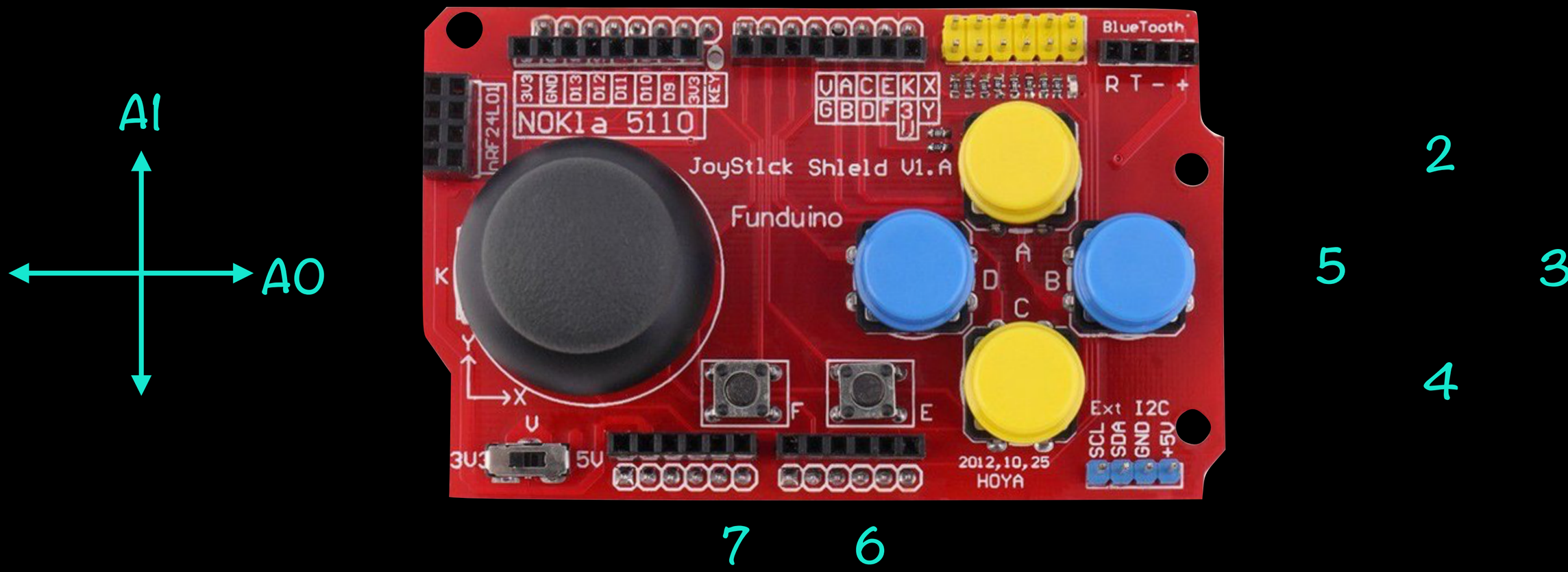
Projeto 08

Controle de Posição – Prática

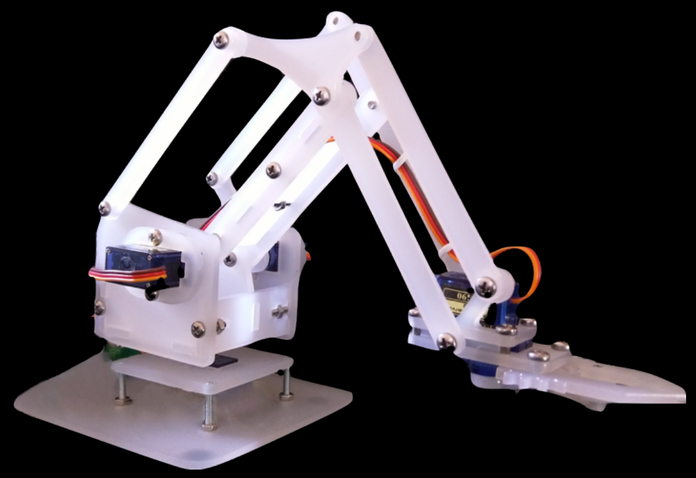
Jan K. S. – janks@puc-rio.br

ENG1419 – Programação de Microcontroladores

Testes Iniciais



A5



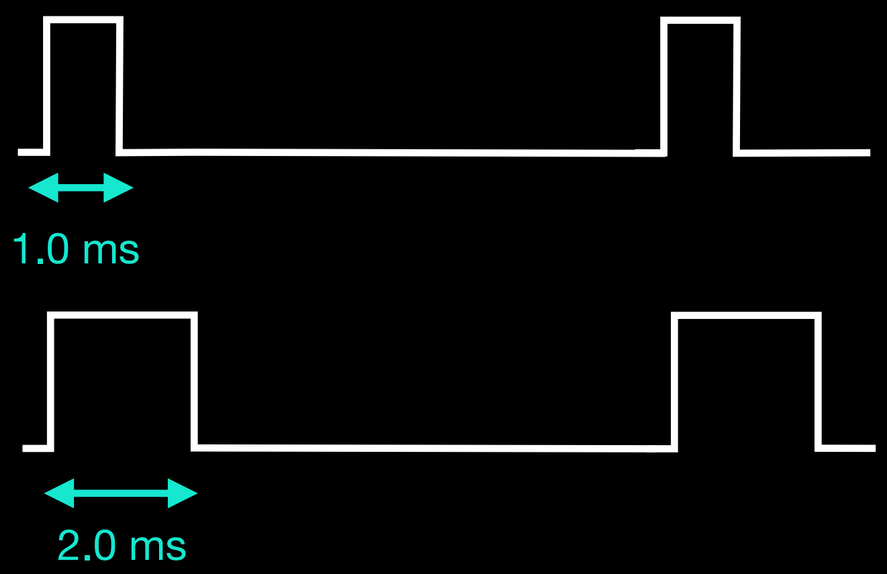
base: 12
ombro: 11
cotovelo: 10
garra: 9

Pinos Usados pelos Componentes

```
#include <EEPROM.h>        #include <Servo.h>

#include <meArm.h>

base.attach(pinoDaBase, 1000, 2000);
ombro.attach(pinoDoOmbro, 1000, 2000);
```



Teste da EEPROM e de Controle de Servo (Sem Usar meArm)

Testes Iniciais

Crie uma variável global indicando quantas vezes o Botão B (Direita) foi apertado, imprimindo a contagem via serial cada vez que ela aumentar.
↳ DICA: use a `GFButton`.

