unesp UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE GUARATINGUETÁ

SILAS DOS SANTOS VERGÍLIO

Uso de robótica educacional para ensino de lógica para alunos do ensino médio.

Guaratinguetá - SP
2018

Silas dos Santos Vergilio

Uso de robótica educacional para ensino de lógica para alunos do ensino médio.

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação em Engenharia Elétrica da Faculdade de Engenharia do Campus de Guaratinguetá, Universidade Estadual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Elétrica.

Orientador (a): Leonardo Mesquita

Guaratinguetá - SP
2018

Vergílio, Silas dos Santos
V497u Uso de robótica educacional para ensino de lógica para alunos do ensino médio / Silas dos Santos Vergílio – Guaratinguetá, 2018.
51 f : il.
Bibliografia: f. 50

Trabalho de Graduação em Engenharia Elétrica – Universidade Estadual Paulista, Faculdade de Engenharia de Guaratinguetá, 2018.
Orientador: Prof. Dr. Leonardo Mesquita

1. Arduino (Controlador programável) 2. Robótica. 3. Ensino médio. I. Título.

CDU 681.5

Luciana Máximo
Bibliotecária-CRB-8/3595

unesp UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE GUARATINGUETÁ

SILAS DOS SANTOS VERGÍLIO

ESTE TRABALHO DE GRADUAÇÃO FOI JULGADO ADEQUADO COMO PARTE DO REQUISITO PARA A OBTENÇÃO DO DIPLOMA DE
"GRADUADO EM ENGENHARIA ELÉTRICA"

APROVADO EM SUA FORMA FINAL PELO CONSELHO DE CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Profa. Dra. Paloma Maria Silva Rocha Rizol
Coordenadora

BANCA EXAMINADORA:

Prof. Dr. LEONARDO MESQUITA
Orientador/UNESP-FEG

Prof. Dr. FERNANDO RIBEIRO FILADELFO
UNESP-FEG

Prof. Eng. JOSÉ MARCELO DE ASSIS WENDLING JUNIOR
UNESP-CTIG

Dezembro de 2018

dedico este trabalho de modo especial, à equipe FEG-Robótica e à todas as crianças envolvidas na First Lego League.

AGRADECIMENTOS

Em primeiro lugar agradeço a minha família pela constante paciência que tiveram neste longo processo mesmo que sem entender completamente ele.

Ao meu irmão Samuel que teve a coragem de compreender toda a minha jornada a fim de me impedir de desistir, por ter acreditado em mim mesmo quando eu mesmo não acreditei.

Ao meu irmão da faculdade, Yuri, que complementou todas as minhas ideias com uma visão realista e prática a qual ainda tenho muito o que aprender.

À Isabela, poética companheira que me ajudou tanto em momentos técnicos quanto em momento em que precisei ver o melhor em mim mesmo.

Ao meu grande colega e sucessor de liderança, Matheus, pela criativa ajuda em momentos em que cada detalhe era importante.

Á Ravena que me acompanhou de perto e de longe, nos momentos fáceis e difíceis e me proporcionou uma necessária perspectiva.

Ao meu grande parceiro, Francisco, por estar sempre ao meu lado, desde quando tudo começou trazendo coerência e relevância a objetivos diversas vezes distantes.

Ao meu orientador Leonardo Mesquita, que me acompanhou por diversos projetos e me deu a necessária experiência para entrar no mercado de trabalho.

A Diana, que foi a ponte entre dois mundos que muitas vezes não se conversam, sua amizade é a maior prova da importância da verdadeira interdisciplinaridade.

A todas as pessoas que durante a graduação eu tive o prazer de ajudar e ensinar, por terem me inspirado a tentar melhorar os processos de aprendizado.

“Na maturidade do novo milênio, espero que salvemos as florestas tropicais deste planeta, a camada de ozônio, os rios e correntes, os oceanos, e salvaremos também, espero, as mentes e corações das pessoas que vivem aqui.”

Andrew Solomon

RESUMO

Este trabalho fundamenta-se no desenvolvimento de um hardware eletrônico, com base em Arduino para o uso em sala de aula de turmas do ensino médio com o intuito de ensinar lógica de uma maneira intuitiva e interessante. Este projeto se propôs a criar uma solução simples e versátil para ensino de uma maneira mais visual e prática do que outros trabalhos estudados. O protótipo consiste em uma matriz de LED's de alto brilho, eles podem assumir diferentes cores e são utilizados para representar diferentes problemas de lógica, o protótipo contém também uma biblioteca que é utilizada em conjunto para resolver diferentes desafios. A ideia central deste projeto é criar um experimento que possa ser usado repetidamente de maneiras diferentes, uma vez que cada um dos elementos da matriz pode assumir diversas cores, indicando diferentes tipos de obstáculos ou trajetos, enquanto ainda é possível diferenciar um ou mais elementos de movimentação. O dispositivo funciona como uma abstração de um trajeto, ou múltiplos, que um robô físico poderia seguir, onde cada configuração dos elementos da matriz está associado a uma série de elementos lógicos educacionais. Embora o termo 'lógica' empregado aqui possa se enveredar por caminhos dos mais abstratos aos mais concretos, deseja-se aqui focar principalmente na lógica de programação, largamente necessária nos dias de hoje para formar quase que qualquer tipo de profissional. Para completar tal objetivo há medidas tanto do ponto de vista do desenvolvimento técnico, ao criar uma biblioteca específica para o hardware desenhado, quanto do ponto de vista da metodologia aplicada que irá visar em aproveitar mais do conhecimento prévio do aluno que já é multidisciplinar.

PALAVRAS-CHAVE: Arduino. Robótica. Educação. Metodologia. Lógica. Ensino Médio.

ABSTRACT

This work is about the development of an electonics hardware using Arduino to use it inside high school classrooms in order to teach logic in a intuitive and interesting manner. This project proposes to create a simple yet robust solution to teach in a more practical and visual way than other works previously studied. The main idea of this project is to create an experiment which can be repeatedly used in different ways, since each of the elements of the proposed led matrix can assume several different colors, which may indicate obstacles or paths. The device will function as an abstraction of one or more movements which the robot may follow, where each of the led's configuration will represent a different logic feature to be teached. Even though the term 'logic' which is often used here may be translated as something more abstract or often more concrete, but is it desired on this project to focus on programming language, a skill largely used nowadays on the becomming of most professionals, which is why this project will not focus on students who already like tbe field of math or science. In order to complete such goal, measures were taken from electronics development to the development of an Arduino library and also the teaching method which aims to reach each student previous knowlegde and use it as fuel to learn new skills

KEYWORDS: Arduino. Robotics. Education. Logic. High School.

LISTA DE TABELAS

Tabela 1 – Características eletrônicas da placa Arduino Leonardo 13
Tabela 2 – Características de luminosidade do LED 5050 13
Tabela 3 – Pinos WS2812 14
Tabela 4 – Consumo fita led WS2812 15
Tabela 5 – Custos do projeto.....21
Tabela 6 – Códigos de cada cor configurada.....29

SUMÁRIO

1	**INTRODUÇÃO**	10
1.1	MOTIVAÇÃO	11
1.2	OBJETIVOS	11
2	**DESENVOLVIMENTO TÉCNICO**	12
2.1	ESTRUTURA ELETRÔNICA	12
2.2	ESTRUTURA FÍSICA	16
2.3	ESTRUTURA DO SOFTWARE	21
3	**DESENVOLVIMENTO DA METODOLOGIA**	33
3.1	OBJETIVOS GERAIS	33
3.2	SEÇÕES DO MÉTODO	33
3.2.1	CONTEXTUALIZAÇÃO	33
3.2.2	SINTAXE	33
3.2.3	FLUXOGRAMA	34
3.2.4	CONFIGURAÇÃO INICIAL	34
3.3	EXEMPLO PRÁTICO	34
3.3.1	CONTEXTUALIZAÇÃO	34
3.3.2	SINTAXE	35
3.3.3	FLUXOGRAMA	35
3.3.4	CONFIGURAÇÃO INICIAL	37
4	**CONCLUSÃO**	38
	REFERÊNCIAS	39

1 INTRODUÇÃO

Lógica é uma palavra muito usada no dia a dia e que corre o risco de perder seu significado conforme é usada. É costume usar a palavra como um constraste de uma ação “lógica” a uma ação “ilógica”, ou procedimento “lógico” em oposição a outro “ilógico”, o que no final das contas é como traduzir para uma palavra como “razoável”.

Segundo COPI o estudo de lógica é o estudo dos métodos e princípio usados para distinguir o racionício correto do incorreto.

A lógica acaba por ter diferentes nuâncias, tomando forma tanto como um viés da matemática, o que exige uma linguagem própria com símbolos específicos e regras para tratamento de problemas, algo que foi muito explorado por Bertrand Russel em seu livro Principia mathematica do ponto de vista mais puro de lógica. Todavia a lógica não precisa sempre usar novos símbolos em relação a nossa linguagem, ela também pode ser estudada dentro do discurso formal, trazendo definições próprias que aprendemos nas aulas de gramática dentro do ciclo básico de ensino. Logo no começo do estudo de introdução à lógica COPI usa o seguinte parágrafo para introduzir algumas características importantes ao lidar com lógica:

> “É necessário distinguir as sentenças das proposições para cuja afirmação elas podem ser usadas. Duas sentenças (ou orações declarativas) que constituem claramente duas orações distintas, porque consistem de diferentes palavras, dispostas de modo diferente, podem ter o mesmo significado, no mesmo contexto, e expressar a mesma proposição”. (COPI,I.M., 1978, p. 22)

O que traz a tona alguns termos como proposições, sentenças, orações, declarações, contexto, entre outros, que são ferramentas para análise e resolução de problemas de lógica, portanto mesmo que a lógica matemática clássica tenha outros símbolos e ferramentas, o que há em comum é o uso da linguagem como principal instrumento.

