

# Projeto 02

## Controle Remoto – Prática

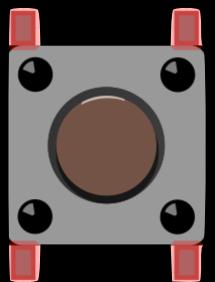
Jan K. S. – janks@puc-rio.br

ENG 1419 – Programação de Microcontroladores

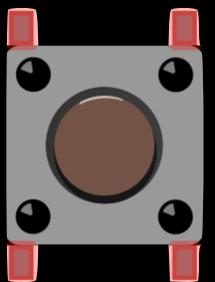
# Testes Iniciais



seleciona um LED  
e  
acende ou apaga



acende todos os LEDs



apaga todos os LEDs



Controle Remoto de LEDs

1 2 3 4 5

seleciona LED  
pelo número

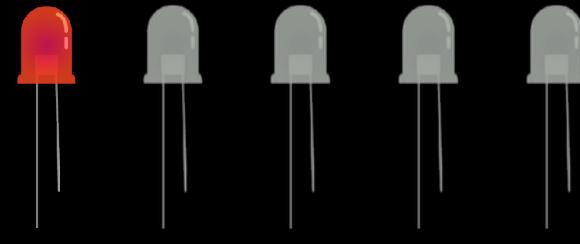
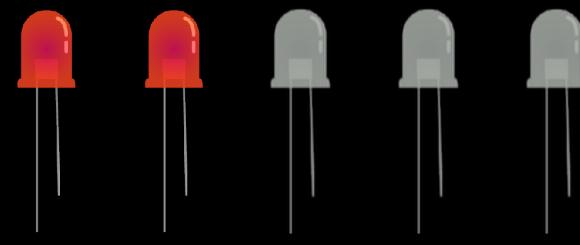
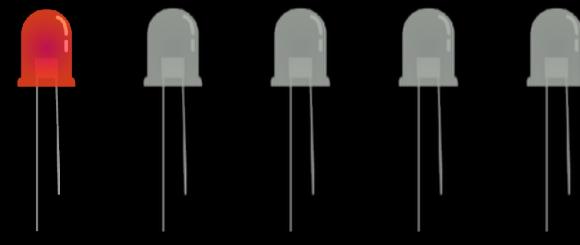
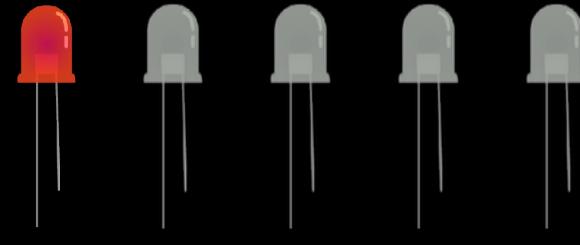
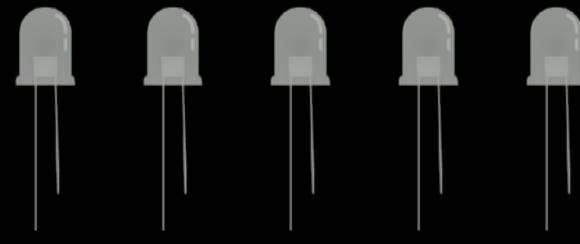
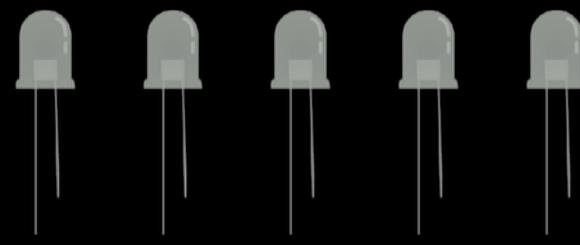


