

# Projeto 01

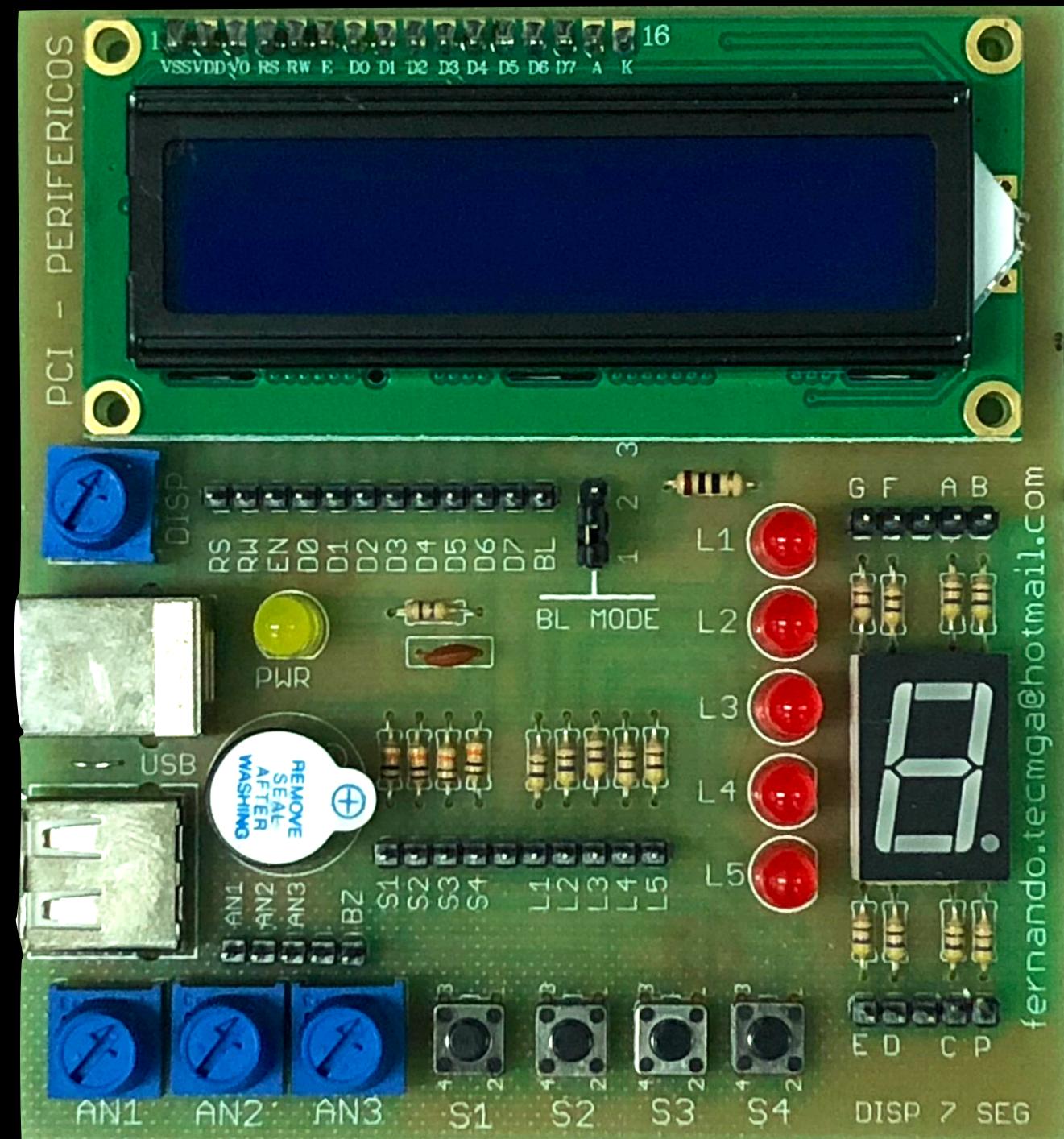
## Controle de Mídia – Prática

Jan K. S. – [janks@puc-rio.br](mailto:janks@puc-rio.br)

ENG 1419 – Programação de Microcontroladores

# Testes Iniciais

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



GPIO 21  
GPIO 22  
GPIO 23  
GPIO 24  
GPIO 25

GPIO 11, 12, 13, 14

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



## Testes Iniciais

Pisque continuamente o LED 1, mantendo-o aceso por 1 segundo e apagado por 3 segundos.  
↪ DICA: chame `blink` logo após a declaração do LED 1.