A tecnologia está cada vez mais presente no dia a dia e junto a ela novas habilidades são esperadas de todos os profissionais de diferentes níveis. De acordo com OLIVEIRA é fundamental se adaptar a esta tecnologia no processo de inserção de um indivíduo na sociedade, não só seu manuseio, mas de fato entender sua dimensão, implicações, vantagens e desvantagens para a então apropriação da tecnologia.

Arduino é uma plataforma *open-hardware*, ou seja, todo o projeto de hardware e software são divulgados gratuitamente, gerando placas de baixo custo bastante versáteis, seu aprendizado facilitado também faz parte da proposta da plataforma. Seu uso em educação como em ALVES demonstra o teor de uma série de trabalhos que possibilitaram pessoas não

ligadas diretamente a tecnologia a transformar ideias em soluções tecnológicas sem a necessidade de ter uma formação técnica.

A programação de computadores entra como uma linguagem comum para lidar com diferentes tecnologias e que de maneira mais prática que a linguagem da matemática formal tem o potencial de exercitar o uso de lógica enquanto se apropria de ferramentas importantes do profissional deste século. A respeito do ensino de programação, ZANETI diz:

> “O processo de aprendizagem dos conceitos iniciais de programação é complexo e, muitas vezes, necessita de um nível de abstração que não está presente na maioria dos alunos iniciantes, havendo a necessidade de se criar um ambiente mais diversificado e motivador para o aluno”. (ZANETTI , H. A. P., 2015)

1.1 MOTIVAÇÃO

Na tentativa de ensinar robótica educacional sem o auxílio de algum kit educacional, usando apenas instrumentos tradicionais como compoenentes, fios, protoboard, há o desafio de ensinar as ferramentas técnicas e lógicas para o uso dos componentes eletrônicos de maneira apropriada em sua forma mais clássica; ensinar princípios de mecânica associados ás montagens a fim de dar contextos ás aulas além de finalmente ensinar tanto lógica quanto a sua sintaxe, por mais que a linguagem Arduino facilite em muitos aspectos a manipulação de seu microcontrolador, ainda é mais um desafio junto aos outros. Na dinâmica de aula pelo fato de existir a possibilidade de pegar um código pronto e obter o resultado esperado do projeto como um todo, na ausência de tempo e ferramentas, o ensino de programação, tanto lógica quanto sintaxe ficam deixados de lado.

1.2 OBJETIVOS

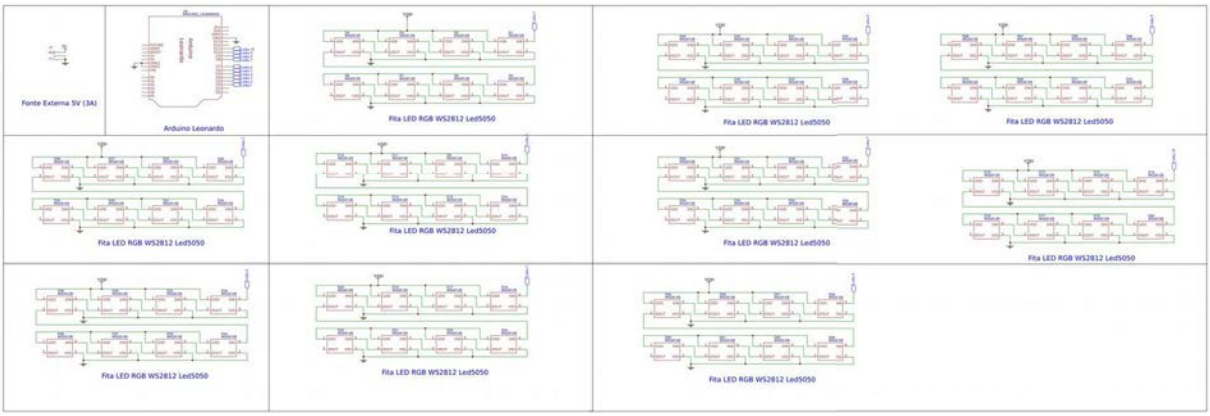
O objetivo deste trabalho é desenvolver um versátil dispositivo para ensino de lógica de programação em disciplinas de ensino médio. Este dispositivo consistirá principalmente de uma matriz de leds em que é possível controlar intensidade de iluminação e cor de cada led individualmente. Associado ao dispositivo haverá também um material didático e uma biblioteca feita em Arduino para auxiliar na condução do material. Todo o desenvolvimento será destino a criar um produto *open source* em todos seus aspectos, usando materiais de fácil acesso, programação disponibilizada e um material didático que possa ser expandido e reutilizado.

2 DESENVOLVIMENTO TÉCNICO

2.1 ESTRUTURA ELETRÔNICA

O circuito adotado neste trabalho consiste em um controlador principal, neste caso a placa Arduino Leonardo, as características necessárias para esta placa vão além do seu potencial técnico. A arquitetura de todo o projeto Arduino é voltado a tornar o desenvolvimento de projetos algo mais acessível e prático, enquanto esta característica não é tão crucial para um engenheiro; dado o propósito educacional deste projeto, espera-se que os próprios usuários do projeto possam interagir de maneira facilitada com o mesmo. O outro elemento importante deste projeto é a combinação do led 5050 como circuito integrado WS2812. Consdeirando o objetivo do projeto de criar uma matriz de leds interativa, cada linha é criada a partir de uma fita de led composta por 10 leds, usando 10 fitas alcançamos uma matriz quadrada de módulo 10. O circuito não conta com sensores, compensando com um controle individual de cada led com alta resolução de cores, podendo assim cumrpri seu papel como material de ensino para lógica de programação. O circuito esquemático completo é apresentado na Figura 1. Quanto a alimentação existem duas fontes previstas, uma para a matriz de led, e outra para a placa Arduino.

Figura 1 – Esquemático do ciruito proposto.



Fonte: Produção do próprio autor

A estrutura eletrônica é centralizada no hardware aberto Arduino visto que um dos objetivos deste trabalho é que ele seja compartilhado com outros professores para que possa crescer, a natureza *open source* da arquitetura do hardware e do software contribui muito, dentre as possíveis placas foi adotada a placa Arduino Leonardo, a placa conta com o

microcontrolador ATMega32u4, suas características eletrônicas mais relevantes para este projeto são elencadas na Tabela 1.

Tabela 1 – Características eletrônicas da placa Arduino Leonardo

Característica	Valor
Microcontrolador	ATMega32u4
Pinos digitais de E/S	20
Clock	16MHz
Tensão de entrada	6V - 20V
Tensão de operação	5V
Conexão com computador	Micro USB
Corrente DC no pino 5V	50mA

Fonte: Adaptado de Arduino.cc (2018)

Embora o modelo Arduino Uno seja muito usado em inclusive referências citadas neste trabalho, a placa aqui adotada traz como principal vantagem o número de pinos digitais na placa, esta contém 6 pinos disponíveis a mais que a versão mais comum do Arduino. As características de PWM, entradas analógicas, comunicação de dois fios, entre outros pontos, não são tão relevantes para o projeto adotado, todavia são pontos que podem ser úteis em uma posterior implementação de novas funcionalidades por outros desenvolvedores que assim acharem necessário.

A seguir devemos compreender o periférico mais importante de todo o desenvolvimento que será a fita de leds WS2812. O intuito deste CI é proporcionar um controle individual de cada led em uma fita de leds, incluindo as suas intensidades nas cores vermelho, verde e azul a fim de criar diferentes cores em cada um dos leds, usado em conjunto com o led modelo 5050, garante uma intensa luminosidade (também controlável) e assim clareza das cores apresentadas, característica importante para garantir a identificação de diferentes elementos lógicos no material de ensino. A Tabela 2 abaixo apresenta as características de luminosidade de cada uma das cores do led.

Tabela 2 – Características de luminosidade do LED 5050

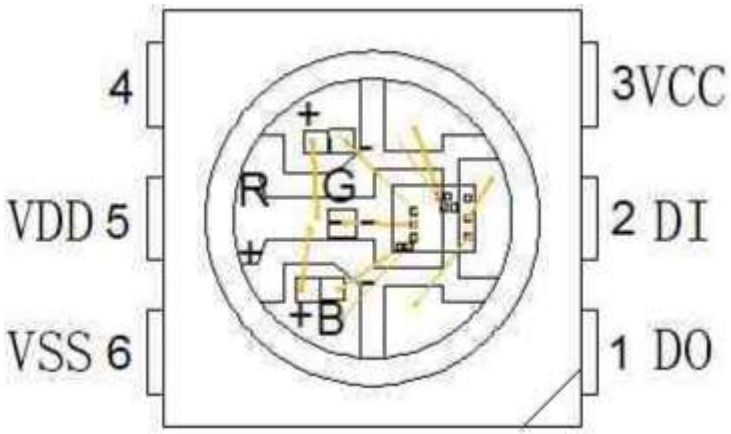
Cores	Luminosidade (mcd)
Vermelho	100
Verde	400
Azul	100

Fonte: Adaptado de Yuanlei (2018)

Um led comum tem uma luminosidade de 50mcd em média. A cor verde deste led em especial apresenta uma alta luminosidade em relação a outros componentes do mesmo tipo.