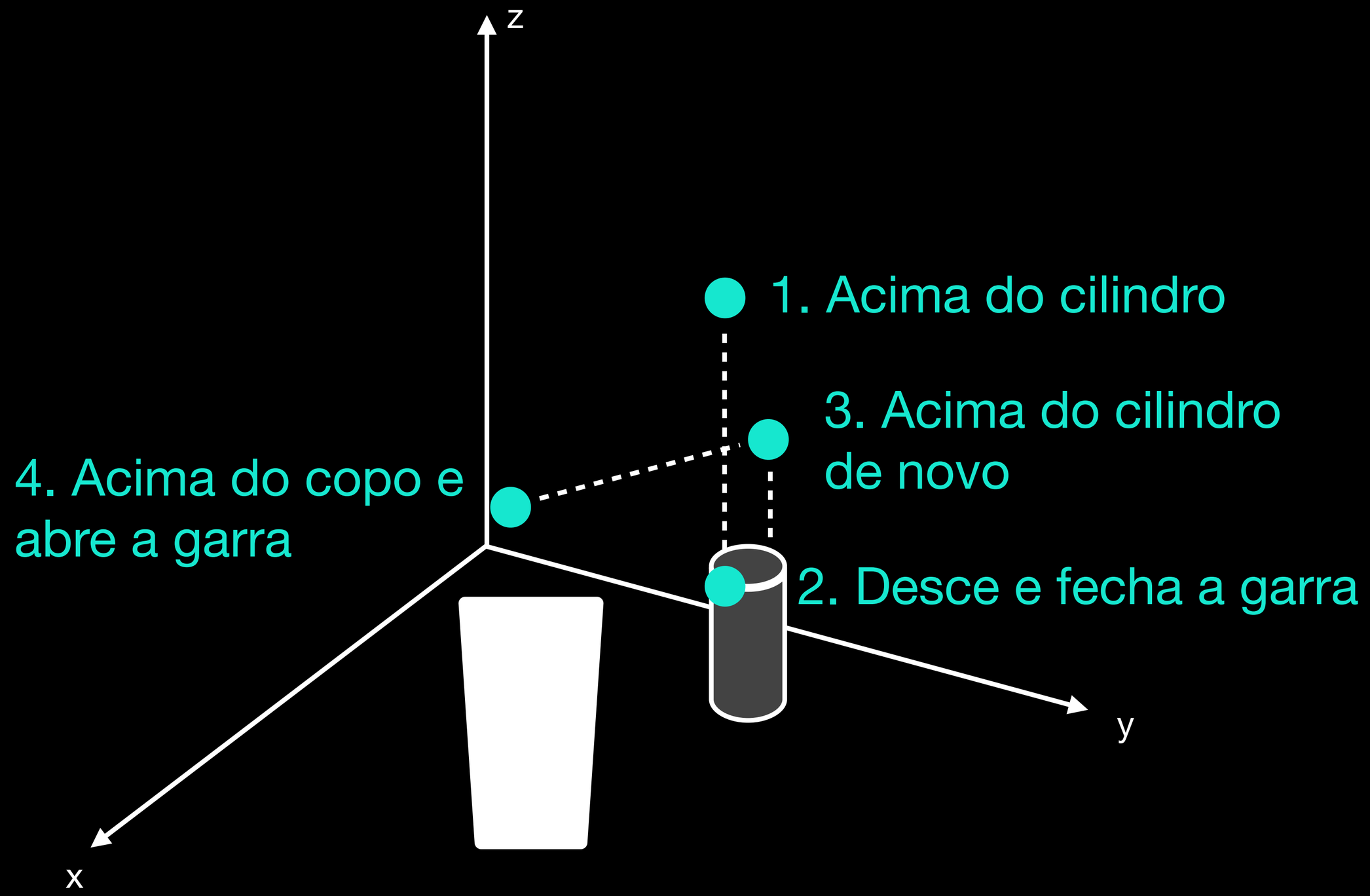
Ao apertar o Botão B (Direita), salve na EEPROM a contagem no endereço 0. Ao iniciar o programa, carregue essa contagem da memória como o valor inicial.

Ao girar o potenciômetro, varie o ângulo do servo da base entre 0 e 180°.
↳ DICA: use a função `map`. Defina os comprimentos de pulso na `attach`, conforme o slide anterior.

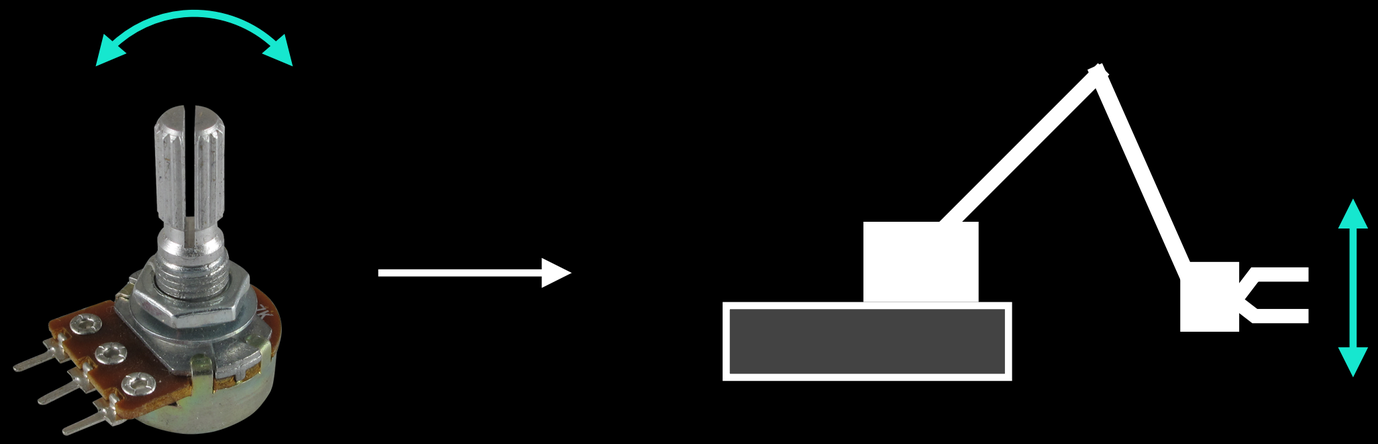
Dentro da loop: se o Botão A (Cima) estiver apertado, diminua um pouco o ângulo do servo do ombro, sem ultrapassar 45; se o Botão C (Baixo) estiver apertado, aumente um pouco esse ângulo, sem ultrapassar 135. Essas mudanças devem ser graduais, com um tempo pequeno de espera.
↳ DICA: crie uma variável global para o ângulo do servo do ombro. Use a `digitalRead` ou a `.isPressed()`.