vai para próximo LED  
ou anterior



apaga/acende  
LED selecionado

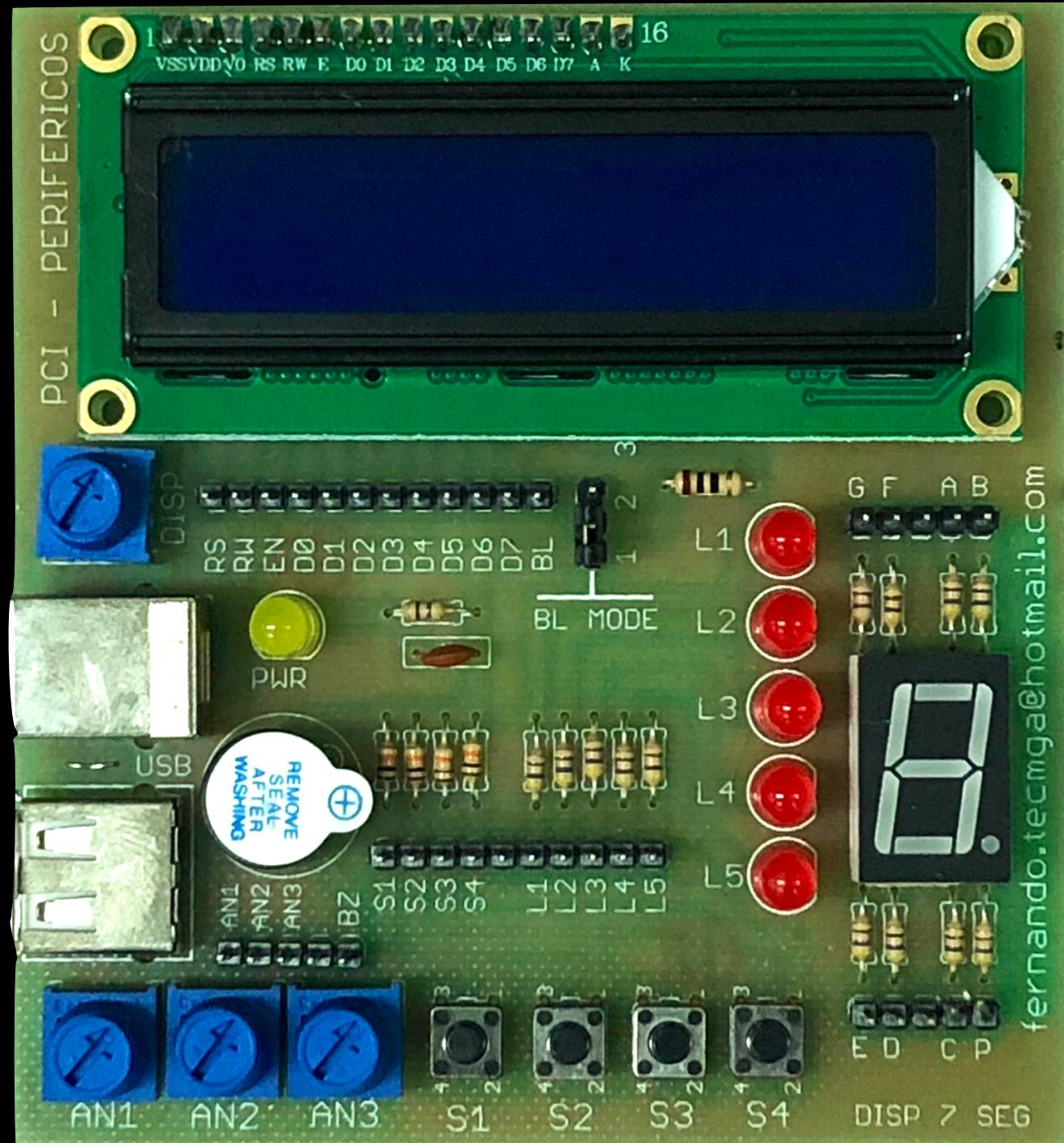
Controle Remoto de LEDs





		KEY_UP	
KEY_LEFT	KEY_OK	KEY_RIGHT	
	KEY_DOWN		
	KEY_1	KEY_2	KEY_3
	KEY_4	KEY_5	KEY_6
	KEY_7	KEY_8	KEY_9
		KEY_0	

GPIO 2, 3, 4, 5, 6, 7



GPIO 11, 12, 13, 14

GPIO 21  
GPIO 22  
GPIO 23  
GPIO 24  
GPIO 25

leds = [LED(21), LED(22), LED(23), LED(24), LED(25)]

Conexões dos Botões, LEDs e LCD de Caracters com as Portas da GPIO



## Testes Iniciais

Ao apertar o **Botão 1 da placa**, acenda todos os LEDs. Ao apertão o **Botão 2 da placa**, apague todos os LEDs.

↳ DICA: use a propriedade `when_pressed` do botão e crie funções que percorram a lista de LEDs.

Ao apertar uma **tecla numérica**, "seleccione" um dos LEDs, mostrando o valor no LCD de caracteres.

↳ DICA: use o código do recebimento de teclas com o `nextcode` dentro do `while True`.

