



# Projeto 10

## Controle Gráfico – Prática

Jan K. S. – janks@puc-rio.br

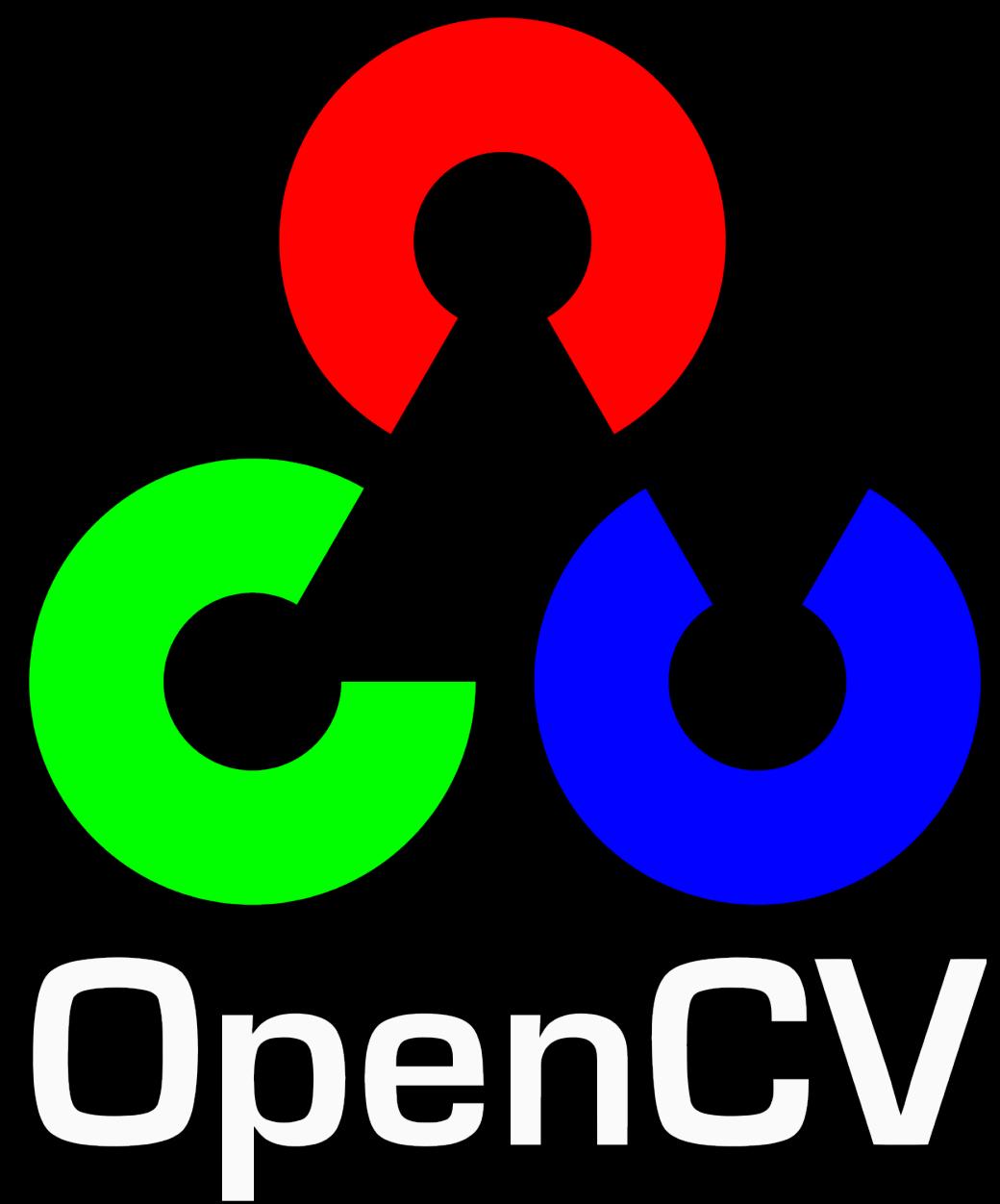
ENG1419 – Programação de Microcontroladores