Mude o estado do LED 2 (aceso/apagado) ao apertar o Botão 2.

↪ DICA: chame o comando `toggle` dentro de uma função sua, e associe-a ao evento `when_pressed` do botão.

Pisque 4 vezes o LED 3 ao apertar o Botão 3.

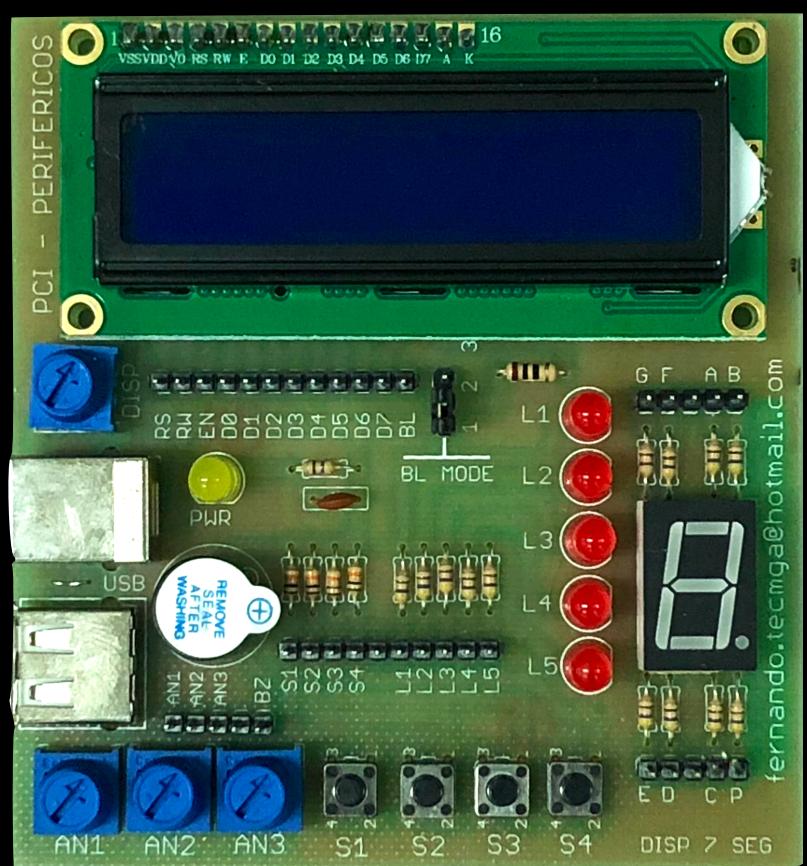
Também ao apertar o Botão 3, aumente uma contagem de quantas vezes ele foi apertado, e mostre-a no LCD.

↪ DICA: coloque mais código dentro da função do item anterior, e use uma variável global para contar.