Ao apertar a **tecla OK**, mude o estado (aceso/apagado) do LED selecionado.

↳ DICA: use uma variável para guardar o índice do LED selecionado.

Ao apertar as **setas para cima/baixo**, altere o LED selecionado. Garanta que o valor desse LED não ultrapasse os limites de 1 a 5.

# Implementação



Controle Remoto de uma TV Digital



Conversor Digital Aquário

# controle "aquario"



KEY\_POWER

KEY\_MUTE

KEY\_1

KEY\_2

KEY\_3

KEY\_4

KEY\_5

KEY\_6

KEY\_7

KEY\_8

KEY\_9

KEY\_0

KEY\_MENU

KEY\_EXIT

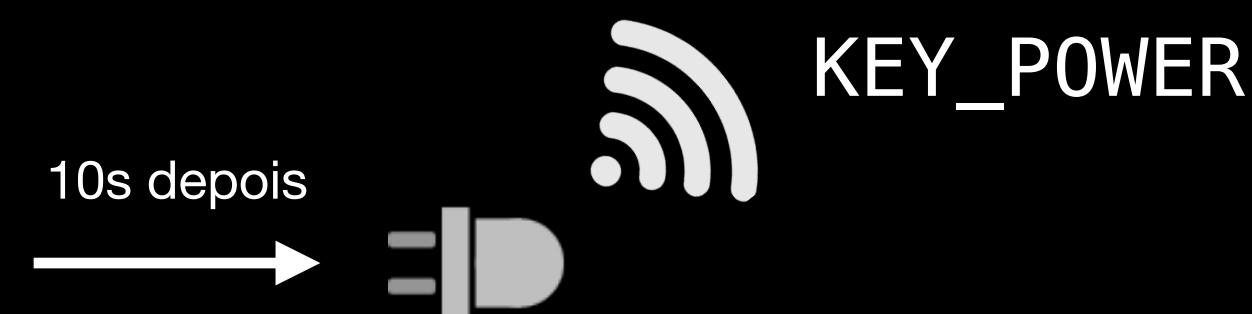
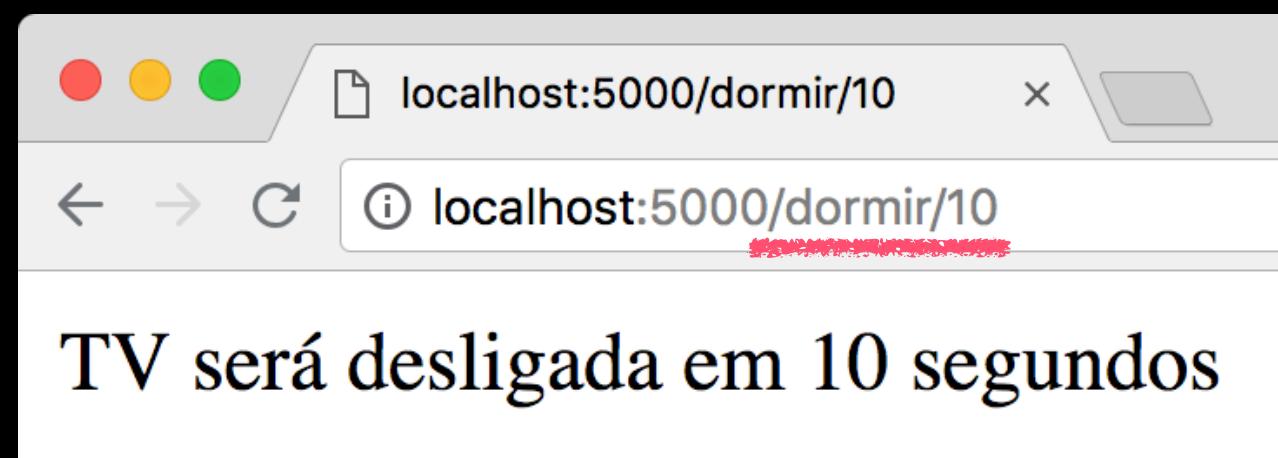
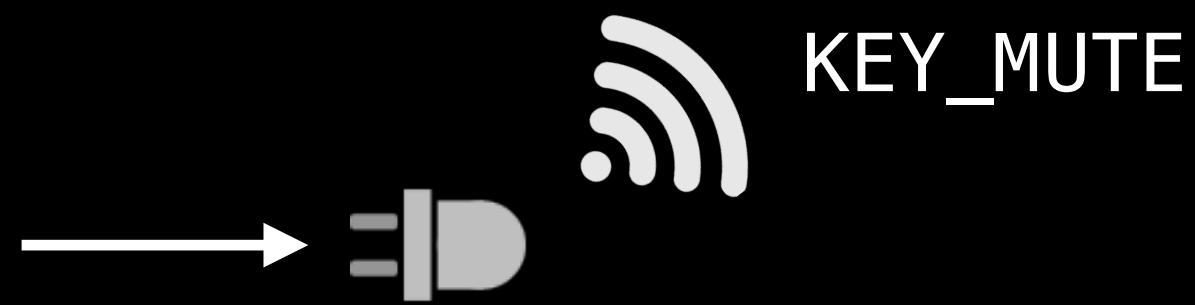
KEY\_CHANNELUP

