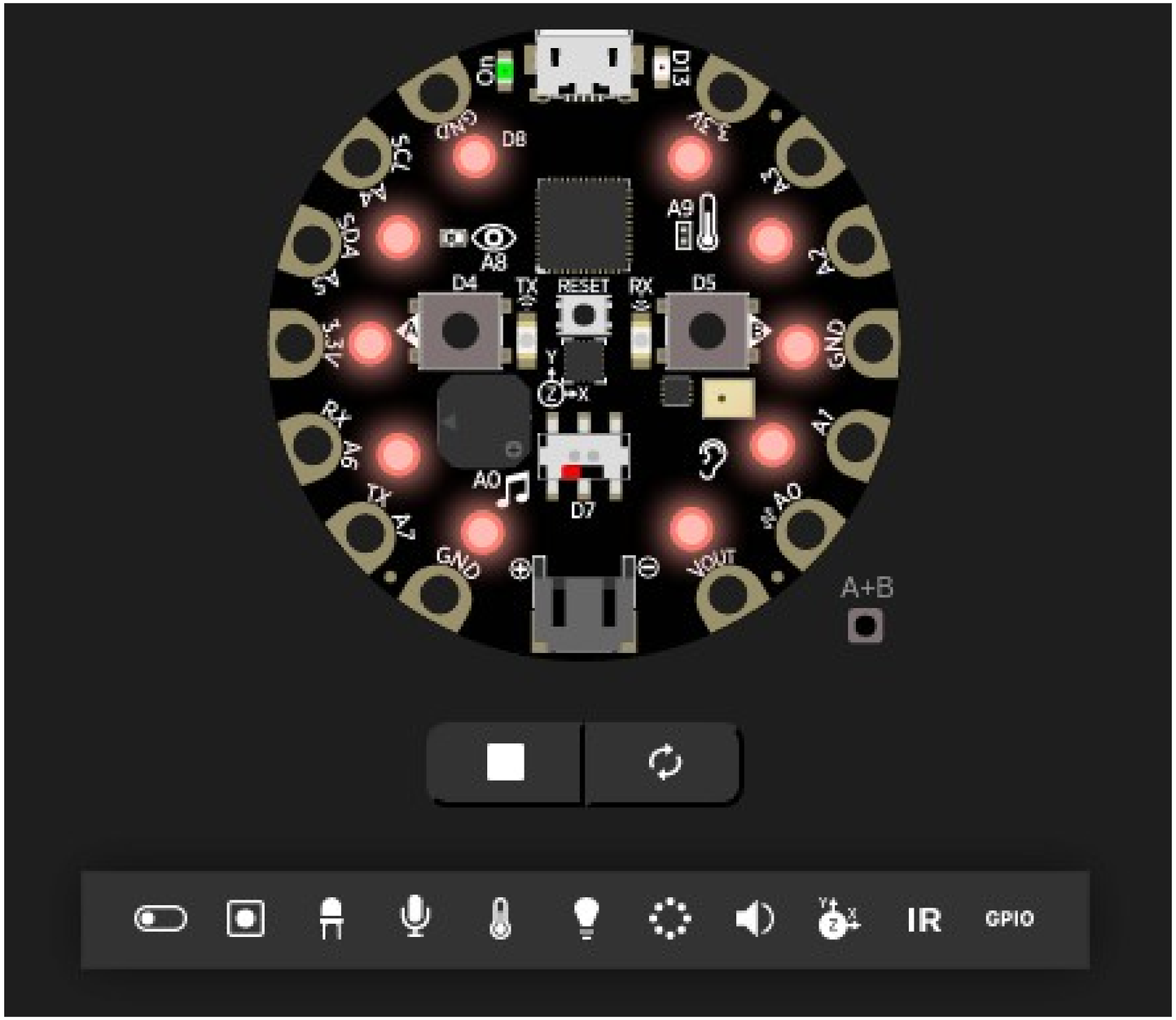
Faroles de Independencia

- Algunas ideas para construir
  - De madera
  - Comprarlo de papel
  - Reciclando botellas de bebidas (“PET”)
  - Con cajas para bebidas (“tetrabrik”)
  - Usar un difusor para las ventanas

Opcional: Como emular

- Instalar editor Visual Studio Code de Microsoft
- Instale el “Device Simulator Express” en este link
  - Ctrl+P
  - ext install ms-python.devicesimulatorexpress
  - Reinicia Visual Studio Code
  - Ctrl+Shift+P → Device Simulator Express New File