# Testes Iniciais

Testes Iniciais 1

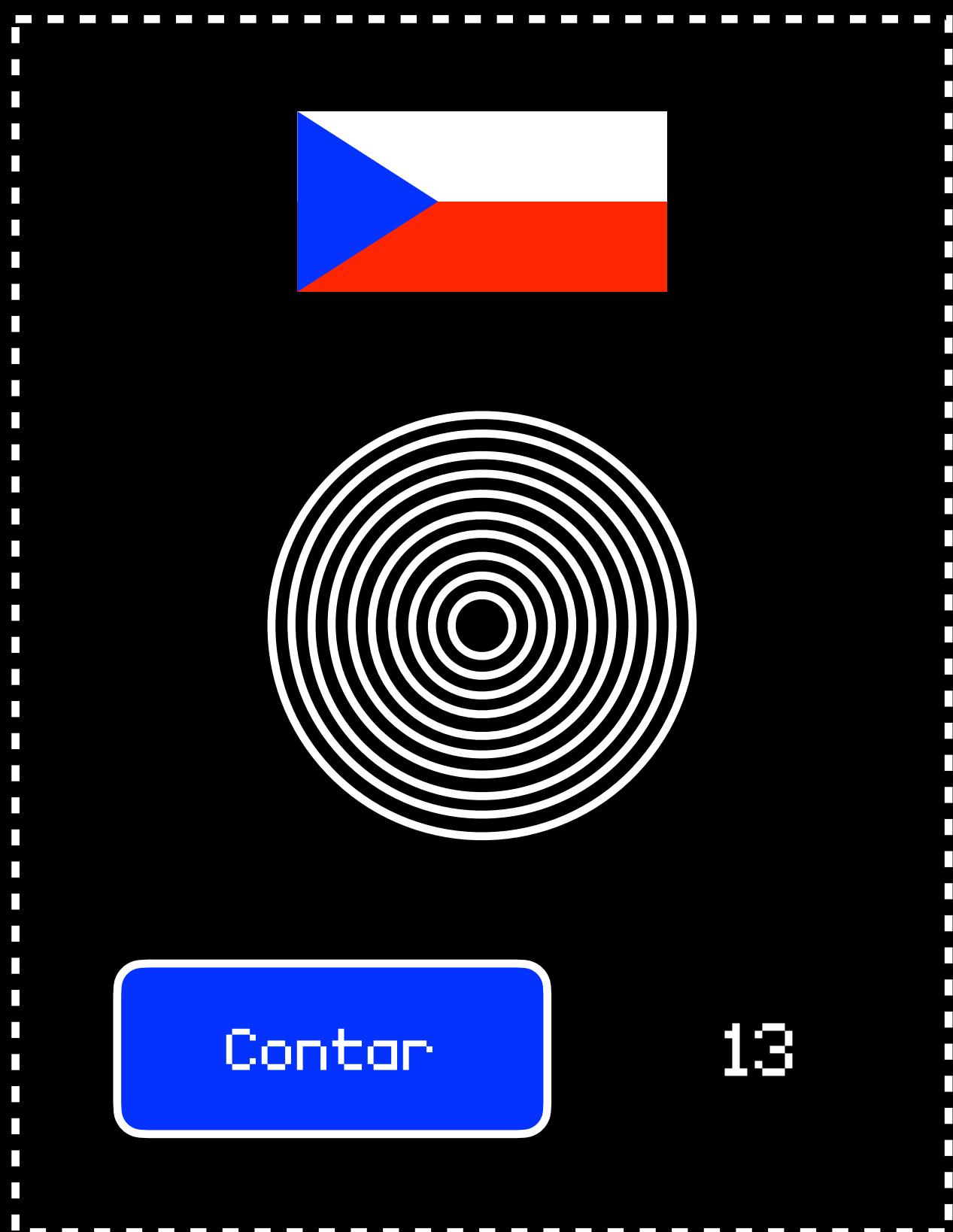


Testes Iniciais 2

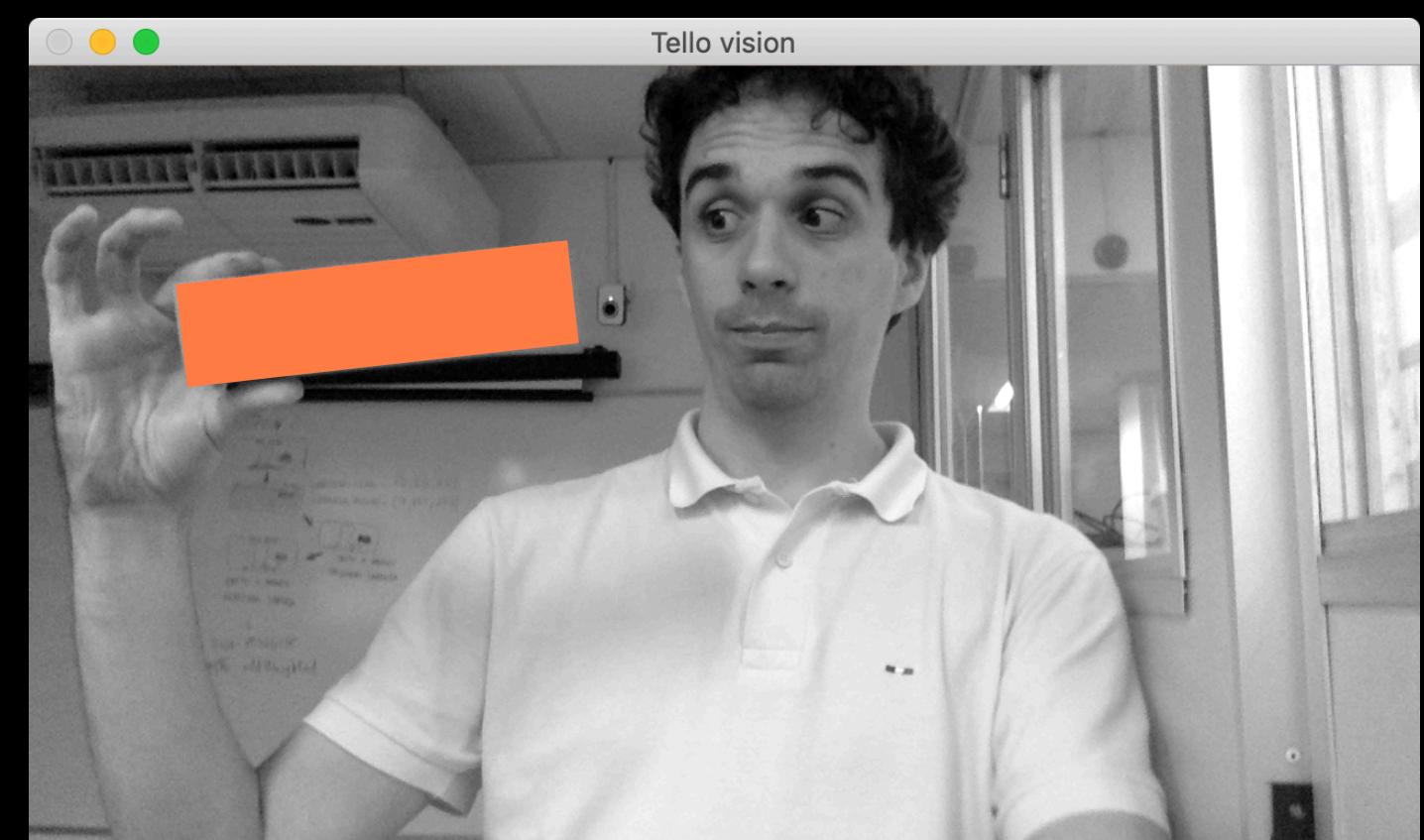


Duas Partes dos Testes Iniciais

## Testes Iniciais 1



## Testes Iniciais 2



Duas Partes dos Testes Iniciais

```
#include <Adafruit_GFX.h>
#include <MCUFRIEND_kbv.h>
```

```
MCUFRIEND_kbv tela;
```