O funcionamento do CI WS2812 consiste em enviar uma completa informação a um led e cada led repassa a informação aos outros leds, permitindo controlar fitas de tamanho variado O esquemático de entradas e saídas do CI é apresentado na Figura 2.

Figura 2 – Pinagem do CI que controla os leds da matriz.



Fonte: Yuanlei (2018)

A função de cada pino é descrita na Tabela 3.

Tabela 3 – Pinos WS2812

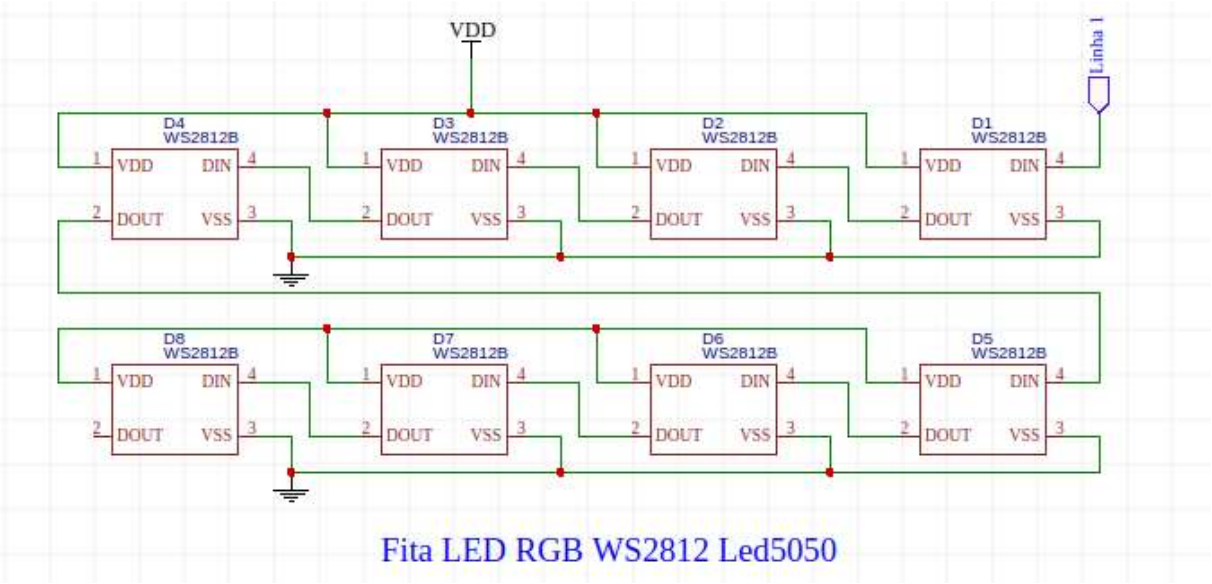
Número	Símbolo	Descrição
1	DOUT	Saída de sinal de controle entre leds
2	DIN	Entrada de sinal de controle de led
3	VCC	Alimentação do circuito de controle
4	NC	Não conectado
5	VDD	Alimentação dos leds
6	VSS	Terra

Fonte: Adaptado de Yuanlei (2018)

O funcionamento do CI WS2812 consiste em enviar uma completa informação a um led e cada led repassa a informação aos outros leds. A informação, de acordo com o datahseet, deve ser enviada em pacotes de 24bits, todos pelo pino DI do primeiro led, ele tem um circuito para que a distorção não se acumule na passagem entre os componentes, garantindo qualidade no controle individual das luzes até mesmo em fitas com maior dimensão. Ainda no critério de garantir a qualidade do projeto, cada um dos componentes tem na entrada de sua alimentação um capacitor, uma vez que uma grande oscilação na alimentação do led por gerar

comportamentos indesejados. Os 24 bits enviados de informação para cada um dos leds são divididos em três grupos de 8 bits onde cada byte representa uma das cores dentro do padrão RGB (*reg*, *green*, *blue* ou seja vermelho, verde, azul). Por exemplo, é possível enviar 3 palavras de 24 bits, cada uma descrevendo o comportamento de um led entre três; graças a um circuito interno de remodelagem e amplificação entre a entrada e saída de cada led. A Figura 3 apresenta um exemplo de ligação de uma fita de led usando 8 leds como exemplo.

Figura 3 – Exemplo de circuito de ligação entre leds 5050 com CI WS2812.



Fonte: Produção do próprio autor.

Quanto a alimentação a primeira coisa a ser observada é a Tabela 4 que indica o consumo de energia relacionado ao led 5050 com o CI WS2812.

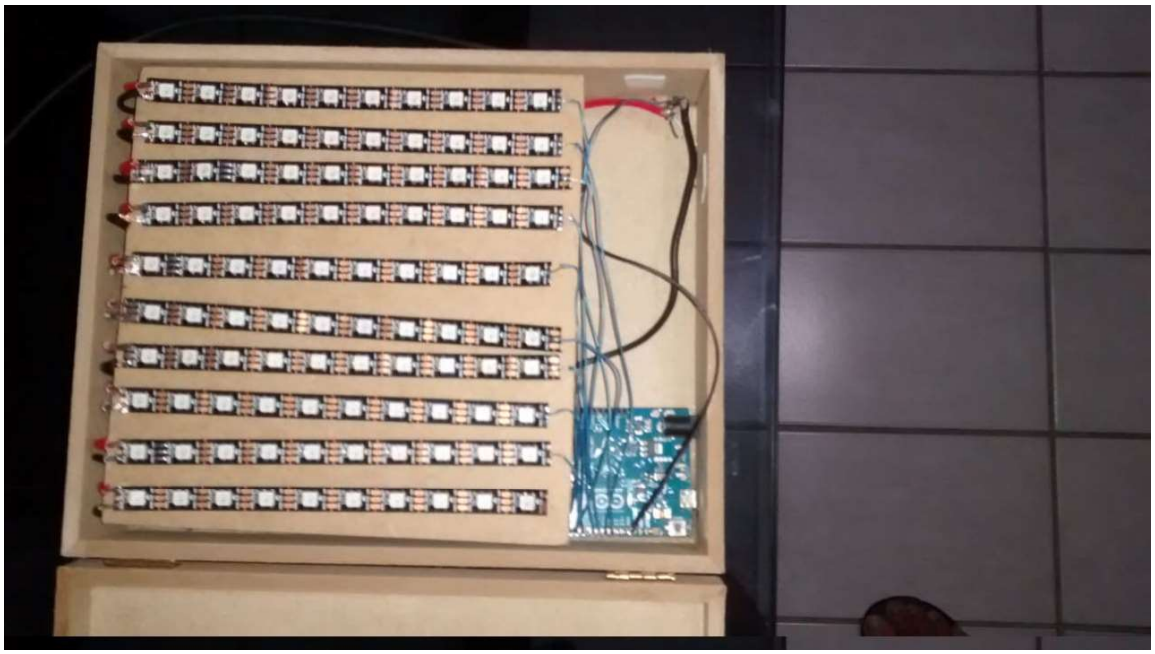
Tabela 4 – Consumo fita led WS2812

Modelo	W/Rolo	W/Metro	A/Rolo	A/metro
SMD5050 150 LEDs por rolo (30 LEDs / m)	36W	7,2W	3A	0,6A
SMD5050 com 300 LEDs por rolo (60 LEDs /m)	72W	14,4W	6A	1,2A

Fonte: Adaptado de Revoled (2018)

O projeto prevê, como indica a Figura 1, 10 fitas de 10 leds cada, totalizando 100 leds, a alimentação do circuito deve ser planejada de modo que satisfaça a situação que mais demanda energia que seria em que todos os leds estariam ligados, ainda que seja improvável, é importante como medida de segurança. A documentação do Arduino indica que o pino que suporta a maior carga (o pino Vin) consegue fornecer em torno de 1A, todavia pela interpretação da Tabela 4 concluímos que se todos os 100 leds do projeto estiverem acesos ao mesmo tempo em sua potência máxima (outra situação improvável, todavia possível), o circuito irá requisitar, apenas considerando os leds, de cerca de 2A. Portanto decidiu-se fazers uso de uma fonta externa que suporta até 2,5A e forneça 5V (características comuns em fontes de celulares); na especificação deste projeto foi adotada a fonte da marca LG, modelo STA-U12BR. Faz-se ênfase na importância de que o terra do Arduino (consequentemente a referência do sinal de informação do LED), deve estar em curto com o terra desta fonte, caso contrário o circuito apresentará comportamentos indesejados. Na Figura 4 é apresentado o circuito final montado.

Figura 4 – Estrutura eletrônica terminada.



Fonte: Produção do próprio autor.