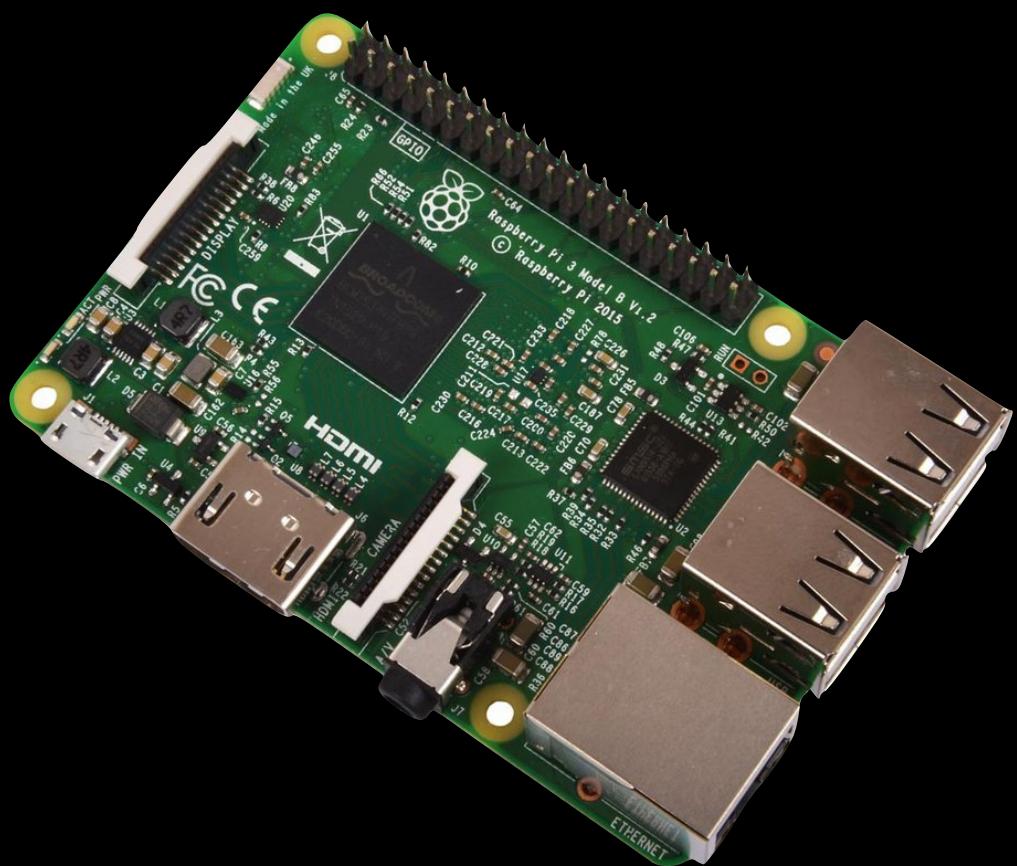
Mantenha o LED 5 aceso se e somente se o botão 1 estiver pressionado e o LED 1 estiver aceso.

↪ DICA: use um `if/else` dentro `while True` com `sleep`.

# Implementação



+



||

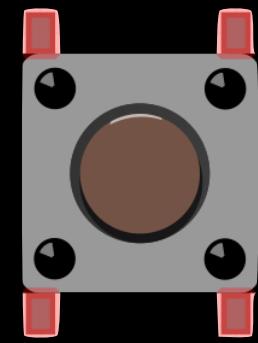
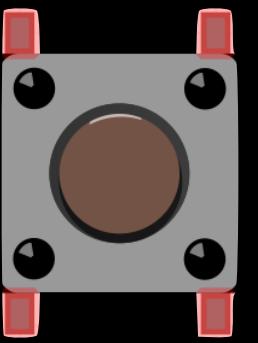
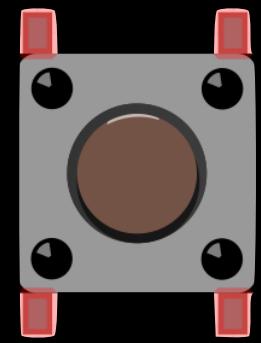


piPod

Controle de Mídia

acesso ao tocar  
piscando na pausa

Nome da Música



Faixa Anterior

Tocar/Pausar

Próxima Faixa

Controles de Mídia com Botões, LED e LCD



Lista de Reprodução

```
>>> print( player.metadata["Artist"] )  
Traceback (most recent call last):  
  File "<stdin>", line 1, in <module>  
TypeError: 'NoneType' object is not subscriptable  
  
>>> metadados = player.metadata  
  
>>> if metadados != None:  
...     print( metadados["Artist"] )
```

## Implementação



Inicie o Player e **carregue a lista de músicas** `playlist.txt`. Implemente a função de **Tocar / Pausar** no Botão 2.