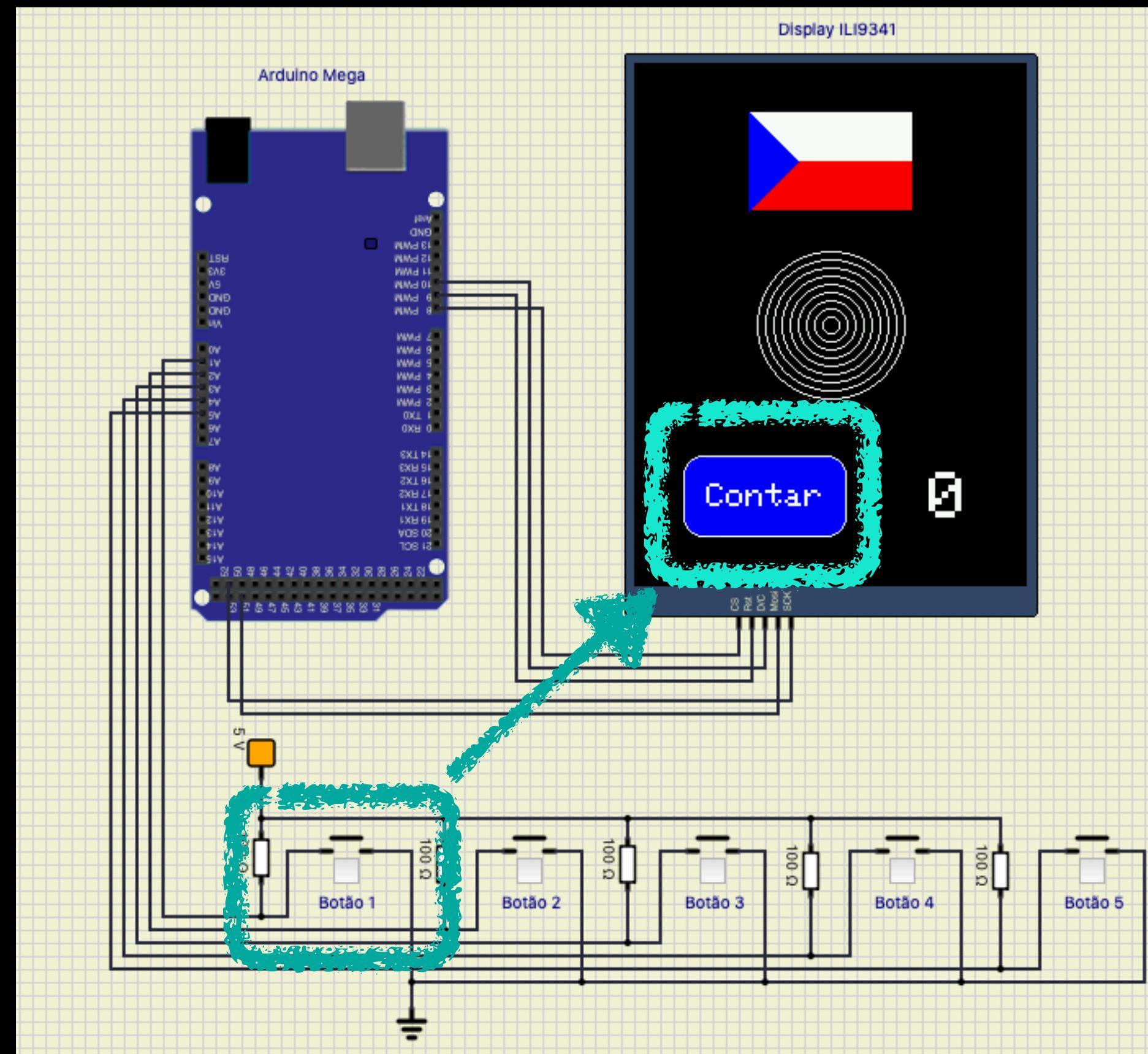
```
tela.begin( tela.readID() );
tela.fillScreen(TFT_BLACK);
```



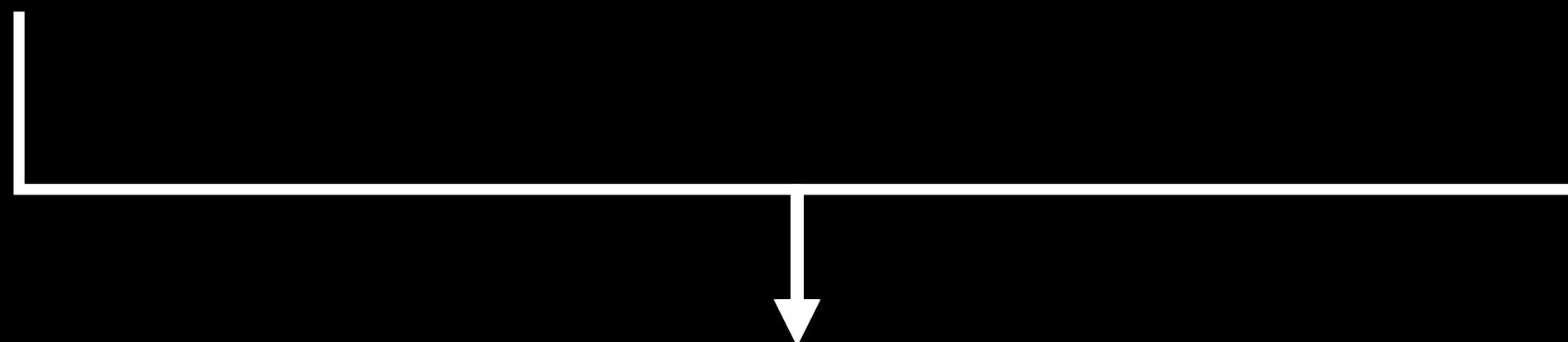
```
#include <Adafruit_GFX.h>
#include <Adafruit_ILI9341.h>
```

```
Adafruit_ILI9341 tela = Adafruit_ILI9341(8, 10, 9);

tela.begin();
tela.fillScreen(ILI9341_BLACK);
```



Limitação do Simulador



Combinação de Duas Imagens para o Efeito Desejado

**Copie as pastas das bibliotecas extras para o diretório Documentos/Arduino/libraries.**

**Desenhe a bandeira da República Checa no topo da tela.**



## Testes Iniciais 01

**Desenhe 10 círculos concêntricos** de contorno branco no meio da tela, incrementando o raio de 5 em 5.  
↳ DICA: use um for.