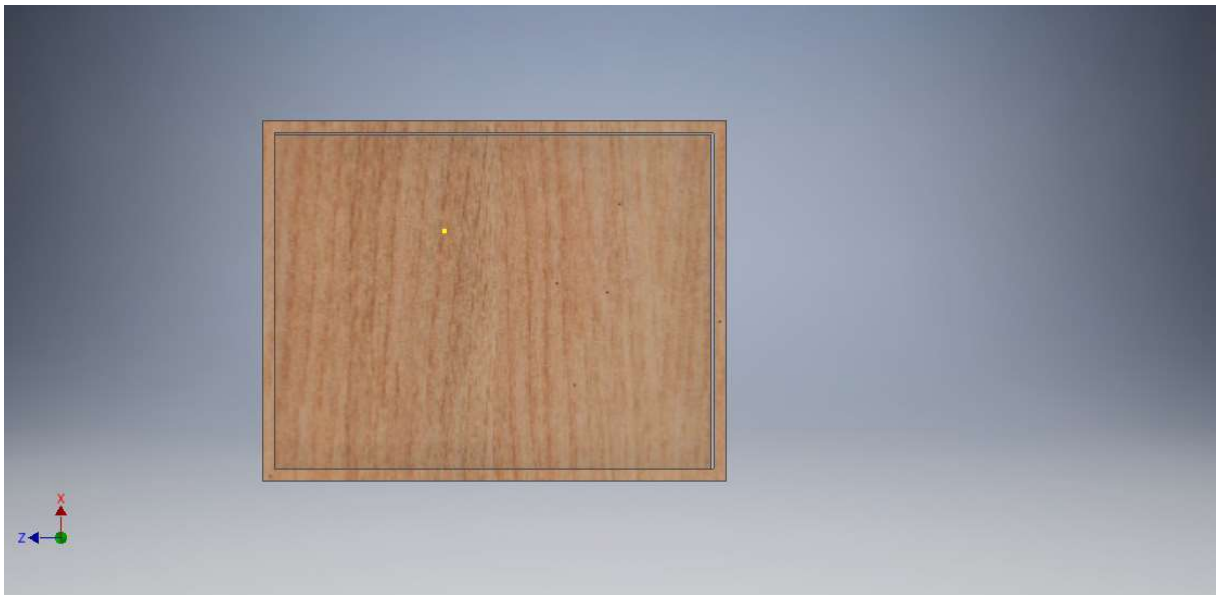
2.2 ESTRUTURA FÍSICA

O principal objetivo da estrutura do projeto foi criar uma estrutura que fosse fácil de ser reproduzida para ir de acordo com os objetivos centrais do projeto; facilmente transportada a

fim de ser um componente versátil em diferentes ambientes de sala de aula, e que em seu design desse foco no que é mais importante que é a fita de leds. Foi também importante na tomada de decisão dos materiais e confecção geral que os materiais usados fossem de fácil acesso, leves e na medida do possível baratos. O produto final deste trabalho não é um produto comercial, é um protótipo funcional que não depende de uma protoboard e cujas diferenças para um produto está apenas nas considerações de produção em massa e custo, uma vez que o mesmo não depende de uma protoboard, todavia não conta com uma placa de circuito impresso, processo este que poderia facilitar o posicionamento físico e também hegemonizar a produção em massa do projeto. O objetivo da confecção é trazer a mesma experiência que o produto derivado desta mesma ideia traria a fim de avaliar a qualidade da experiência que o projeto traz aos alunos beneficiados pela sua implementação, além de usar como plataforma para entender todos os detalhes envolvidos no desenvolvimento de um produto final a ser lançado no mercado ou implementado em nível profissional.

A confecção deste projeto foi realizada em 4 elementos principais. O primeiro e principal foi uma caixa de madeira de dimensões 295mm x 270mm x 15mm, feita de madeira MDF, muito usada para trabalhos artísticos, ela é de fácil acesso, barata, não é resistente, porém este não é um fator muito determinante visto que ela não pretende receber muitos esforços físicos. A Figura 5 abaixo apresenta o projeto da caixa feito no software *Autodesk Inventor*.

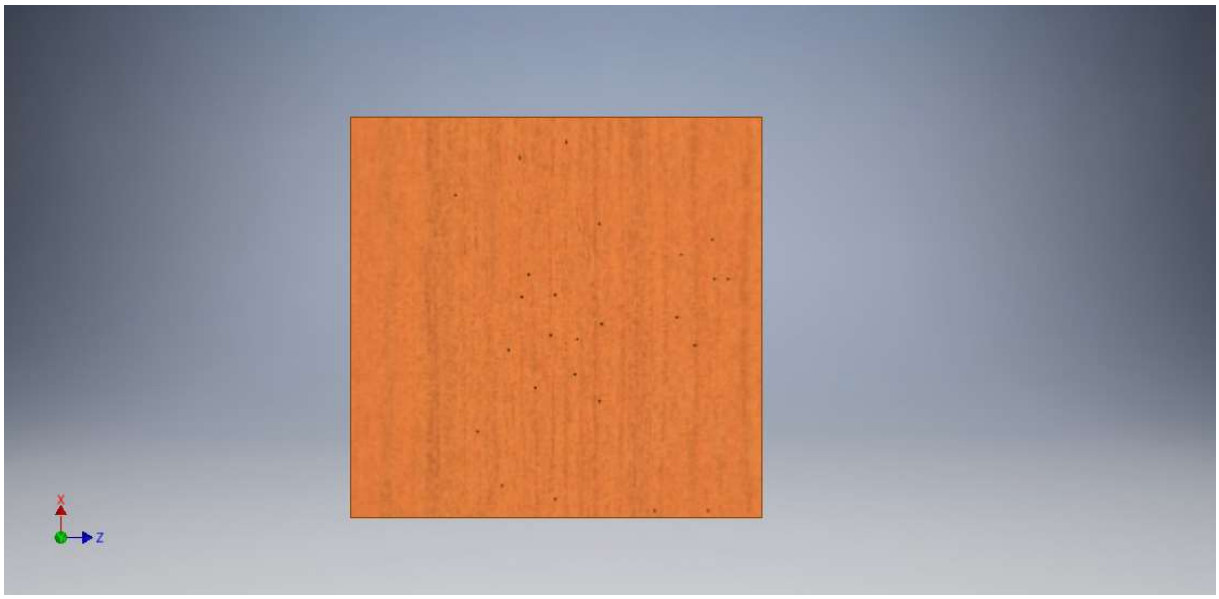
Figura 5 – Caixa do projeto projetada no software *Autodesk Inventor*.



Fonte: Produção do próprio autor.

A segunda parte da estrutura física do projeto consiste no apoio das fitas de led porque no intuito de colocar como foco do projeto a luz proveniente dos leds, colocá-los na base da caixa atrapalharia a visualização simples e impediria a fluidez desejada na execução da aula. A base das fitas é feita também de madeira MDF nas seguintes dimensões 150 mm x 150 mm x 3mm. A sua pequena espessura é devido ás fitas serem uma estrutura bastante leve e que não necessita de uma base mais resistente. Seus dimensões não são exatamente as medidas internas da caixa apresentada anteriormente, pois o circuito necessita de um vão entre a base e a caixa para a passagem de fios de alimentação, visto que cada linha da fita tem três conexões, uma de alimentação positiva, outra de terra e outra de sinal. A Figura 6 abaixo ilustra o projeto da base das fitas feito no software *Autodesk Inventor* na versão de estudante.

Figura 6 – Base de leds projetada no software *Autodesk Inventor*.

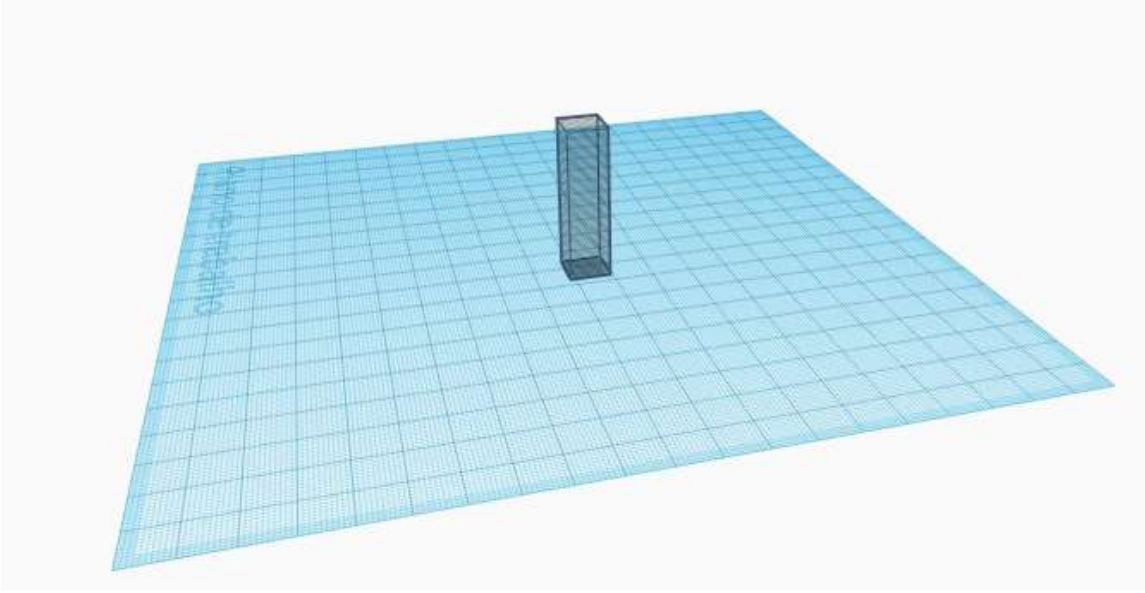


Fonte: Produção do próprio autor.