Opcional: Como emular



Rifa: CircuitPlayground Express

- Si no tienes tu tarjeta CircuitPlayground Express, y **resides en Costa Rica**, puedes participar en la rifa
- Nuestro aliado del evento, CRCibernética va a estar enviando, sin costo alguno, **2** tarjetas CircuitPlayground Express de Adafruit
- Para participar, envía un correo a info@diacircuitpython.org con:
  - Nombre, número de identificación
  - Número de teléfono
  - **Dirección exacta**

Rifa: CircuitPlayground Express

- **Para participar, debe enviar en el correo:**

Palabra código: Quetzalblinka

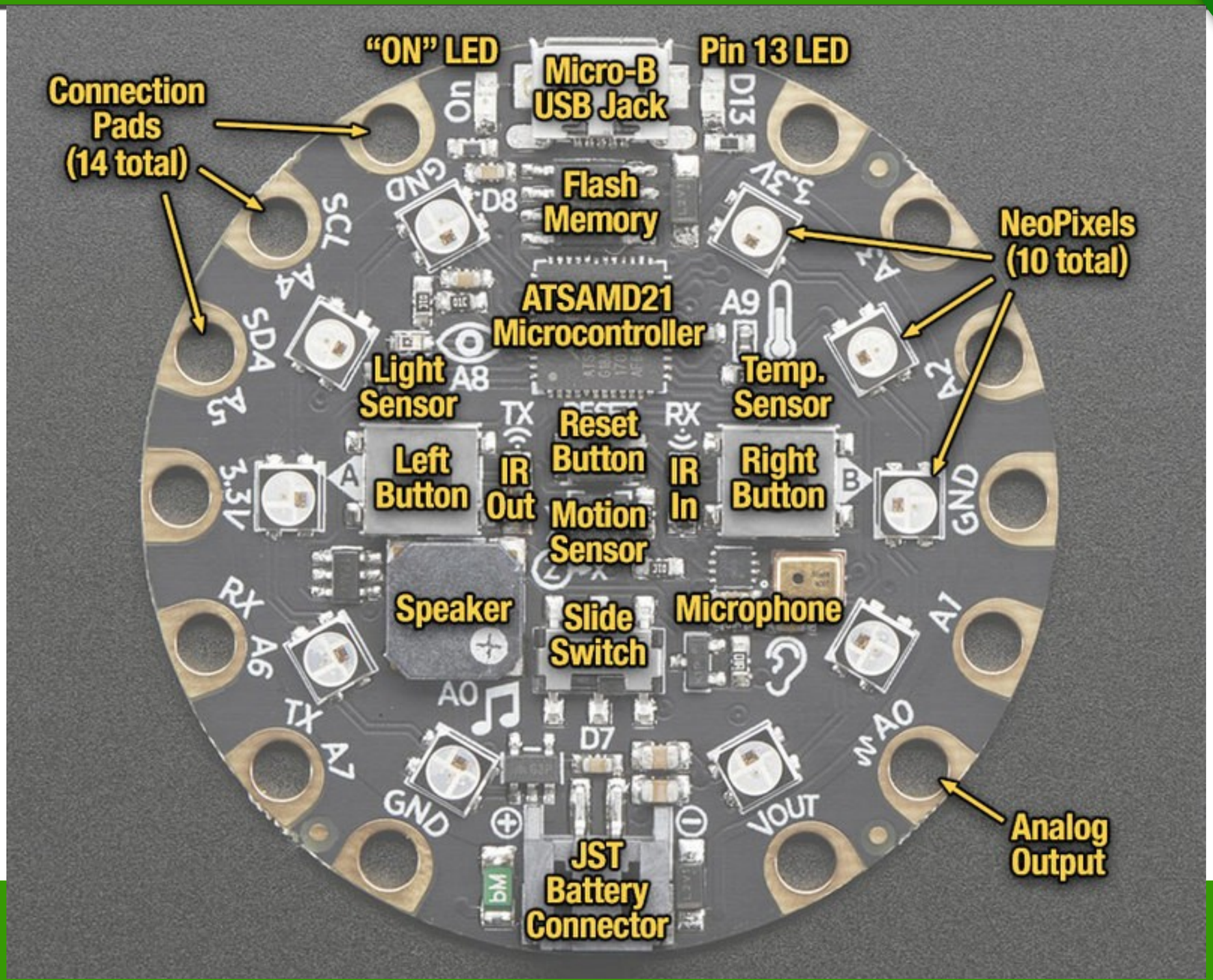
¿Que es CircuitPython?