Desenhe um **botão com o texto "Contar"** no fundo da tela. Ao tocar no botão, incremente uma variável de contagem e imprima pela serial.  
↳ DICA: lembre-se que o touch do JKSTButton no Simulide é reconhecido apertando o botões abaixo do display.

Ao incrementar a contagem, **desenhe o número** ao lado do botão.  
↳ DICA: lembre-se de apagar o texto anterior antes de desenhar o novo valor.

Pegue um objeto de 1 cor da sua casa. **Rode o arquivo escolherCor.py**. Mexa nos controles até que apenas a sua cor apareça. Anote os valores mínimos e máximos das componentes HSV. Aperte "q" para sair.

Abra o arquivo `10a_testes_iniciais_2.py`. Comece fazendo o **stream da câmera** direto na tela.



A partir da imagem da câmera, crie e exiba uma imagem que mostre **apenas a cor escolhida**, colocando o **resto como preto**.

Crie e exiba uma outra imagem que **substitua a cor escolhida por preto** e que coloque o **resto em tons de cinza**.

↪ DICA: adapte o exemplo da máscara invertida da teoria.

## Testes Iniciais 2

**Some as duas imagens anteriores**, gerando uma que seja colorida nas regiões da cor escolhida e com tons de cinza no resto. Exiba essa imagem na tela.

↪ DICA: sim, é para usar o + mesmo.

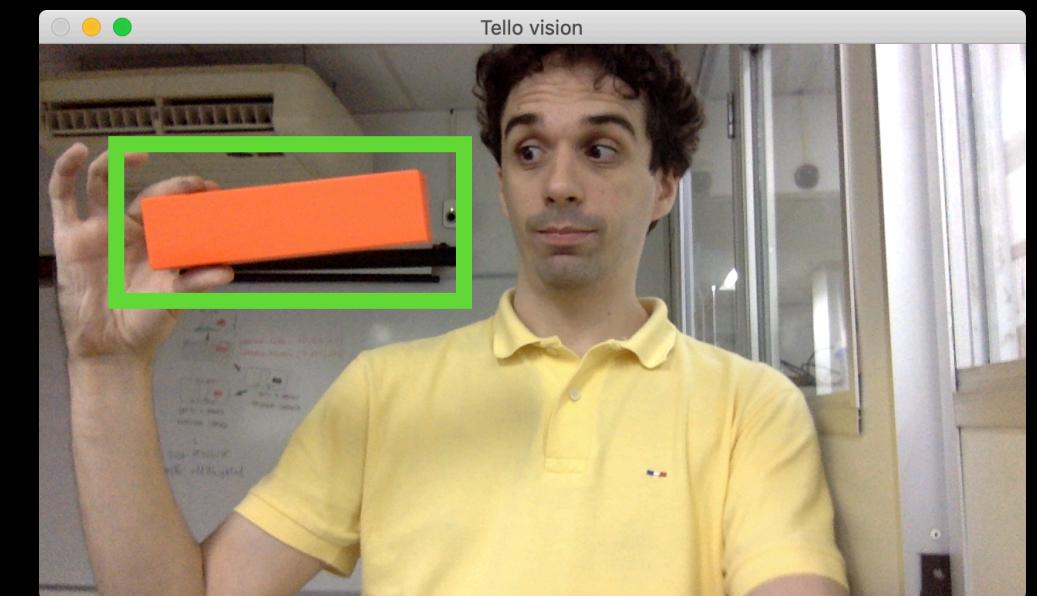
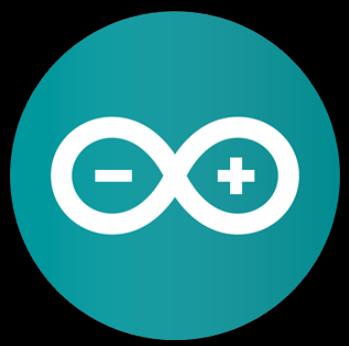
**Escreva** o nome do seu artista favorito no topo à direita da imagem resultante, com a **cor laranja**.

↪ DICA: lembre-se que a cor aqui está no formato BGR.

# Implementação



Drone Tello



Controlador de Voo + Reconhecimento de Imagem



Crie os botões na tela que enviem pela serial os textos "decolar" e "pousar" ao serem pressionados.

Crie os botões na tela que imprimam na serial os textos "esquerda", "frente" e "direita" ao serem pressionados e "parar" ao serem liberados.  
↪ DICA: use a cor `ILI9341_LIGHTGREY`.

## Implementação 01



## Implementação 02

Detecte **regiões da cor escolhida nos Testes Iniciais** na imagem da câmera, e **desenhe retângulos verdes** ao redor delas.

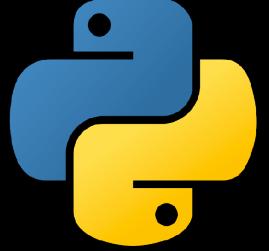
↪ DICA: use a imagem capturada pela biblioteca do drone. Ela permite testar esta parte mesmo sem uma câmera no computador – nesse caso, use os limites de cor (0, 60, 60) e (20, 255, 255).