O material MDF é interessante para aplicações com espessuras pequenas até cerca de 10mm, a fim de elevar a base de leds mostrada anteriormente fez-se necessárioa criação de alguns apoios da base feitos em material destinado principalmente para desenvolvimento em impressora 3D, no caso o plaśtico PLA. Esta peça específica poderia ser feita de outros materiais, mas sua escolha se deve a sua facilidade de produção em uma impressora 3D além da sua versatilidade, é possível determinar a densidade, formato do preenchimento da peça, entre outros detalhes. Conforme a necessidade é possível modificar a altura ou mudar a quantidade de apoios. Neste projeto adotou-se quatro apoios nas dimensões 7mm x 7mm x 15mm. A Figura 7 mostra o projeto feito na plataforma Tinkercad, uma plataforma gratuita,

online de desenvolvimento 3D, usada aqui no intuito também de facilitar o compartilhamento, todavia é importante salientar que esta plataforma funciona bem para projetos mais simples e não detém de recursos mais complexos como o *Autodesk Inventor*.

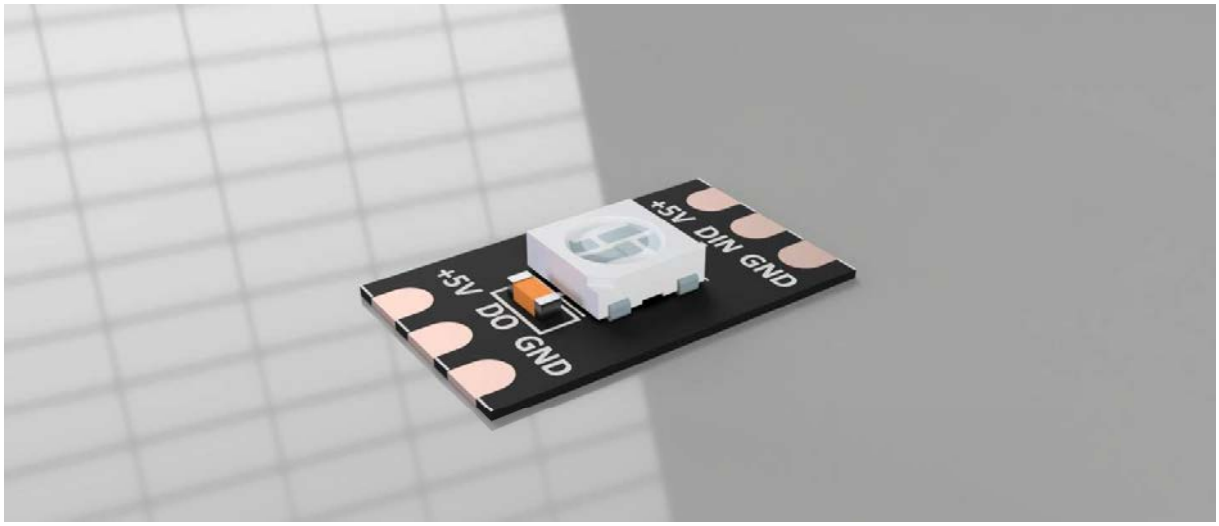
Figura 7 – Apoio projetado no site Tinkercad



Fonte: Produção do próprio autor.

A última principal estrutura física são as fitas de led. Por mais que a subseção anterior tenha abordado as características eletrônicas do led em si, existem diferentes fornecedores de fitas usando o CI WS2812. Estas fitas são muito utilizadas para iluminação interna doméstica e são por vezes vendidos associados a um controlador a ser conectado em um controle remoto, sistema de som, entre outros. As fitas de led foram adquiridas como uma fita única com indicações de onde cortar em sub fitas, de maneira que todos os leds podem ser utilizados individualmente, o modelo comprado vem com a sua base preenchida por uma fita de cola forte da marca 3M. A figura abaixo mostra o modelo individual de um dos leds, o modelo foi adquirido na base de designs compartilhados *GrabCad*.

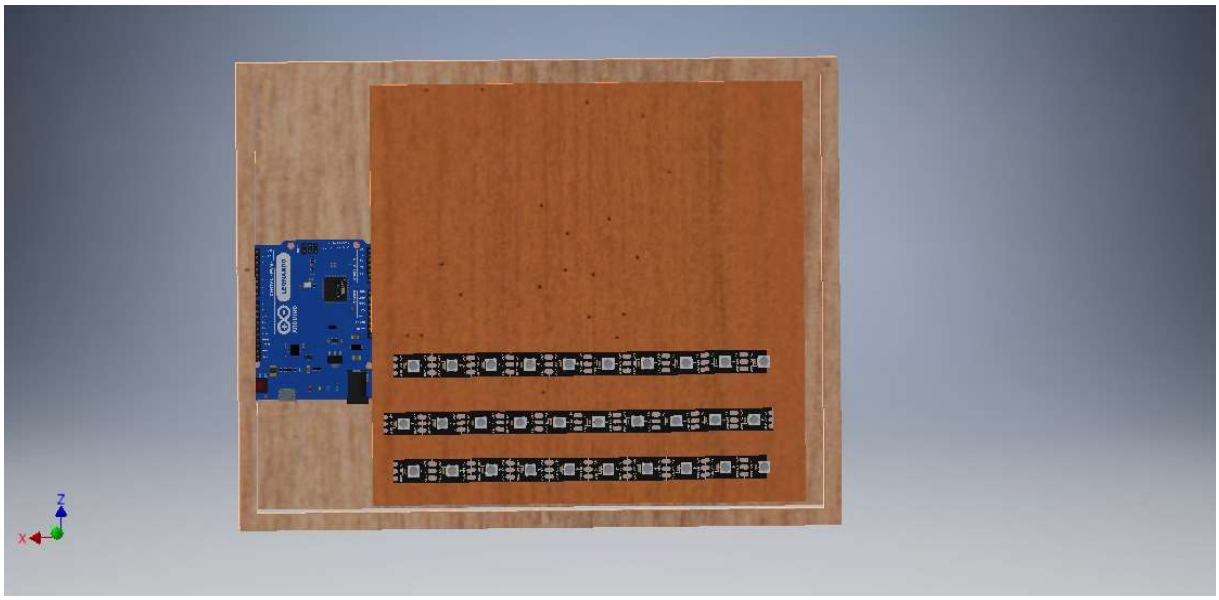
Figura 8 – Design 3D de um led individual da fita de led.



Fonte: Produção do próprio autor.

A Figura 9 mostra a estrutura final desenvolvida no software *Autodesk Inventor*.

Figura 9 – Projeto final realizado no software *Autodesk Inventor.*



Fonte: Produção do próprio autor.

As bases feitas em impressora 3D foram coladas entre a caixa e a base das fitas de led usando uma fita de cola forte da marca 3M, e as fitas de led foram coladas em sua base usando a cola proveniente na própria fita, a Tabela 5 apresenta os custos do projeto.

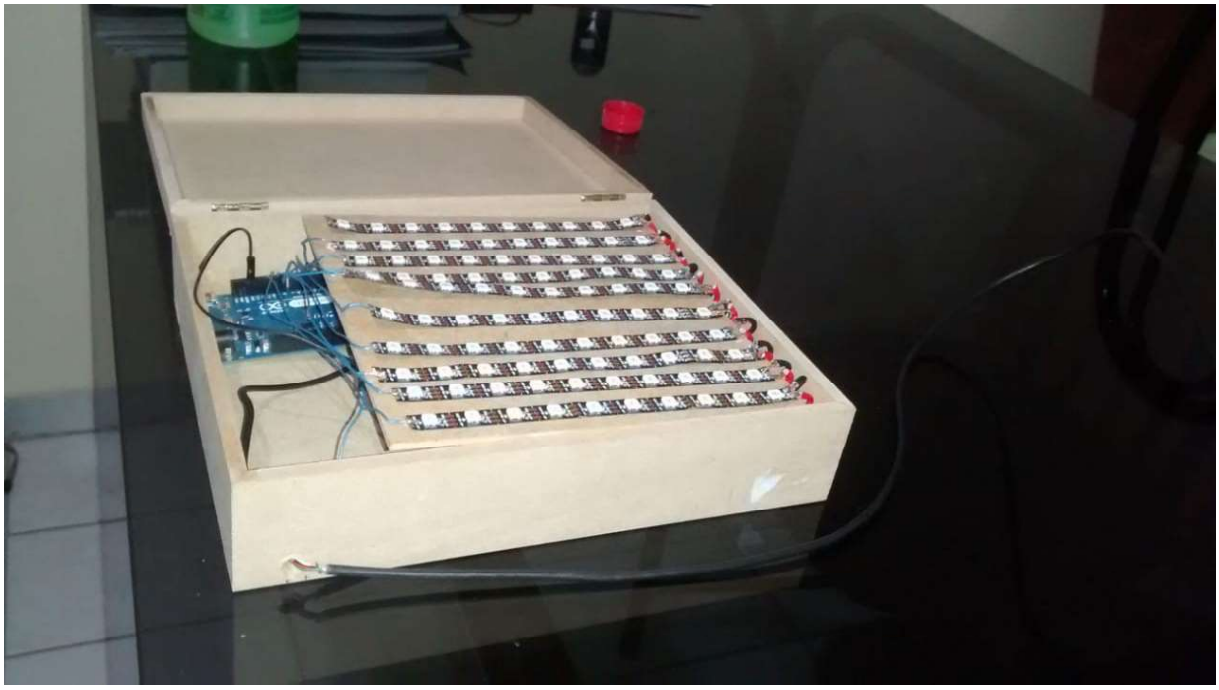
Tabela 5 – Custos do projeto

Peça	Preço
Caixa	R$ 13,00
Base das fitas de led	R$ 5,00
Fitas de Led	R$ 60,00
Apoios em PLA	R$ 3,00
Arduino Leonardo	R$ 40,00
Fios e conectores	R$ 10,00
Fonte	R$ 30,00
Total	R$ 131,00

Fonte: Produção do próprio autor

A Figura 10 mostra a confecção física do projeto.