KEY\_VOLUMEDOWN

KEY\_OK

KEY\_VOLUMEUP

KEY CHANNELDOWN





## Implementação

Crie um **servidor com páginas** que transmitam comandos infravermelho da tecla Power, aumentar o volume, diminuir o volume e função "mudo". Teste essas páginas no navegador.

↪ DICA: use a `send_once`. Lembre-se que toda a função de página deve retornar um texto.

Crie uma **página que mude para um canal**, recebendo um parâmetro tipo "string" com o texto dos dígitos numéricos (ex: "0431") e transmitindo em seguida cada uma dessas teclas.

↪ DICA: use um `for` para acessar cada caracter da string, e faça concatenação para enviar o código.

Crie uma **página** com um parâmetro numérico N no endereço **que desligue a TV** após N segundos.

↪ DICA: não use a função `sleep!` Pesquise sobre o objeto `Timer` do Python, da biblioteca `threading`.

# Aperfeiçoamento



02b\_implementacao.py

cópia  
----->



02c\_aperfeicoamento.py

Cópia do Código da Implementação para o Aperfeiçoamento



Troca de Canais pelo Celular



02.01	TV Brasil HD
02.02	NBR
02.03	TV Escola
02.04	Canal Saúde
02.31	TV Brasil 1seg
04.01	Globo HD
04.31	Globo 1seg
06.01	Rede TV! HD
06.31	Rede TV! 1seg
07.01	Band HD
07.31	Band 1seg
09.01	CNT HD
09.31	CNT Movel
11.01	SBT HD
11.31	SBT 1seg
12.01	NGT HDTV
12.31	NGT 1seg
13.01	Record HD
13.31	Record 1seg
14.01	RCI 1seg
14.31	RCI Movel
20.10	Canção Nova HD
20.31	Canção Nova 1 seg

...



Crie uma página principal com links para os canais (pelo menos 5, exibindo o nome deles) e as outras funcionalidade (volume, mudo, power, dormir, etc).  
↪ DICA: use a `render_template`.

Modifique as funções da Implementação para redirecionarem para a página principal, em vez de retornarem um texto.

Use o `ngrok` para controlar a TV pelo celular.

## Aperfeiçoamento

Otimize o tamanho da página para a tela do celular.  
↪ DICA: pesquise no Google sobre a tag HTML `meta viewport`

# Desafio Extra

~~página.html~~

```
<ul>
  link para canal 1
  link para canal 2
  link para canal 3
  link para canal 4
  ...
</ul>
```

canais.json

```
[  
  {  
    "nome": "Globo",  
    "codigo": "0431"  
  },  
  {  
    "nome": "Band",  
    "codigo": "0731"  
  },  
  ...  
]
```



~~página.html~~

```
<ul>
  para cada canal:  
    gera link do canal  
</ul>
```

Geração da Lista de Canais a Partir de um JSON



Crie um **arquivo JSON** com uma lista de dicionários de canais (cada um com chaves "nome" e "código").

**Carregue o arquivo JSON** ao iniciar o servidor, gerando uma lista de dicionários dos canais.

**Passe a lista de dicionários para o template** ao renderizar página principal.

↪ **DICA:** pesquise no Google como passar parâmetros para um template **\*\*em Flask\*\***.

## Desafio Extra

**Gere os links de canais** no template HTML **a partir da lista**.

↪ **DICA:** pesquise no Google como criar um loop dentro de um template HTML **\*\*em Flask\*\***.



[janks.link/micro/projeto02.zip](https://janks.link/micro/projeto02.zip)

Material do Projeto 02