- Python es un lenguaje de programación, apto para muy novatos y muy expertos, de mucho crecimiento, nuestro preferido para microcontroladoras
- No ocupamos herramientas raras, solo un editor de texto. El código se guarda en la placa en una memoria usb
- Nota: Charla 2pm con Yeffri Salazar

Cargando CircuitPython

- El flujo de trabajo hace que sea muy rápido desarrollar, en una forma similar a como lo vamos a hacer hoy
- Para siempre tener la última versión, vamos a circuitpython.org donde damos click en Downloads. Buscamos nuestra tarjeta, y por último seleccionamos “Spanish” para el idioma
- Nos queda un archivo extensión **uf2**

¿Que puedo hacer con una CircuitPlayground Express?



¿Que puedo hacer con una CircuitPlayground Express?

- Si la deseas usar para robótica, le puedes agregar una Crikit
- En learn.adafruit.com existen decenas de ejemplos de proyectos, tanto con CircuitPython como con la CircuitPlayground Express
- Para correr muchos de los ejemplos, solo debemos hacer copy+paste del código hacia la tarjeta

Luces LED

- Para prender un LED, solo es necesario conectar a batería
- Si queremos muchas luces, con colores diferentes, la cantidad de cables y el enredo se vuelve mucho
- Tenemos luces inteligentes o Neopixeles
  - Se conectan como una tira
  - Cambian de color
  - Se define un color diferente para cada luz en la tira
  - Mucho más sencillo de conectar y alimentar

Luces LED: En CircuitPython

- Se recomienda la guía traducida al español de CircuitPython sin complicaciones (capítulo de neopixeles)
- En esta guía tenemos ejemplos sencillos de como usar los componentes incluidos en la tarjeta
- Usamos una librería de CircuitPython. Esto permite obtener funcionalidad adicional, sin tener que mantener partes complejas o que tienen que ver con electrónica. Esto hace que el código siempre sea sencillo de leer

Luces LED: En CircuitPython

- Con este primer ejemplo, vamos a prender solo una luz, de color verde

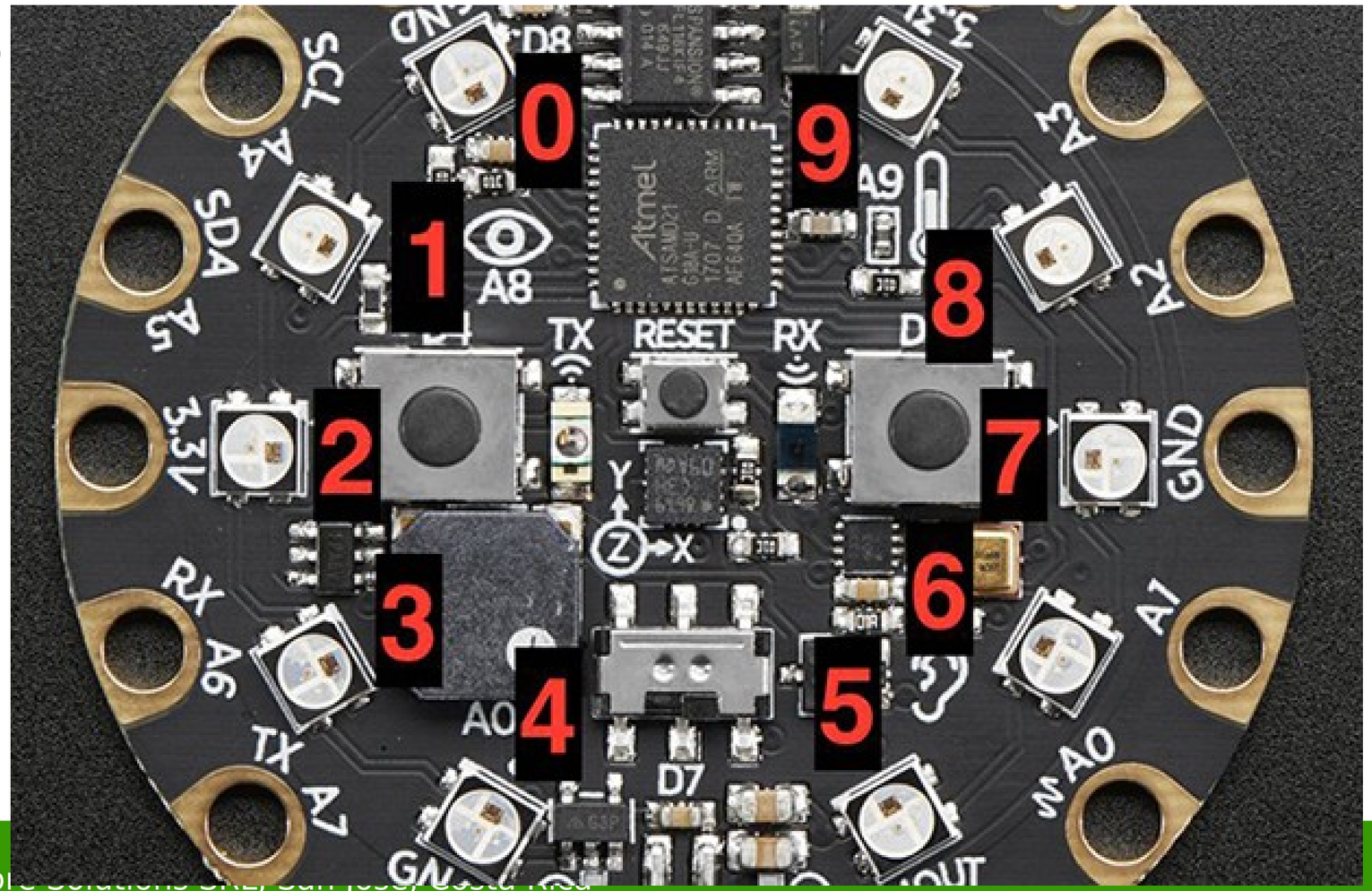
```
from adafruit_circuitplayground import cp

cp.pixels[0] = (( 0, 255, 0 ))


while True:  # Ciclo principal
    pass
```