Implementação

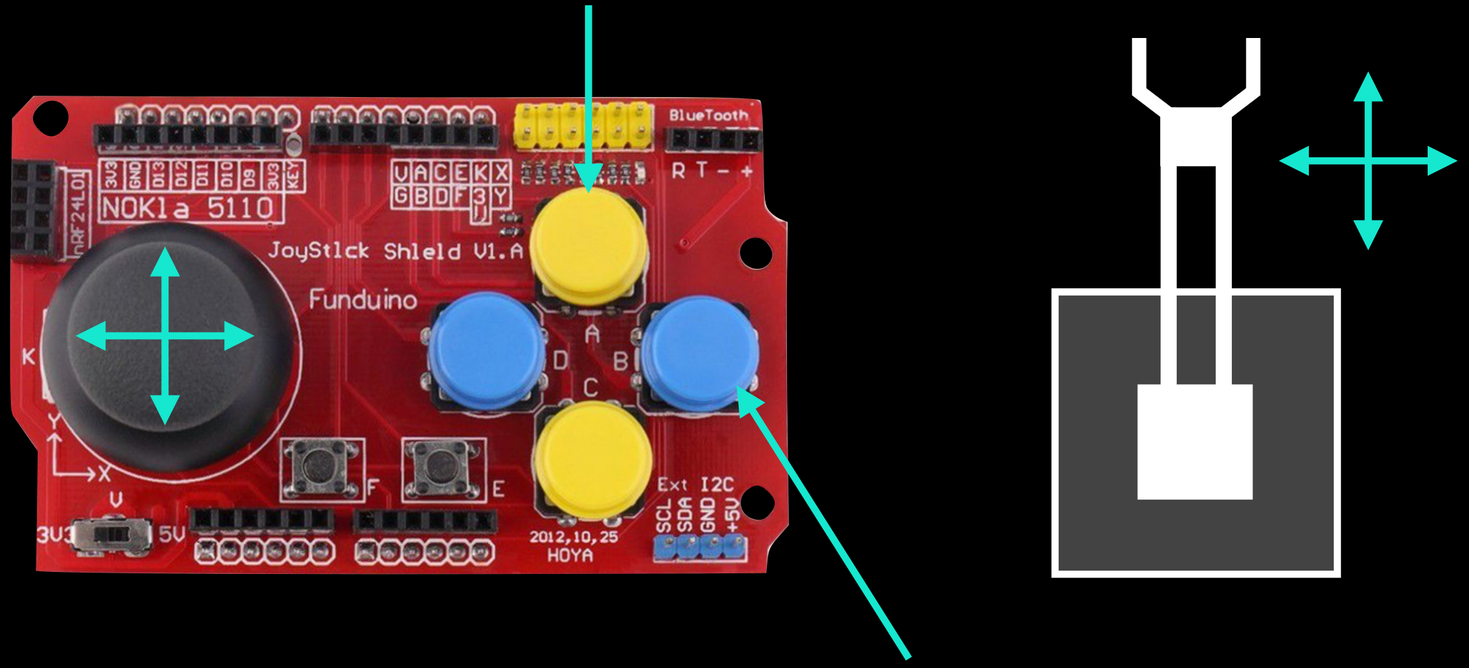
Controle do Braço Mecânico



Trajeto Desejado: Colocar o Pino dentro do Copo

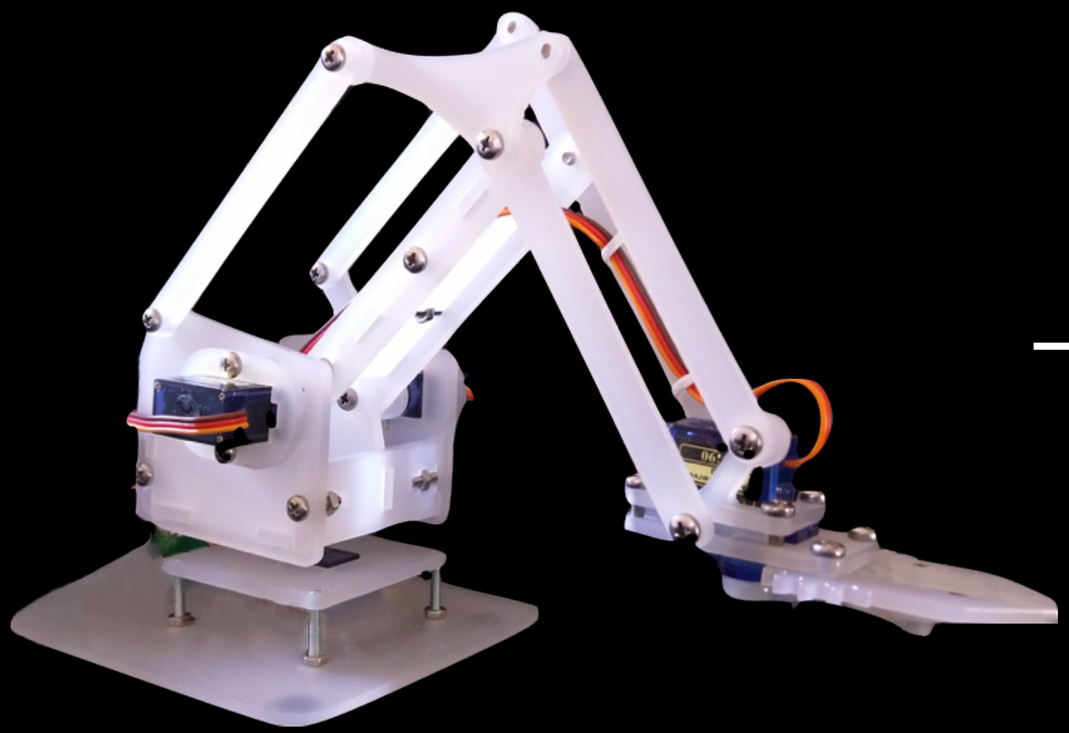


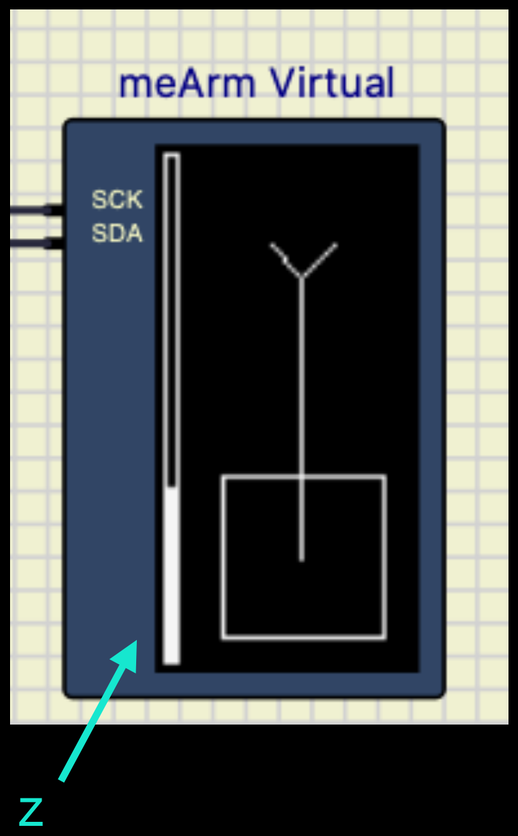
abre/fecha garra