**Use o LED 1 para indicar o status do player:** se estiver pausado, pisque continuamente o LED; caso contrário, mantenha-o aceso.

↪ DICA: na mesma função do item anterior, verifique se o player está tocando ou não.

Implemente a função de **avançar para Próxima Faixa** no Botão 3.

Implemente a função de **ir para a Faixa Anterior** no Botão 1. Contudo, se o tempo atual for maior que 2 segundos, o player deve **voltar ao começo da faixa atual**.

Exiba e mantenha atualizado o **nome da faixa atual** na linha superior do LCD de caracteres.

↪ DICA: use um `while True` com `sleep` de 0.2s.

# Aperfeiçoamento



01b\_implementacao.py

cópia  
----->

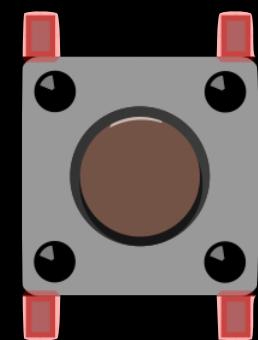
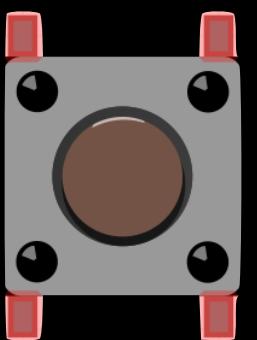
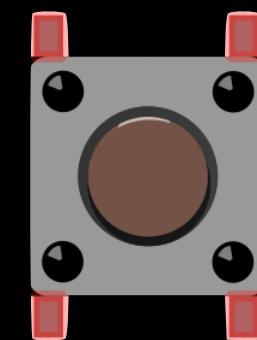


01c\_aperfeicoamento.py

Cópia do Código da Implementação para o Aperfeiçoamento

acesso ao tocar  
piscando na pausa

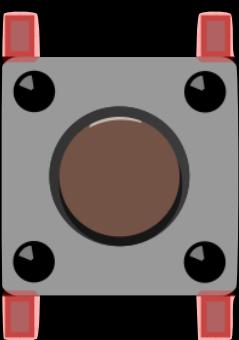
Nome da Música  
Tempo Atual e Total



Faixa Anterior

Tocar/Pausar

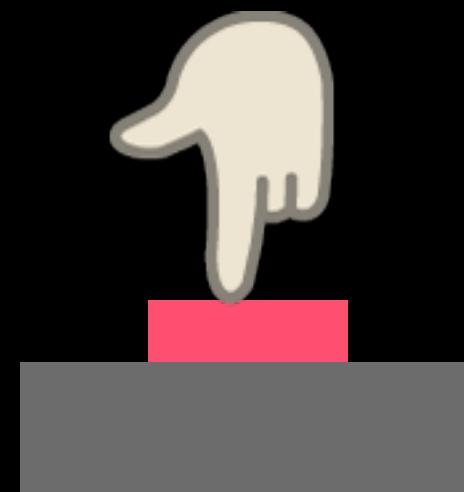
Próxima Faixa  
ou Acelerar



Como fazer para  
acelerar e pular faixa  
com o mesmo botão?