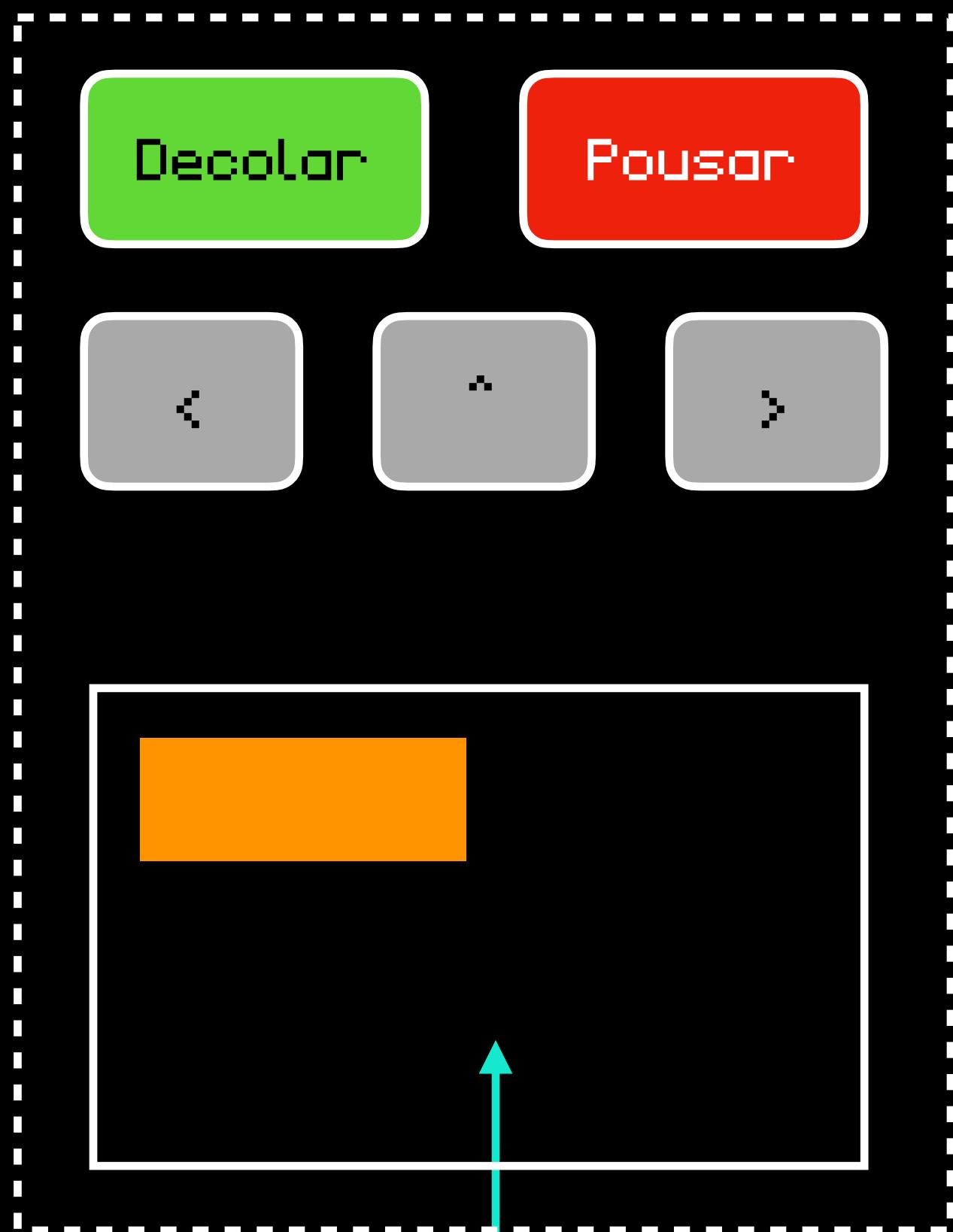
Modifique o código acima para desenhar **apenas o retângulo que tiver a maior área**, e apenas se essa área **for maior que 2000**.

↪ DICA: crie variáveis antes do for para armazenar os dados da maior área.

# Aperfeiçoamento



 python



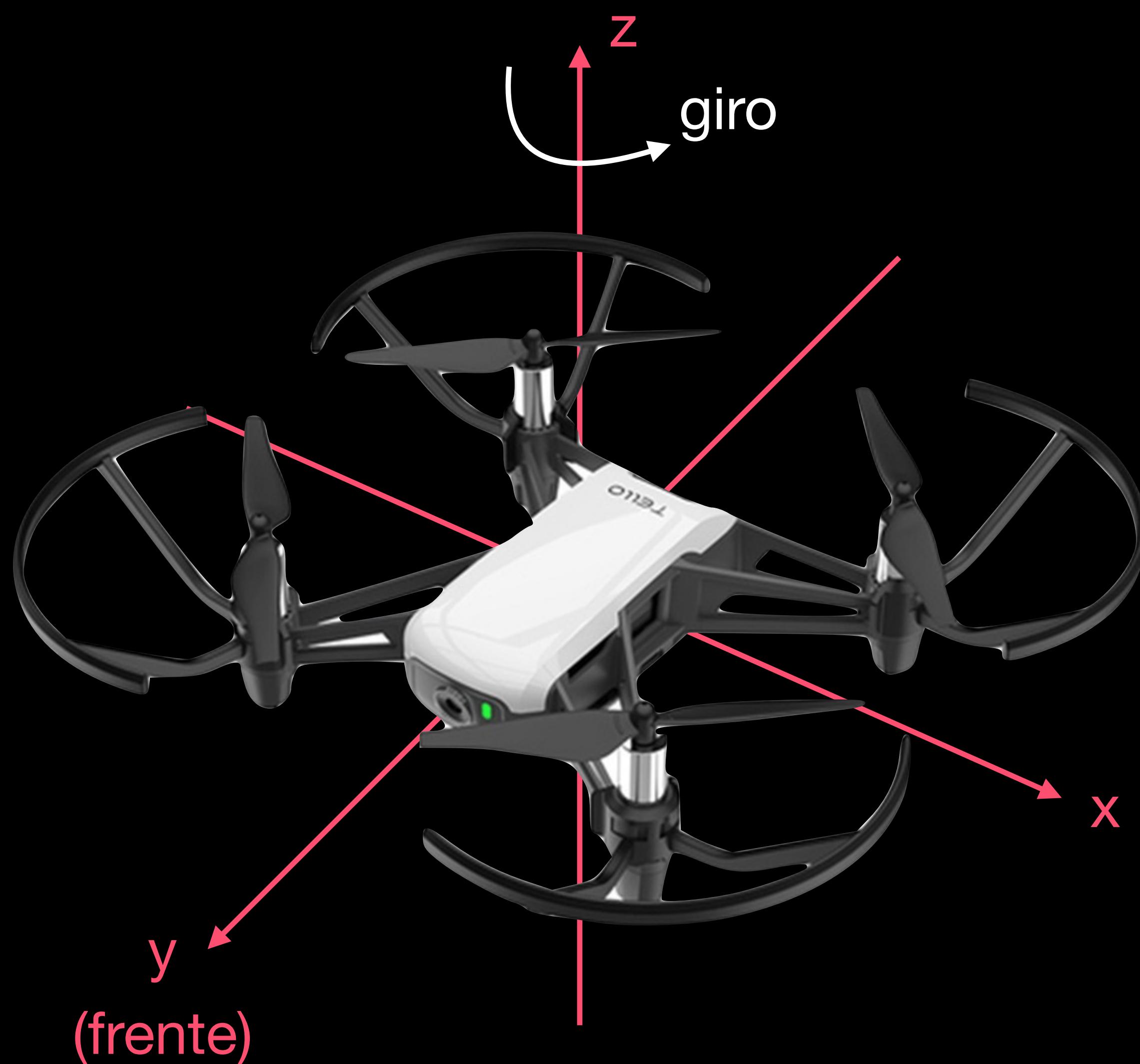
"decolar"  
"pousar"  
"esquerda"  
"frente"  
"direita"  
"parar"