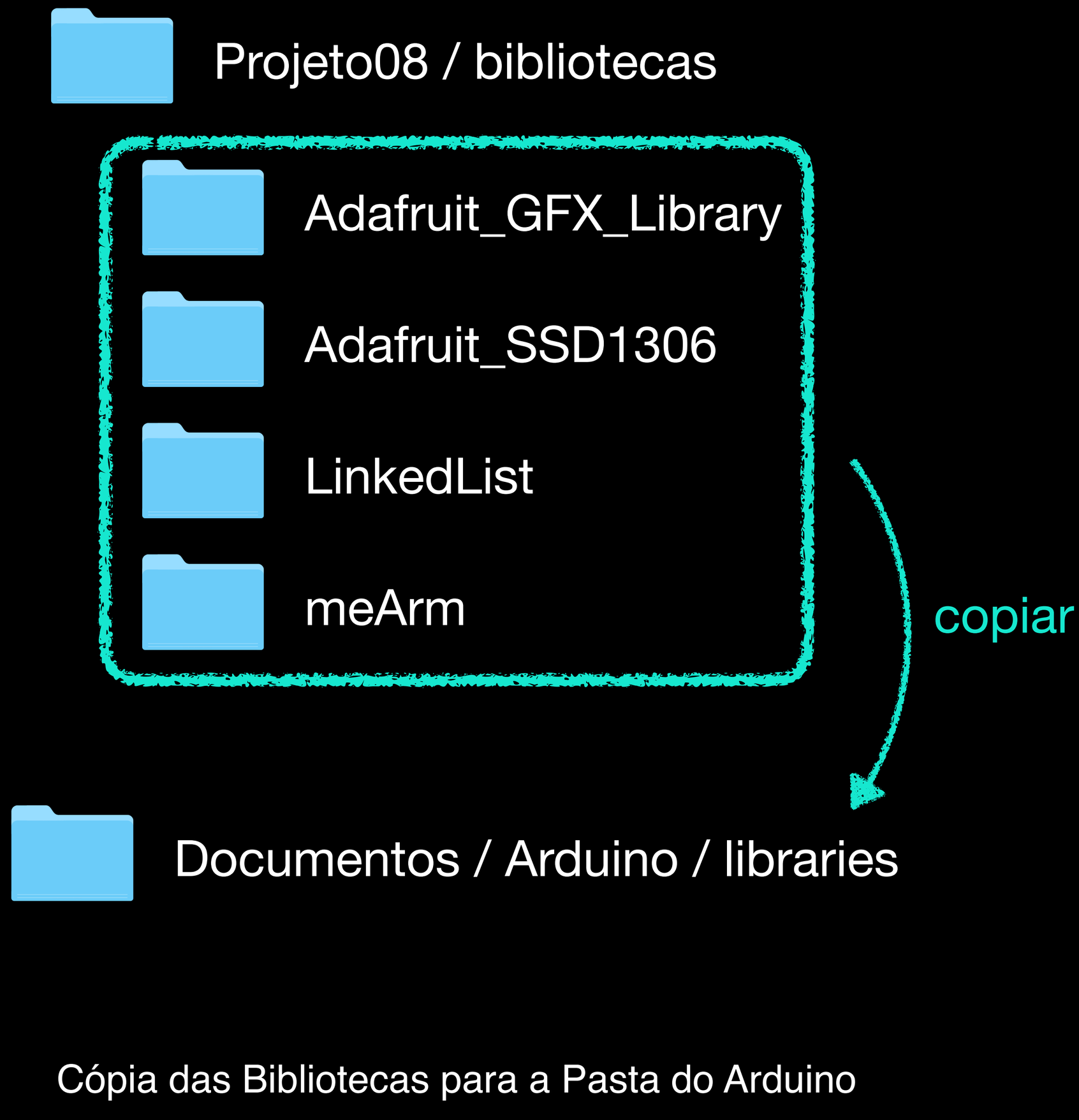
modo absoluto / relativo

Controle Analógico de 3 Coordenadas





Representação do Braço Mecânico no SimulIDE



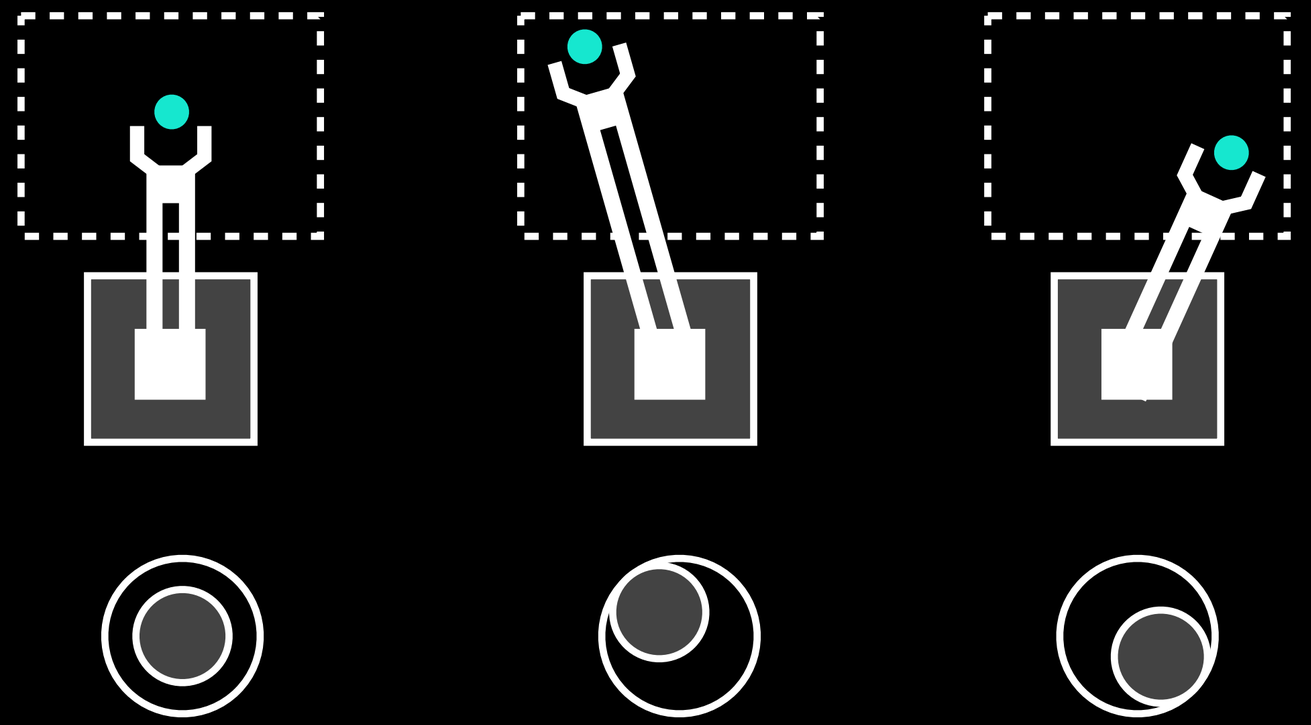
Cópia das Bibliotecas para a Pasta do Arduino