Luces LED: En CircuitPython

- Cuando enviamos un color hacia las luces, se usa una “tupla RGB”, donde los valores que van adentro, son los colores rojo, verde y azul, con valores de 0 a 255



Luces LED: En CircuitPython

- Ahora, en lugar de solo una, las prendemos **todas.** Ojo, ya no usamos el símbolo de **=**

```
from adafruit_circuitplayground import cp

cp.pixels.fill (( 0, 255, 0 ))


while True:  # Ciclo principal
    pass
```

Luces LED: En CircuitPython

- Las luces pueden ser muy brillantes, por lo que en lugar de 255, bajamos a 10

```
from adafruit_circuitplayground import cp

cp.pixels.fill (( 0, 10, 0 ))

while True:  # Ciclo principal
    pass
```

Fractales

- Para que la animación parezca una flama de una candela o vela, vamos a usar un truco muy sencillo, que también lo usa la naturaleza, los números fractales
- Si cambiamos el brillo de forma lineal, nuestro cerebro lo detecta como algo artificial. Pero si lo cambiamos usando fractales, la animación parece natural y real
- Nota: La flama se ve mucho más realista en vivo que en la cámara

Fractales

- También usamos una librería que se llama random
- Permite tomar valores al azar, o sea, lo que sucede cuando tiramos una moneda al aire (“¿escudo o corona?”) o cuando tiramos un dado de 6 caras o de 20 caras
- Esto permite que la aplicación no sea predecible y aburrida, sino que siempre se comporta diferente, al igual que una candela real
- Para ordenar mejor el código, creamos una función

Código con fractales

- from adafruit_circuitplayground import cp
- import math
- import random
-
- anterior = 128

Código con fractales

```
def dividir(brillo_inicial, brillo_final, cambio):
    if cambio != 0:
        mitad = ((brillo_inicial + brillo_final + 1) / 2 + random.randint(-cambio, cambio))
        cambio = int(cambio / 2)
        dividir(brillo_inicial, mitad, cambio)
        dividir(mitad, brillo_final, cambio)
    else:
        nivel = math.pow(brillo_inicial / 255.0, 2.7) * 255.0 + 0.5
        cp.pixels.fill((int(nivel), int(nivel / 8), int(nivel / 48)))
```

Código con fractales

```
while True:  # Ciclo principal
    azar = random.randint(64, 191)
    dividir(anterior, azar, 32)
    anterior = azar
```

Búsqueme en:
alvaro@greencore.co.cr

https://github.com/fede2cr
https://twitter.com/fede2_cr
Youtube: https://bit.ly/2X0jmIV