"retangulo 020 030 100 032"  
x y comprimento altura



Controlador de Voo + Reconhecimento de Imagem

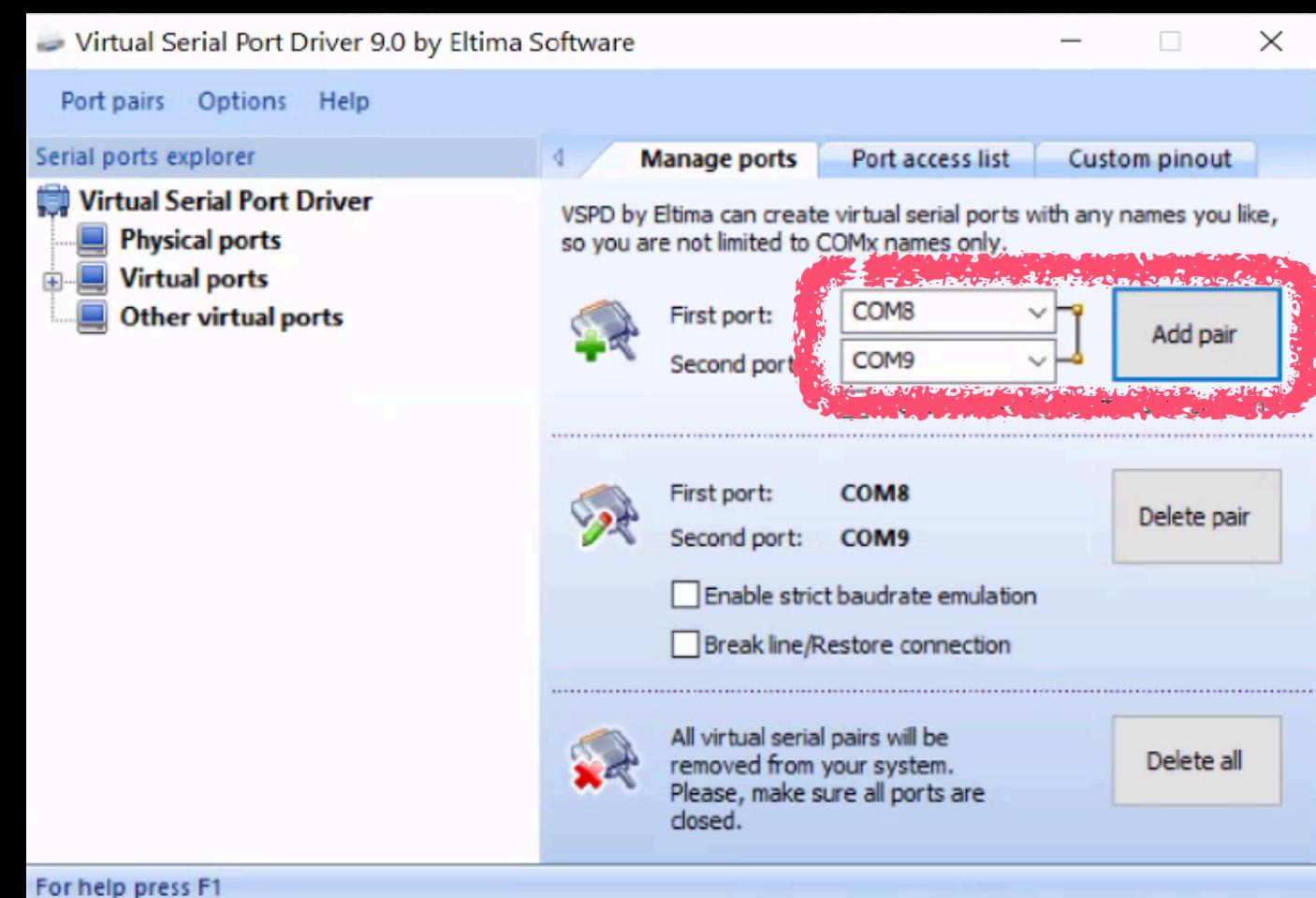
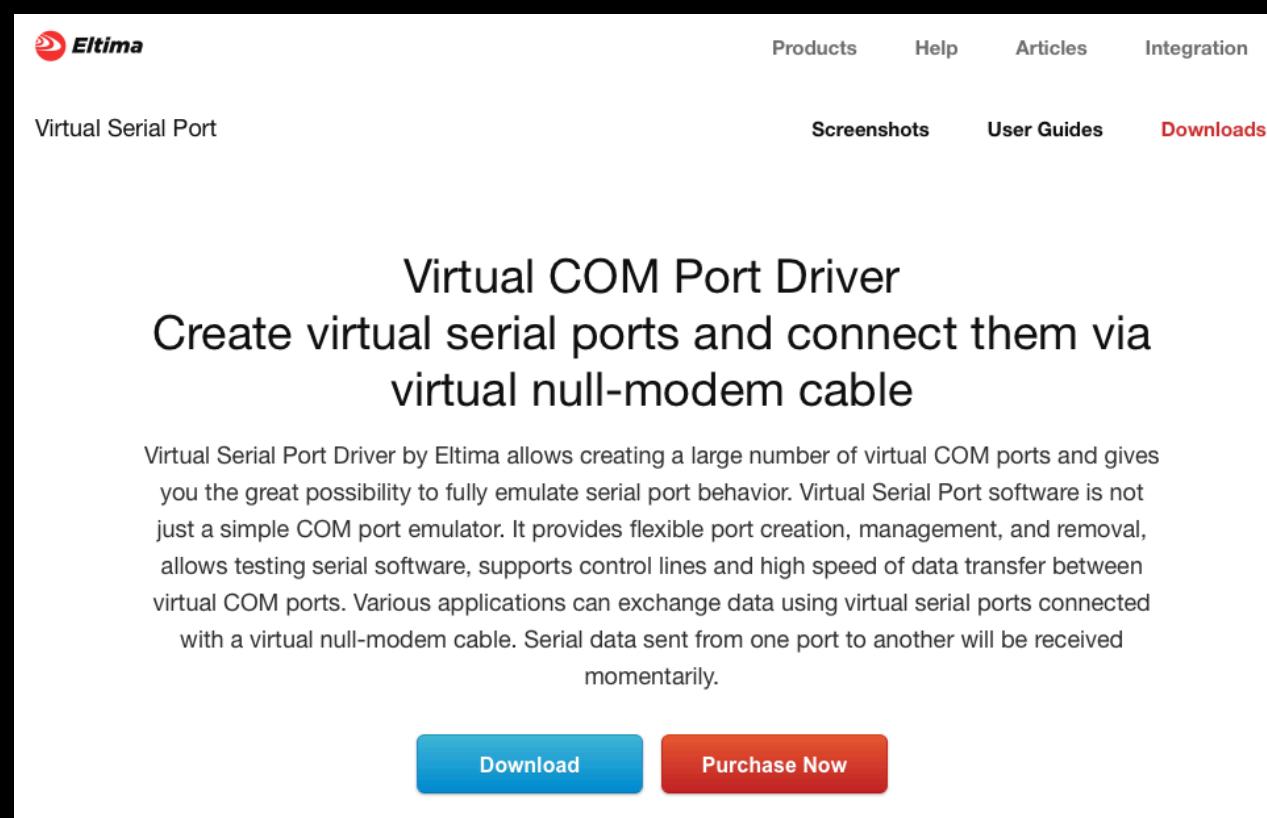