Movimento Absoluto

posição do joystick

=

posição do braço

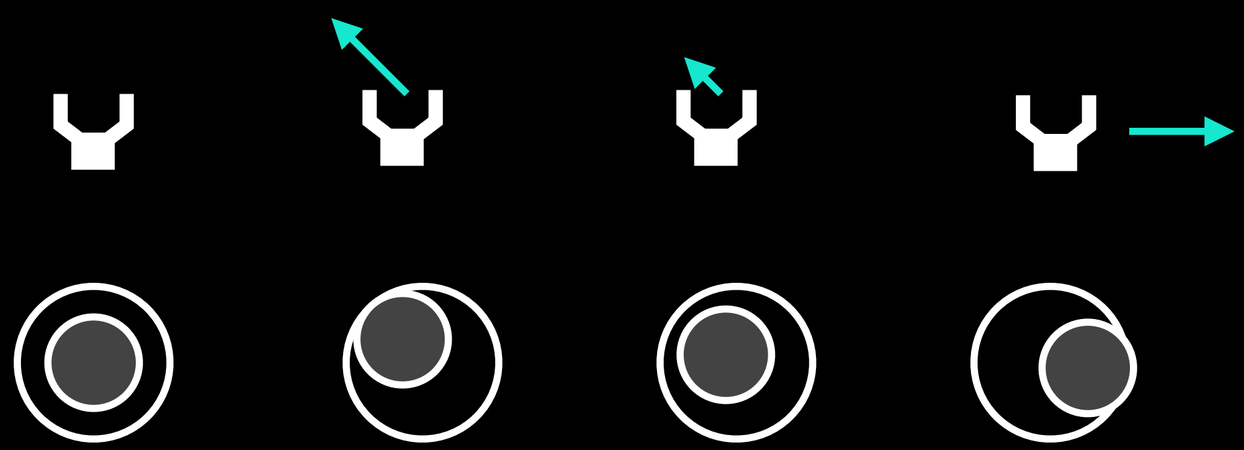


Movimento Relativo

posição do joystick

=

velocidade do braço



"Modo Absoluto": Ajuste da Posição

Movimento Absoluto

1. *Criar variáveis globais X, Y e Z.*
2. *Mapear valores do joystick para X (entre -150 e 150) e Y (entre 100 e 200), e do potenciômetro para Z (entre -30 e 100).*
3. *Mover a garra **SUAVEMENTE** para essas coordenadas.*

Movimento Relativo

1. *Usar as mesmas variáveis globais X, Y e Z do modo absoluto.*
2. *Mapear eixos do joystick para valores entre -10 e 10.*
3. *Usar valores como incremento de X e Y, com um delay de 50ms. Z permanece que nem no modo absoluto.*
4. *Garantir que X e Y não ultrapassem os limites do braço.*
5. *Imprimir variáveis pela Serial para verificar os valores e adicionar um fator de correção (se necessário).*
6. *Mover a garra **DIRETAMENTE** para essas coordenadas.*

Algoritmos para Movimento Absoluto e Relativo

Implementação

Copie as bibliotecas do zip para a pasta do Arduino.

Ao iniciar o programa, mova suavemente a garra para o ponto (0, 130, 0) e fecha a garra.
↪ DICA: use a biblioteca `meArm`.

Ao apertar o Botão A (Cima), alterne o estado da garra entre aberto e fechado.
↪ DICA: crie uma variável global tipo bool (true/false) para salvar esse estado.

Ao apertar o Botão B (Direita), alterne entre "modo absoluto" e "modo relativo" e imprima esse estado na serial.
↪ DICA: crie uma outra variável global para salvar esse modo.

No loop: se o modo for absoluto, ajuste as posições X e Y do braço de acordo com o joystick e a posição Z de acordo com o potenciômetro, conforme o algoritmo do slide anterior.

Se o modo for relativo, ajuste a velocidade em X e Y de acordo com o joystick e a posição Z de acordo com o potenciômetro, conforme o algoritmo do slide anterior.