Figura 10 – Estrutura física pronta do projeto.



Fonte: Produção do próprio autor

2.3 ESTRUTURA DO SOFTWARE

O desenvolvimento deste projeto visa criar um produto genérico que deve servir para um número não definido de aplicação, a ferramenta computacional que podemos usar para satisfazer este objetivo é a biblioteca. A biblioteca consiste em uma série de funções que serão usadas na aplicação do material. Contribuindo ainda com o objetivo de gerar um produto que a comunidade de tecnologia consiga usar como base para novos projetos em todos os

aspectos; a biblioteca criada neste projeto segue ao máximo as diretrizes para escrita de bibliotecas indicada pela Arduino.

O uso deste dispositivo como plataforma de ensino de lógica precisa ter comandos que sejam ao mesmo tempo genéricos do ponto de vista que podem ser usados de maneira combinadas com estruturas como *while, for, if, else,* entre outras para criar diferentes soluções lógicas para desafios propostos pelo método. A seguir será explicado a estrutura geral do código da biblioteca, incluindo como ela segue as diretrizes indicadas, e uma explicação detalhada do funcionamento de cada uma das funções implementadas.

A linguagem aceita pelo compilador padrão do *hardware* embarcado Arduino é C++, esta linguagem por sua vez é classificada como uma linguagem orientada a objetos, portanto a estrutura desta biblioteca é desenvolvida como uma classe. A biblioteca é então dividida em dois arquivos, um arquivo *header* usado para fazer a declaração dos atributos e métodos da classe, e um arquivo *.cpp* que irá comportar as implementações dos métodos declaradores no outro arquivo. A seguir iniciaremos a documentação deste código pelo arquivo silverLogic.h, sendo este o arquivo *header* que leva o nome comercial adotado para este projeto. Começamos esta descrição pela declaração das bibliotecas utilizadas na Figura 11.

Figura 11 – Declaração das bibliotecas.

```
#include "Arduino.h"
#include <Adafruit_NeoPixel.h> //Biblioteca para manipulação das fitas de led
```

Fonte: Produção do próprio autor

A diretiva de programação #include faz o papel de anexar ao programa o conteúdo de um arquivo externo. A principal a ser adicionada que é necessária para utilizar os comandos básicos do Arduino é a biblioteca “Arduino.h”. Logo em seguida é adicionada a biblioteca “Adafruit Neopixel”, esta biblioteca foi desenvolvida pela distribuidora de *hardware* eletrônico Adafruit e foi concebida para lidar com os principais produtos de CI’s para controle individual de led, ela é essencial para facilitar a manipulação das fitas, embora esta biblioteca já seja uma importante ferramenta para a utilização deste projeto, ela precisa ser implementada dentro da biblioteca para criar algo mais compatível e direcionado para o produto. A Figura 12 mostra a declaração dos métodos públicos da classe, sendo eles as ferramentas a serem utilizadas.

Figura 12 – Assinatura dos métodos públicos.

```
class SilverLogic
{
//Declaração dos métodos da classe SilverLogic
public:

//Declaração do construtor sem parâmetros
SilverLogic();
//Declaração do construtor com parâmetro de luminosidade
SilverLogic(int luminosidade);
//Declaração da função que inicializa o experimento
void inicializar(int linha, int coluna);
//Método para configurar uma das posições da fita
void setPosicao(int linha, int coluna, String cor);
//Método que retorna a matriz que representa a fita de led
int** getMatriz();
//Método que retorna a distância do Led azul até uma determinada cor
int distancia(String cor,String sentido);
//Método que faz o led azul mover-se em alguma direção
void mover(String sentido);
```

Fonte: Produção do próprio autor

Os dois primeiros métodos declarados são os construtores, são métodos responsáveis por alocar a memória necessaria para a criação do objeto, neste caso fez-se uso da característica comum em linguagens orientadas a objeto, chamada de sobrecarga, em que é possível declarar dois métodos com o mesmo nome porém com característica de parâmetros de diferente, a fim de executar tarefas semelhantes sob diferentes conjuntos de valores; neste caso temos um construtor sem parâmetros que atribui um valor padrão de luminosidade da matriz de leds, enquanto que no outro é possível determinar a a intensidade luminosa por meio do parâmetro do tipo inteiro *luminosidade*.

A próxima função chamada de inicializar, executa uma sequência de comandos básicos para o uso do projeto, como definição do sentido das portas, além de definir o ponto de início da luz azul que será como um personagem nesta aplicação, todas as movimentações, verificações e tarefas estarão relacionadas a posição da luz azul. A visualização desta posição pode variar de atividade para atividade, em outras etapas deste trabalho será apresentado uma ferramenta que pode tornar este, e outros processos de inicialização, menos abstratos, tornando o planejamento da aula melhor.

O método setPosicao() representa uma das funcionalidades internas do desenvolvimento mais básicas, que é determinar a cor em uma determinada posição da matriz,

esta função também pode ser beneficiada de uma ferramenta que usando este comando diversas vezes construa a estrutura inicial ele recebe como parâmetros a posição dentro da matriz, e a cor desejada em formato de String. As opções de cores que foram programadas foram limitadas, todavia podem facilmente ser modificadas alcançando uma grande variedade de cores graças a qualidade do led. Esta é uma cor que tem grande utilidade ao implementar outros métodos, todavia tem o potencial de ser utilizada dentro de aplicação na metodologia.

O guia de desenvolvimento de API disponibilizado pela Arduino não recomenda que as bibliotecas desenvolvidas tenham a necessidade de conhecimento sobre ponteiros, visto que Arduino é visto como uma plataforma inclusive e este ser um tema de muita dificuldade entre pessoas que começam a aprender a linguagem C++, mas ao mesmo tempo ele apresenta situações em que as biblitoecas oficiais da plataforma também tiveram que usar esse recurso, seja de uma maneira que futuramente pode ser revisada, ou porque não encontraram opção melhor. No caso do método **getMatriz() foi necessário adotar o uso de ponteiros, pois foi a única maneira de haver um método que retornasse uma matriz que representasse os estados de cada um dos leds da matriz de led, ele é como uma informação computacional do que está acontecendo no objeto físico. Esta função foi criada para dar um subsídio de uso da biblioteca com algum recurso gráfico usando, por exmeplo, a linguagem Processing que tem a capacidade de integrar recursos gráficos e embarcados. Usando este método ou não, esta matriz é um recurso constante dentro da lógica da biblioteca. A definição dos estados usando-a se fez com a associação de cada cor com um número, esta estrutura computacional servirá de base para outros métodos mais aplicáveis.

O método distancia() recebe como parâmetro um sentido e uma cor. Ele retorna um número inteiro que indica quantos espaços de distância há entre o led azul e o primeiro led da cor correspondente. Ele retorna -1 caso não encontre nenhuma cor naquele sentido.

A principal ação a se executada dentro de uma dinâmica de aula é a movimentação, dado esta informação nota-se que a função mover() é bastante importante. Como o led não tem nenhum tipo de indicação física que poderia de maneira clara mostrar o sentido em que aquele movimento ocorre, é preciso enviar a informação do sentido em que o movimento está acontecendo em formato de String.

Uma vez que fisicamente o que mais define este objeto seria a matriz de leds, os atributos adotados para a definição de um objeto da classe SilverLogic são apresentados abaixo.

Figura 13 – Exemplo de declaração dos atributos privados da classe.

```
private:
//Declaração dos pinos das fitas de led
int pinoLinha1 = 2;
//Declara a matriz que representa o estado de cada led;
int** matriz;
```

Fonte: Produção do próprio autor

Como boa prática de programação orientada a objetos, os atributos da classe são declarados como privados, usando a palavra-chave *private*. Os primeiros atributos declarados representam os pinos em que cada fita de led, que representa uma linha da matriz cada, está conectada. A princípio havia sido optado por tornar este atributo um valor que poderia ser alterado dentro do construtor, todavia a maneira como o programa foi estruturado obrigou que a variável de cada pino fosse definida logo em sua declaração. A Figura 14 apresenta a declaração de mais um atributo.