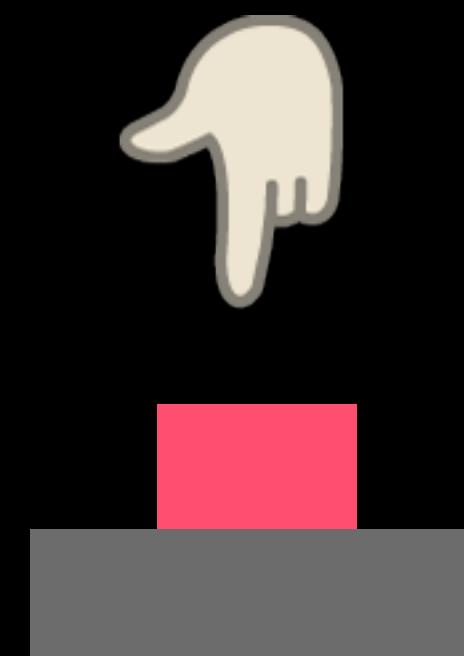


## Próxima Faixa ou Acelerar



mantendo pressionado

Ajusta velocidade para 2



ao soltar  
se velocidade  
estiver alta

Volte a velocidade para 1

caso contrário

Vai para a próxima faixa

Lógica de Avançar ou Acelerar

```
>>> segundos = player.time_pos % 60
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for %:
'NoneType' and 'int'

>>> posicao = player.time_pos
>>> if posicao != None:
...     segundos = posicao % 60
```

## Aperfeiçoamento



Ajuste a velocidade para 2 ao segurar o botão de Avançar. Volte ao normal ou pule de faixa ao soltar.  
↪ DICA: substitua a função `when_pressed` pelas funções `when_held` e `when_released`, conforme ilustrado no slide anterior.

Exiba o tempo atual e o tempo total da música na linha inferior do LCD, exatamente conforme ilustrado.

↪ DICA: pesquise no Google como formatar um número com zeros à esquerda em Python, para colocar os minutos e segundos sempre com 2 dígitos.

# Desafio Extra



01c\_aperfeicoamento.py

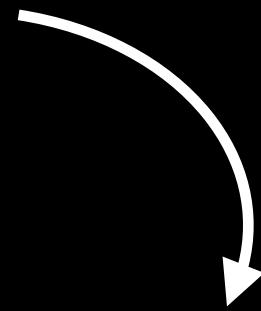
cópia  
----->



01d\_desafio.py

Cópia do Código do Aperfeiçoamento para o Desafio Extra

# Rolagem de Nomes Grandes



Summer: III Pres  
01:47 de 03:00

Summer: III Pres  
↓ ⌂  
Summer: III Pre  
↓ ⌂  
Summer: III Pres  
↓ ⌂  
...  
↓ ⌂  
to C in G Minor)



Etapa 1: Rolagem de Nomes Grandes

```
playlist.txt  
extra/musicas/Rock.mp3  
extra/musicas/Pop.mp3  
extra/musicas/HipHop.mp3  
extra/musicas/Jazz.mp3  
extra/musicas/Samba.mp3  
extra/musicas/Forro.mp3  
extra/musicas/Classica.mp3  
extra/musicas/Salsa.mp3  
extra/musicas/Funk.mp3  
extra/musicas/Sertanejo.mp3
```

embaralha



```
nova_playlist.txt — Editado  
extra/musicas/Samba.mp3  
extra/musicas/Classica.mp3  
extra/musicas/Sertanejo.mp3  
extra/musicas/Forro.mp3  
extra/musicas/HipHop.mp3  
extra/musicas/Funk.mp3  
extra/musicas/Jazz.mp3  
extra/musicas/Rock.mp3  
extra/musicas/Pop.mp3  
extra/musicas/Salsa.mp3
```

Etapa 2: Nova Lista de Reprodução Embaralhada



## Desafio Extra

Role o nome da faixa no topo caso ela seja maior que o display.

↪ DICA: use a notação texto[inicio:fim] para capturar um trecho do texto. Use e atualize uma variável global que indique a posição inicial desse trecho. NÃO USE OUTRO WHILE OU FOR.

Toque as músicas em uma ordem aleatória quando o programa começar. Para isso, leia o arquivo original da lista de músicas e gere um novo arquivo com as faixas embaralhadas.

↪ DICA: pesquise no Google como ler/escrever um arquivo e como embaralhar uma lista em Python.



[janks.link/micro/projeto01.zip](https://janks.link/micro/projeto01.zip)

Material do Projeto 01