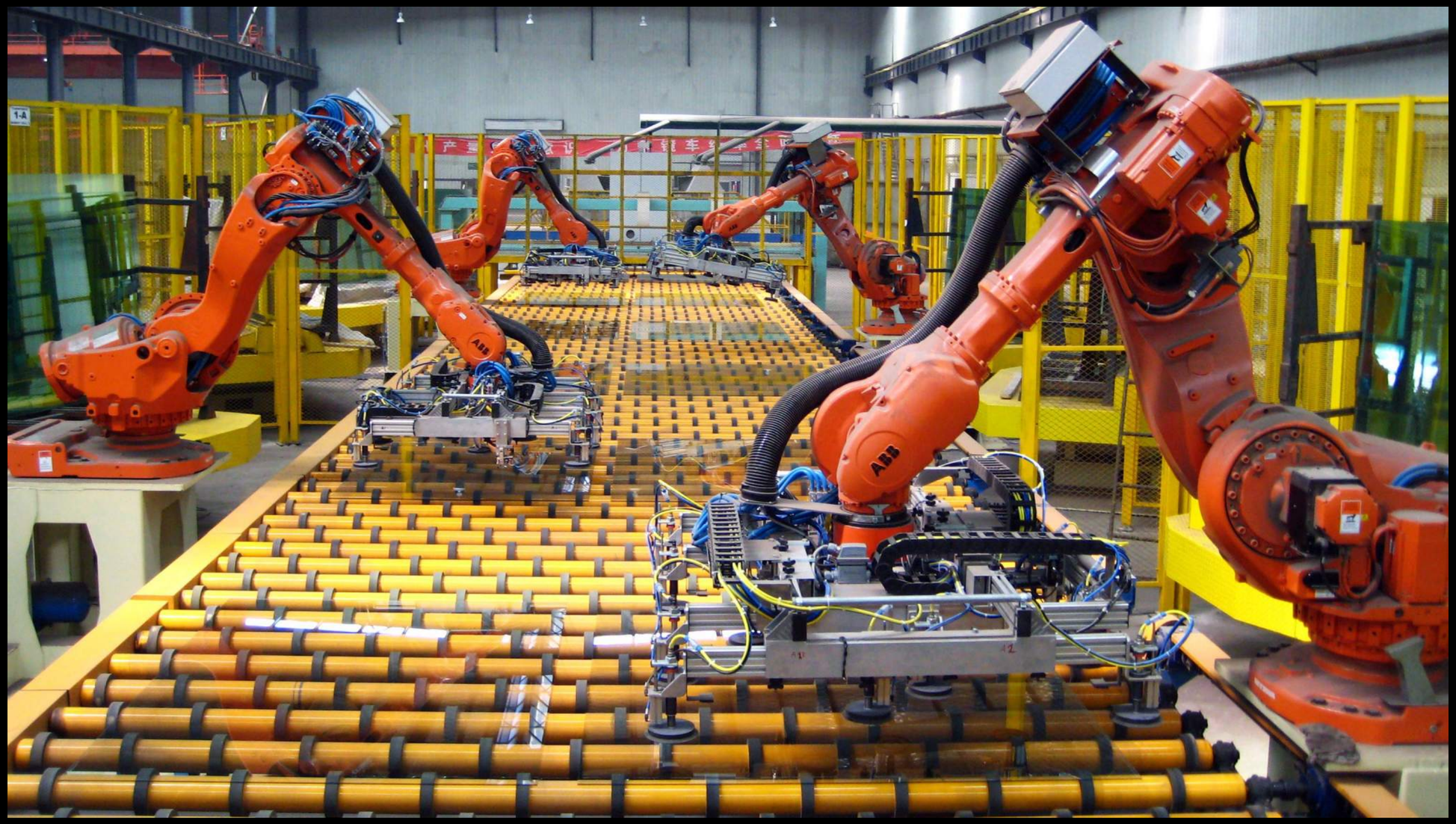
Aperfeiçoamento

p08b_implementacao.ino

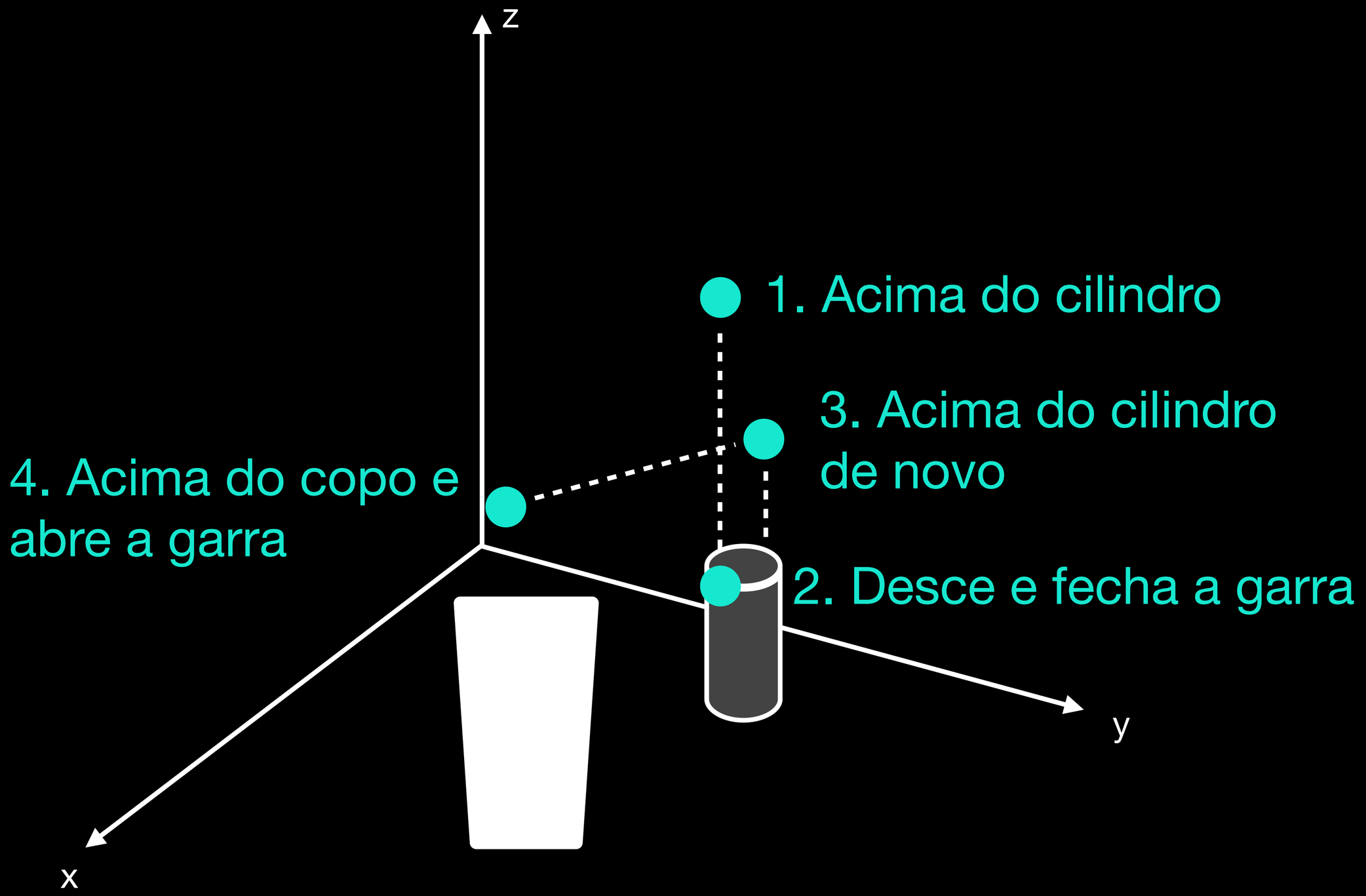
cópia

p08c_aperfeicoamento.ino

Cópia do Código da Implementação para o Aperfeiçoamento



Controle Automático do Braço



Trajeto Desejado: Colocar o Pino dentro do Copo

Botão C (Baixo)

move o braço para a posição

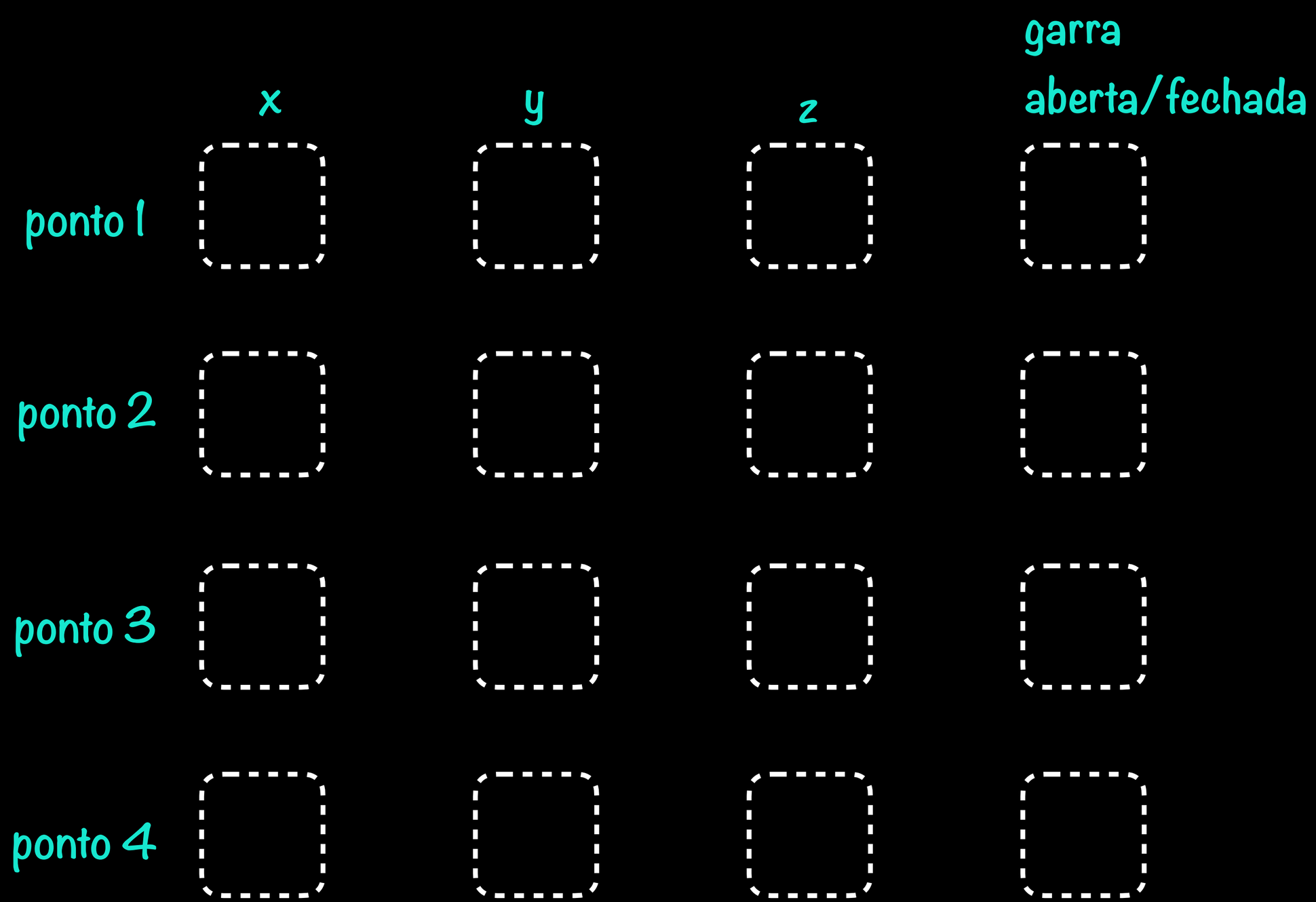
salva coordenadas e garra na matriz (linhas de 0 a 3)