Coordenadas do Drone Tello

```
>>> from extra.tello import Tello  
  
>>> drone = Tello("TELLO-D023AE", test_mode=True)  
>>> drone.inicia_cmds()  
  
>>> drone.takeoff()  
>>> drone.land()  
>>> drone.goto(deltaX, deltaY, deltaZ, velocidade)  
>>> drone.rc(velocidade_x, velocidade_y, velocidade_z,  
velocidade_giro) # velocidades entre -100 e 100  
>>> imagem = drone.current_image  
>>> drone.state # dados como bateria, wifi, etc  
{'mid': 32.0, 'x': 0.0, 'y': 0.0, 'z': 0.0, 'mpry':  
'0,0,0', 'pitch': 1.0, 'roll': 0.0, 'yaw': 53.0,  
'vgx': 0.0, 'vgy': 0.0, 'vgz': 0.0, 'templ': 82.0,  
'temph': 83.0, 'tof': 10.0, 'h': 0.0, 'bat': 90.0,  
'baro': -111.55, 'time': 4.0, 'agx': 21.0, 'agy':  
-9.0, 'agz': -1000.0}
```

# Windows

<https://www.virtual-serial-port.org>



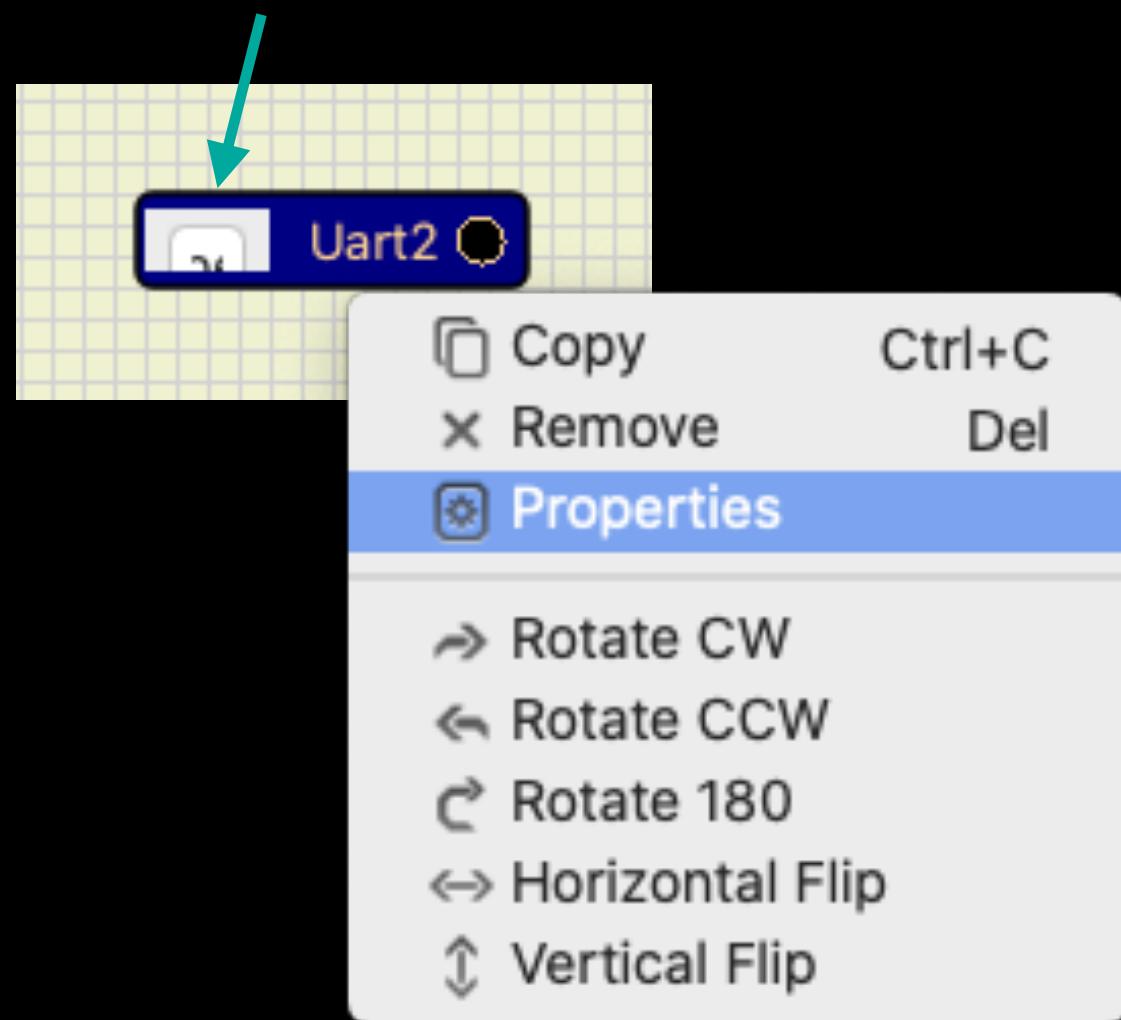
brew install socat

macOS e Linux

```
>> socat -d -d pty,raw,echo=0 pty,raw,echo=0
2020/11/24 17:50:53 socat[2952] N PTY is /dev/ttys005
2020/11/24 17:50:53 socat[2952] N PTY is /dev/ttys006
```

```
meu_serial = Serial("COM9", baudrate=9600, timeout=0.1)
```

clique para abrir a serial!



Name	Value
itemtype	SerialPort
id	SerialPort-78
Show id	false
Auto Open	false
Mcu Uart	2
Port Name	COM8
BaudRate	Baud9600
DataBits	Data8
Parity	NoParity
StopBits	OneStop
FlowControl	NoFlowControl

Definição das Portas Seriais no Python e no SimulIDE

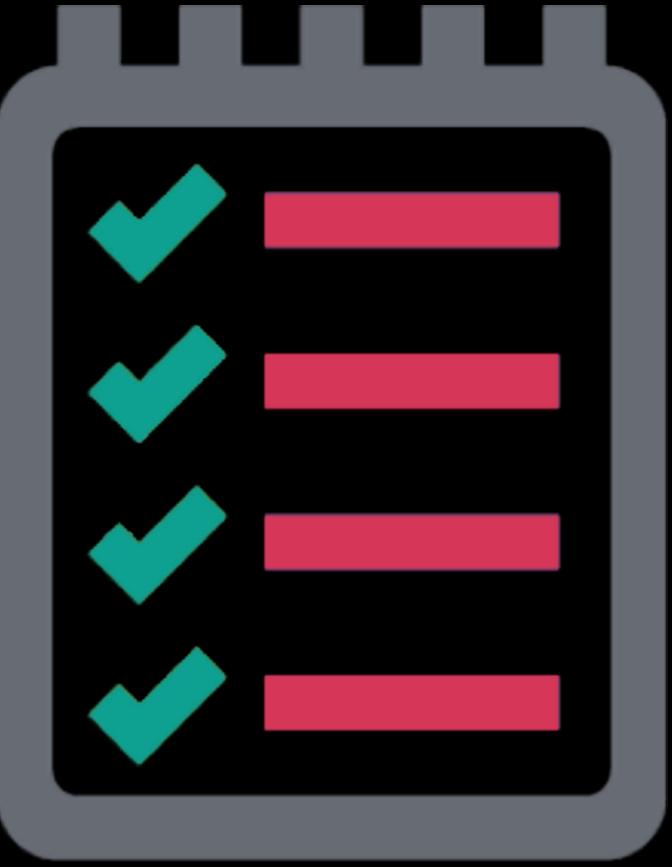


Desenhe uma área retangular com contorno branco no fundo da tela, com dimensões 202x152.