Figura 14 – Declaração do atributo de luminosidade.

```
//Declaração da variável que controla a luminosidade da aplicação
int luminosidade;
```

Fonte: Produção do próprio autor

O próximo atributo declarado foi a luminosidade. Como citado em outros pontos do trabalho, os leds tem uma forte intensidade luminosidade por isso este é um fator que pode ser alterado para definir a luminosidade durante todo o uso do projeto, este é um valor que varia de 0 a 100.

Com o intuito de lidar de maneira mais prática e não ter que repetir muitas vezes trechos de código optou-se por usar um vetor de objetos que controlam a fita de led para lidar com a matriz como um todo. Primeiramente é importante compreender a estrutura de um único objeto para controlar a fita.

Figura 15 – Chamada do construtor de um objeto da fita.

```
Adafruit_NeoPixel(10,pinoLinha2, NEO_GRB + NEO_KHZ800)
```

Fonte: Produção do próprio autor

A classe adotada para tal fim foi a classe *Adafruit_Neopixel*, ela foi desenvolvida pela distribuidora de produtos eletrônicos Adafruit a fim de criar uma solução mais simples para

lidar com os diversos modelos de leds que eles vendiam. O primeiro argumento passado para o construtor do objeto da classe é a quantidade de leds que existem na fita, uma vez que as fitas podem ser cortadas para ter o tamanho que o usuário desejar. O segundo argumento representa qual o pino que irá controlar aquela fita, por fim o último argumento e uma constante que varia de modelo para modelo, no nosso caso foi consultado o fabricante para saber quais constantes utilizar. Uma vez que estamos lidando com uma matriz de módulo 10, foi criado um vetor de 10 posições, onde cada posição contém um objeto que controla de maneira individual 10 leds.

Por fim é declarada uma matriz em formato de ponteiros para que ele possa ser passado como parâmetros em eventuais métodos e também retornado por métoimento do programa dos. Ela é a representação do estado de cada um dos leds da matriz, isso será útil para um auto reconhecimento do programa.

A Figura 16 passa para o entendimento do arquivo .cpp que contém as implementações dos métodos acima discutidos. Começaremos discutindo os construtores da classe.

Figura 16 – Construtores da classe.

```
SilverLogic :: SilverLogic()
{
luminosidade = 20;
}

SilverLogic :: SilverLogic(int luminosidade)
{
if(luminosidade >= 0 && <= 100) this->luminosidade = luminosidade;
else luminosidade = 20;
}
```

Fonte: Produção do próprio autor

O único atributo determinado dentro do construtor é a luminosidade. No construtor sem parâmetros a luminosidade é definida por padrão como 20, já no construtor com parâmetros a luminosidade é um valor definido pelo parâmetro da função. A fim de impedir *overflow* do parâmetro existe um condicional que força o valor padrão caso o valor inserido como parâmetro esteja fora do intervalo de 0 a 100.

Primeiramente ocorre a inicialização da comunicação serial do Arduino, este protocolo de comunicação é muito utilizado para fazer a eventual depuração do código. A seguir o uso de um vetor é usado como vantagem para executar os comandos necessários para inicializar cada uma das fitas de led. Ao invés de ter que repetidamente chamar cada um dos objetos que

comandam a fita, faz-se de um *for* para percorrer o vetor e executar os comandos em cada uma das fitas. O primeiro comando begin() é uma função de inicialização das fitas, prepara os pinos para enviar a informação e aloca a memória necessária para controlar cada led. O segundo comando show() é sempre usado para atualizar a fita de leds, ou seja , após fazer alguma alteração na fita quanto a qual led iluminar, sua intensidade luminosa ou cor, é preciso chamar a função show() para que a mudança na fita apareça fisicamente ao usuário.

Figura 17 – Inicialização da matriz.

```
//Inicializa a matriz quadrada de ordem 10 que representa os leds
matriz = new int*[10];

for(int i = 0; i < 10; i++)
{
matriz[i] = new int[10];

for(int j = 0;j < 10; j++)
{
matriz[i][j] = 0;
}
}
```

Fonte: Produção do próprio autor

A Figura 17 motra como a matriz é construída em dois processos. Primeiramente a variável é inicializada como um vetor de 10 posições, depois ele passa por um laço *for* em que cada elemento do vetor anteriormente criado recebe também um vetor de 10 posições. Ou seja, a matriz é composta por 1 vetor de 1posições em que cada posição faz referência a outro vetor de 10 posições. Após a construção desta matriz todos seus valores são inicializados com 0, indicando que a matriz começa com todos os leds desligados.

Figura 18 – Inicialização do LED azul.

```
//Inicializa com o led azul na posição inicial
this->setPosicao(linha,coluna,"Azul");
matriz[linha][coluna] = 1;
}
```

Fonte: Produção do próprio autor

A última etapa da inicialização mostrada na Figura 18 é responsável por iniciar o LED azul na posição que os parâmetros “linha” e “coluna” definiram, estes parâmetros serão

importantes no planejamento de aulas para a posição inicial do led azul controlável. O método usado chamado setPosicao() é definido a seguir nesta biblioteca. Sempre associado a iluminação de um led, deve haver também uma atualização na matriz criada anteriormente, por isso na posição em que o led azul foi inicializado o valor passa de 0 para 1.

Figura 19 – Método para obter a matriz da classe.

```
//Retorna a matriz que representa o estado de cada led.
int** SilverLogic :: getMatriz()
{return matriz;
}
```

Fonte: Produção do próprio autor

A função apresentada na Figura 19 é apenas uma função para retornar a matriz caso ela precise ser acessada externamente para visualização ou retrabalho dos valores, até mesmo adoção de métodos de inteligência artificial ou outro algoritmo focados na matriz como, por exemplo, autômatos celulares.

O método da Figura 20 recebe como argumentos uma posição da matriz, definida por linha e coluna e também uma cor como String.

Figura 20 – Método para alterar uma posição.

```
//Definição da função que muda a cor de um led numa posição específica.
void SilverLogic :: setPosicao(int linha,int coluna, String cor)
{
if(cor == "Azul")
{
fitaLinha[linha].setPixelColor(coluna,30,144,255);
fitaLinha[linha].show();
matriz[linha][coluna] = 1;
}
```

Fonte: Produção do próprio autor

Primeiramente existe uma sequência de verificações usando *if* para determinar qual cor foi selecionada, mantendo também uma condição para o caso de que nenhuma cor válida foi selecionada. Este é um ponto de fácil expansão, visto que é fácil encontrar o código para diferentes cores desejadas e assim expandir o código para suportar diferentes cores, dentro do objetivo deste código, a ideia é que cada cor tenha um diferente significado de acordo com cada aula, dando também espaço também para significados universais ao longo de diferentes desafios. Tendo uma das cores selecionadas por um dos condicionais, o vetor de fita de led na

posição da linha desejada é definida na cor desejada, na posição indicada pela coluna usando o método *setPixelColor* acompanhado do método *show.* Por fim a matriz da classe é atualizada para a cor correspondente de acordo com a tabela abaixo.

Tabela 6 – Códigos de cada cor configurada

Cor	Código
Desligado	0
Azul	1
Vermelho	2
Verde	3
Laranja	4
Roxo	5

Fonte: Produção do próprio autor

Os métodos que foram discutidos agora são a base da própria classe, foram métodos que são utilizados bastante dentro de outros métodos que serão mais utilizados pelos usuários finais do produtos, os alunos e professores. Para realizar construções lógicas desafiadores é preciso ter informações dentro do desafio, neste caso do projeto a informação mais importante que teremos é a distância do led azul até uma determinada cor e num determinado sentido.

Figura 21 – Método para retornar uma distância.

```
int SilverLogic :: distancia(String cor,String sentido)
{
//Verifica o sentido do movimento
if(sentido == "Cima")
{
//Percorre o sentido em que se encontra o led azul, para cima
for(int i = posicaoAzulLinha; i >= 0; i--)
{
if(matriz[i][posicaoAzulColuna] == numeroCor) return (posicaoAzulLinha - i);
}
return -1;
}
```

Fonte: Produção do próprio autor

O método acima recebe como parâmetros a cor a qual se deseja saber a distância e depois o sentido que se deseja verificar a distância. A primeira tarefa deste processo é fazer a equivalência entre a string que foi recebida e o código equivalente na matriz como indicado na Tabela 6, uma vez que a verificação da distância se derá na matriz equivalente ao que está

ocorrendo com os leds. Feito isso é preciso encontrar a referência á distância desejada, logo, o led azul; a posição em que ele se encontra tanto em posição de linha quanto coluna são armazenados em variáveis. Após isso uma série de condições *if* são utilizados para cada um dos sentidos a fim de percorrer naquele sentido usando o laço *for* tendo sempre a preocupação em limitar a busca para até encontrar a cor desejada ou o fim da matriz no sentido explorado, em caso de a cor jamais ser encontrada o valor que é retornado é de -1, valor este impossível para uma situação comum de busca por isso este valor vai servir como uma maneira de saber se existe em algum lugar daquele sentido uma distância válida. Caso a cor não seja uma cor válida o valor -1 é usado novamente como maneira de indicar o erro.

O método que acabou de ser explicado serve para receber informações importantes para tomada de decisões em projetos de ensino de lógica, já o método a seguir serve para reagir a essa informação dada por meio de um movimento controlado.