Botão D (Esquerda)

move braço para as coordenadas salvas

Trajeto Desejado: Colocar o Pino dentro do Copo

```
float pontosSalvos[4][4];
```



Armazenamento dos Pontos

Aperfeiçoamento

Ao apertar o Botão C (Baixo), salve as coordenadas e o estado da garra (aberto/fechado) na matriz 4x4, variando o ponto atual de 0 a 3.

Ao apertar o Botão D (Esquerda), leia os dados salvos e mova o braço suavemente para cada uma das 4 posições, abrindo ou fechando a garra, com intervalos de 500 ms entre cada ponto.

Ao salvar o ponto, guarde a matriz dentro da EEPROM. Ao iniciar o programa, carregue a matriz da EEPROM.
↪ DICA: só é necessário escrever 2 linhas de código neste item.

Desafio Extra

cópia

p08c_aperfeicoamento.ino

p08c_desafio.ino

Cópia do Código do Aperfeiçoamento para o Desafio

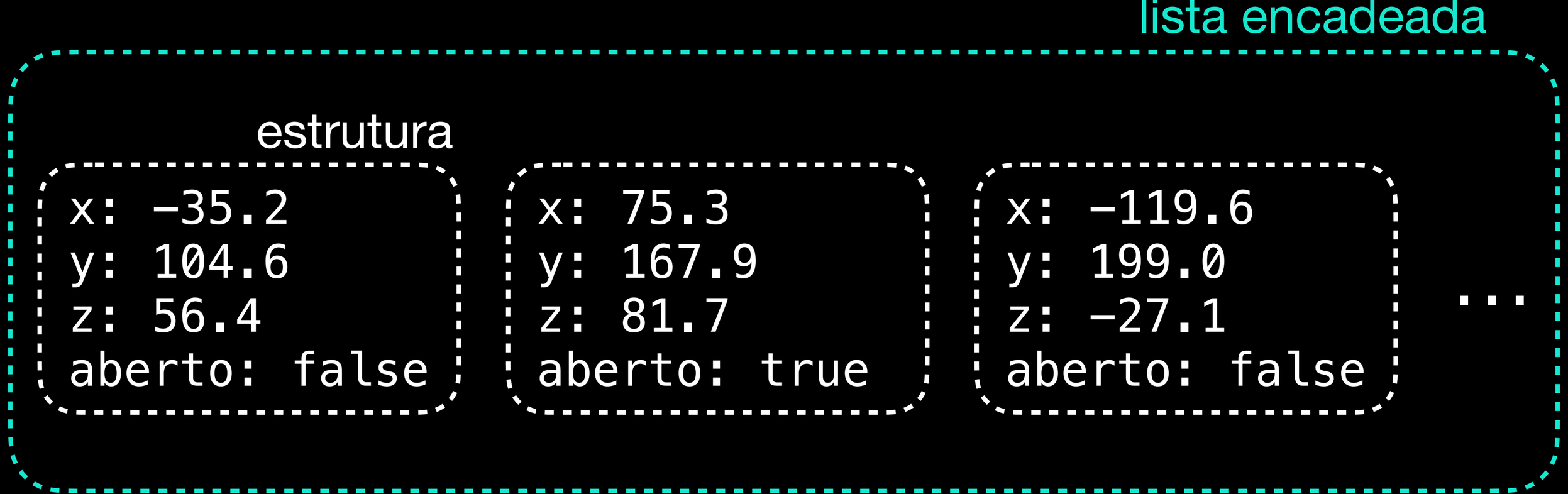
```
float pontosSalvos[4][4];
```

E se eu quiser mais posições?

```
float pontosSalvos[1000][4];
```

Desperdiça muita memória e não identifica direito os dados dentro da matriz.