Ao receber um texto na Serial no formato "retangulo 020 030 100 032", imprima o texto recebido e desenhe um retângulo laranja na posição e dimensões especificadas dentro da área, apagando o retângulo anterior.  
↳ DICA: é necessário somar um valor de deslocamento nas coordenadas recebidas, para desenhar dentro da área do item anterior.

## Aperfeiçoamento 01

Inicie a Serial1. Ao clicar nos botões virtuais, envie o texto pela Serial1 também. E, no código do item anterior, mude a leitura para a Serial1.



## Aperfeiçoamento 02

Inicie a serial virtual no seu computador, com o software adequado para o seu sistema operacional.

Ao receber os textos "decolar", "pousar", "esquerda", "frente", "direita" e "parar" pela serial, envie comandos para o drone. Gire o drone no caso da "esquerda" ou "direita".  
↪ DICA: use os comandos `drone.takeoff`, `drone.land` e `drone.rc`. Use -40, 0 ou 40 como valores de velocidade.

Faça um timer recorrente de 1 segundo que imprima no console e envie pela serial as coordenadas e as dimensões do retângulo detectado no formato "retangulo 020 030 100 032". Esses valores devem estar mapeados para as dimensões 200 x 150.

↪ DICA: pesquise no Google como obter o comprimento e altura de imagens no OpenCV. Em seguida, faça uma regra de três para os quatro valores antes de enviar.

Teste a conexão serial entre o SimulIDE e o Python.



[janks.link/micro/projeto10.zip](https://janks.link/micro/projeto10.zip)

Material do Projeto 10