Problemas com a Solução Matricial

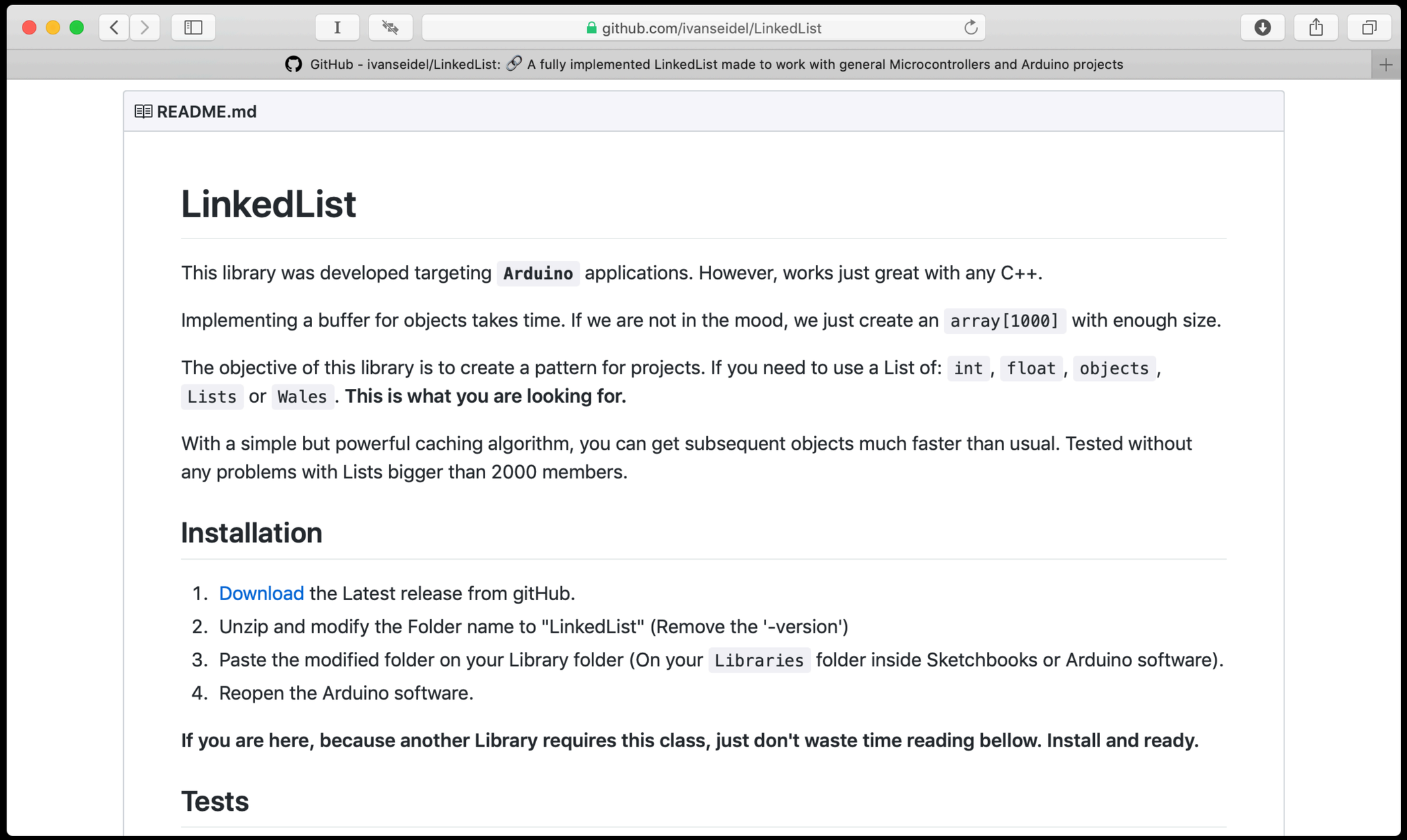


Solução Elegante com Estruturas de Dados e Lista Encadeada

```
struct Posicao {
    float x;
    float y;
    float z;
    bool garraAberta;
};

...

Posicao novaPosicao;
novaPosicao.x = 2.4;
novaPosicao.y = 162.3;
novaPosicao.z = -19.8;
novaPosicao.garraAberta = true;
```



Biblioteca LinkedList

```
#include <LinkedList.h>

// criação de lista para um certo tipo de elemento
LinkedList<float> listaDeDecimais;
LinkedList<bool> listaDeBooleanos;
LinkedList<Posicao> listaDeEstruturas;

...

// adiciona elemento no final da lista
listaDeEstruturas.add(elemento);

// acessa elemento da lista pelo índice (posição)
Posicao elemento = listaDeEstruturas.get(indice);

// total de elementos
int total = listaDeEstruturas.size();

// remove todos os elementos
listaDeEstruturas.clear();
```

Exemplo de Uso da Biblioteca LinkedList

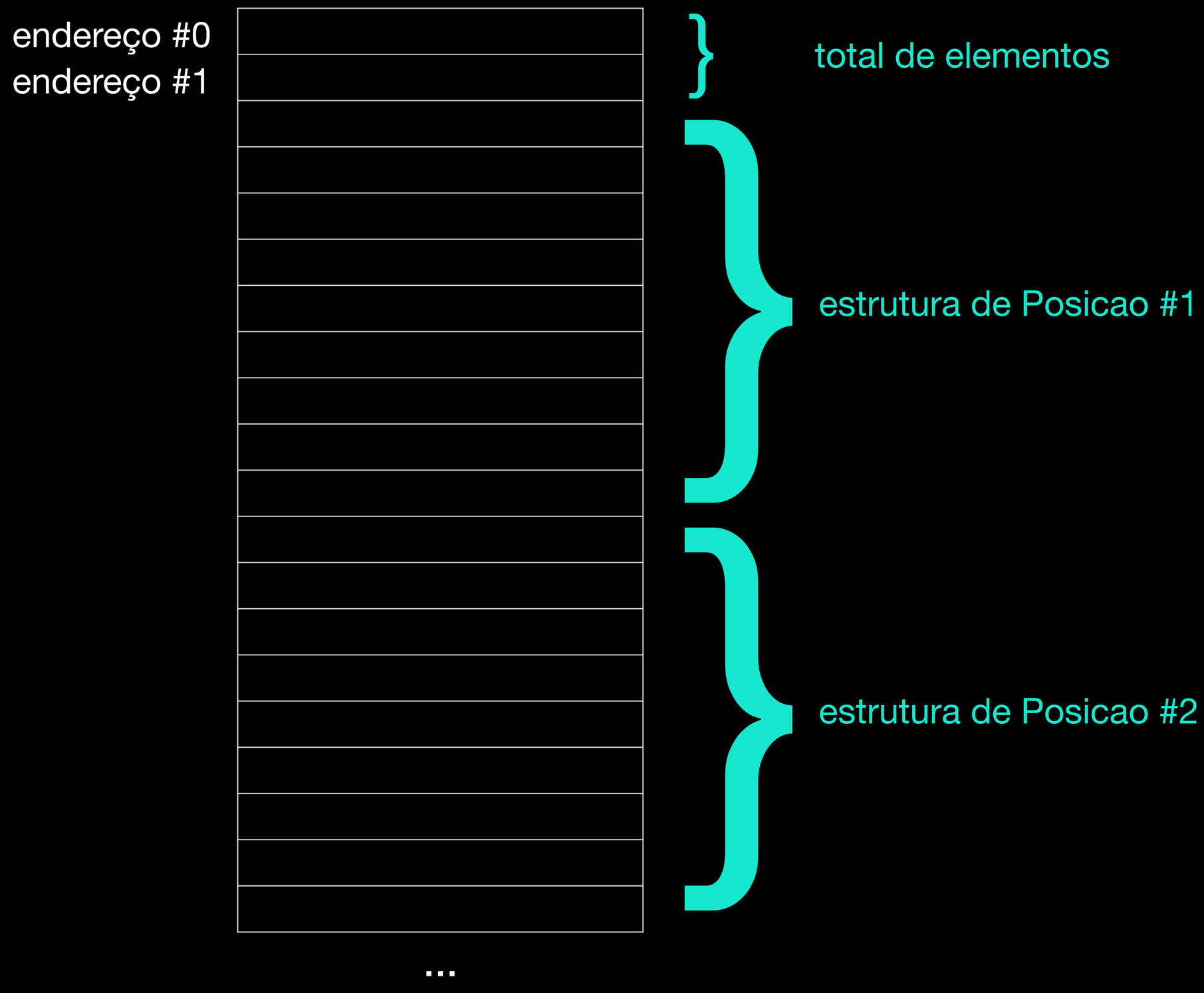
```
Posicao novaPosicao;
novaPosicao.x = 2.4;
novaPosicao.y = 162.3;
novaPosicao.z = -19.8;
novaPosicao.garraAberta = true;

EEPROM.put(endereco, novaPosicao); // funciona!
```

```
LinkedList<Posicao> lista;
lista.add(novaPosicao1);
lista.add(novaPosicao2);

EEPROM.put(endereco, lista); // não funciona!
```

Criação da Lista em C++ com Tipo de Elemento



Nova Organização de Dados na EEPROM

Desafio Extra

Adicione a definição da estrutura e variável global de lista encadeada. Em seguida, modifique o código do Aperfeiçoamento para salvar as estruturas de posições na lista em vez de na matriz.

Modifique o código da reprodução dos pontos salvos para percorrer a lista de posições em vez da matriz.

Modifique o código do Aperfeiçoamento para salvar o total de elementos e cada estrutura na memória.
↳ DICA: usa a função `sizeof` para calcular quantos bytes cada estrutura vai ocupar na memória.

Modifique o código para adicionar os dados da EEPROM na lista encadeada global ao iniciar o programa.

Ao apertar o botão E (porta 6), apague a lista de posições salvas e atualize o total na EEPROM.

janks.link/micro/projeto08.zip

Material do Projeto 08