Figura 22 – Método para mover o LED azul.

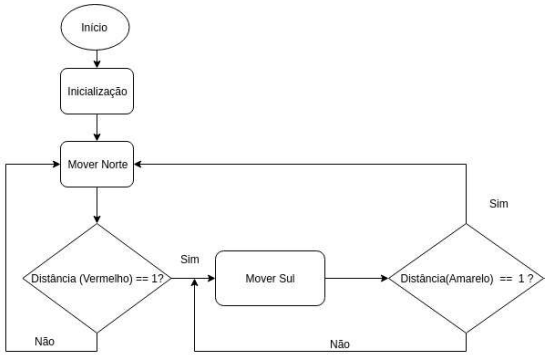
```
void SilverLogic :: mover(String sentido)
{
//Verifica o sentido do movimento
if(sentido == "Cima")
{
if( (posicaoAzulLinha - 1) >= 0 ) //Verifica se alcançou o fim da matriz
{
Serial.println("Cima");
Serial.println(posicaoAzulLinha);
Serial.println(posicaoAzulColuna);
this->setPosicao(posicaoAzulLinha,posicaoAzulColuna,""); //Desliga o led na posição
atual
matriz[posicaoAzulLinha][posicaoAzulColuna] = 0; //Zera a posicao atual na matriz
this->setPosicao(posicaoAzulLinha - 1,posicaoAzulColuna,"Azul");
matriz[posicaoAzulLinha - 1][posicaoAzulColuna] = 1;
delay(500);
```

Fonte: Produção do próprio autor

O processo do movimento se assemelha ao processo de identificar a distância, tendo como primeira tarefa identificar a posição em que o led azul está, feito isso uma série de condicionais *for* lida com cada tipo de movimento. Cada movimento então é limitado para que o led azul pare ao alcançar uma das paredes da matriz de leds. É este método em a função *setPosicao*() é utilizada, a posição é alterada a cada a cada ciclo em que o led ainda irá se movimentar. Ao final de cada ciclo de movimentação existe um delay de 500ms para que seja visualmente perceptível a quem usa.

A seguir é apresentado um exemplo de um programa utilizando a biblioteca, nele o aluno é desafiado a programar seguindo o fluxograma abaixo:

Figura 23 – Fluxograma de programa exemplo para utilizar classe SilverLogic.



Fonte: Produção do próprio autor.

A ideia do fluxograma acima é que o azul comece em uma posição central entre uma posição ao norte em vermelho e uma posição ao sul amarela. A partir desta configuração inicial, a posição azul se move ao norte até o vermelho, depois se move na posição contrária até encontrar a cor amarela quando então se move novamente para o norte, repetindo a lógica. Existem diferentes maneiras de resolver este probleme, usando diferentes estruturas. Em uma turma que já aprendeu o uso da estrutura de laço *while* o código da Figura 24 é apropriado.

Figura 24 – Exemplo de código usando a biblioteca.

```
#include <SilverLogic.h>
SilverLogic tabuleiro = SilverLogic(20);
void setup()
{
 tabuleiro.inicializar(4,0);
 tabuleiro.setPosicao(7,0,"Vermelho");
 tabuleiro.setPosicao(2,0,"Amarelo");
}
void loop()
{
 while(tabuleiro.distancia("Vermelho","Cima") > 1)
 {
 tabuleiro.mover("Cima"); //Move a luz azul para cima
 }
 while(tabuleiro.distancia("Amarelo","Baixo") > 1)
 {
 tabuleiro.mover("Baixo"); //Move a luz azul para baixo
 }
```

Fonte: Produção do próprio autor

Todavia para o caso da turma ainda não trabalhar com laços e lidar apenas com um condicional mais simples como a estrutura *if-else* é possível que seja obtido o seguinte código que não usa nenhuma estrutura lógica de laço, mostrando como um mesmo experimento, uma mesma configuração inicial pode ser aplicada em diferentes cenários.

Figura 25 – Exemplo de código usando a biblioteca.

```
//Inclui a biblioteca SilverLogic no programa
#include <SilverLogic.h>

//Cria o objeto do tipo SilverLogic com luminosidade 20
SilverLogic tabuleiro = SilverLogic(20);

//Função que é executada apenas uma vez ao ligar o Arduino
void setup()
{
 //Inicializa com a cor azul na posição 4,0
 tabuleiro.inicializar(4,0);
 //Ilumina em vermelho na posição 7,0 e amarelo na posição 2,0
 tabuleiro.setPosicao(7,0,"Vermelho");
 tabuleiro.setPosicao(2,0,"Amarelo");

 int movimento = 0;

}

//Função executada repetidas vezes
void loop()
{
 //Laço para mover para cima enquanto a distancia do led vermelho for
maior que 1
 if(tabuleiro.distancia("Vermelho","Cima") > 1)
 {
 movimento = 0;
 }

 //Laço para que mover para baixo até achar o amarelo
 else if(tabuleiro.distancia("Amarelo","Baixo") > 1)
 {
 movimento = 1;
 }

 if(movimento == 0) tabuleiro.mover("Cima");
 else if (movimento == 1) tabuleiro.mover("Baixo");
```

Fonte: Produção do próprio autor

3 DESENVOLVIMENTO DA METODOLOGIA

3.1 OBJETIVOS GERAIS

O principal objetivo da metodologia é extrair ao máximo o potencial do dispositivo, usando uma estratégia que dê liberdade a quem usar a fim de tornar este produto o mais generalista possível, ao dar muitas ferramentas para o instrutor ao invés de um material fechado. Outro importante objetivo desta metodologia é focar na solução de problemas e não em maneiras específicas de resolver os problemas, a fim de criar um sistema que inclua a individualidade de pensamento de cada aluno. O objetivo deste material é capacitar o aluno para encontrar soluções gerais e não focadas na sintaxe do Arduino, ela é apenas um instrumento, todavia necessário, para a resolução dos desafios.

3.2 SEÇÕES DA METODOLOGIA

A metodologia é divida em seções a fim de auxiliar o professor na criação do material mantendo uma padronização, o que ajuda tanto na organização e planejamento de aula quanto na avaliação da qualidade e impacto das aulas.

3.2.1 CONTEXTUALIZAÇÃO

O ponto de entrada da metodologia é a contextualização, o objetivo aqui é concentrar o aluno na proposta daquela aula e mostrar a ele como o conteúdo se relaciona com os conhecimento que ele já tem. Lógica é uma coisa que o ser humano usa o tempo todo, é importante nesta seção trazer a tona os conhecimentos do aluno, não só para que ele fique concentrado e focado na aula, mas para que ele comece o processo de aprendizado dele com confiança de que ele tem as ferramentas necessárias para compreender o conteúdo que lhe será apresentado.

3.2.2 SINTAXE

Ainda dentro do processo de dar ferramentas ao aluno, é importante, sempre que preciso, a introdução á sintaxe necessária para a realização do experimento a seguir. Nas primeiras aulas mais introdutórias é natural que apareça muito conteúdo nesta seção a respeito de sintaxes

básicas da IDE do Arduino, como a função *void loop(),* entre outros. Em outras aulas mais intermediárias imagina-se que as funções disponíveis na biblioteca SilverLogic serão apresentadas aos poucos. Em paralelo com as sintaxes necessárias para qualquer solução de problema, são introduzidos também estruturas mais gerais da linguagem de programação, que são os assuntos de maior interesse para o curso proposto. Em aulas mais avançadas pode-se começar a usar essa seção para fazer uma introdução a estruturas mais complexas como algoritmos de seleção, vetores,entre outros.

3.2.3 FLUXOGRAMA

Este é um tema de grande debate a respeito de sua importância, há cursos de programação que acabam deixando de lado o ensino específico de fluxograma, todavia isso tem sido mais frequente em cursos que atingem linguagens específicas em que pode-se focar em habilidades e estratégias mais específicas. No nosso caso em que o foco é uma lógica mais generalista, o fluxograma assume um papel mais importante. O propósito desta seção é fornecer uma explicação do desafio usando uma linguagem padrão. Dois fluxogramas pelo menos devem ser apresentados, um como exemplo junto ás linhas de código correspondentes, como se fosse uma espécie de tradução. E outro como base para resolver o desafio.

3.2.4 CONFIGURAÇÃO INICIAL

Todo desafio precisa de uma configuração inicial. A maneira mais prática de mostrar aos alunos ela é mostrando diretamente no dispositivo. Sua programação corre com uma combinação de chamadas ao comando *setPosicao*. Em um projeto futuro é possível criar uma aplicação web para gerar a partir de uma interface gráfica a configuração inicial e devolver ao usuário, geralmente um professor, as linhas de código necessárias para criar a configuração inicial do desafio.

3.3 EXEMPLO PRÁTICO

3.3.1 CONTEXTUALIZAÇÃO

Muitas vezes não sabemos quantas vezes devemos repetir uma determinada ação, sabemos apenas que devemos repetir algo até que um determinado resultado seja alcançado. É como o

processo de fazer um doce, como brigadeiro por exemplo, seria difícil dizer o quanto mexer a a mistura, então determina-se mexer até que seja possível ver o fundo da panela, portanto ao invés de determinar um número específico de giros ou movimentos na panela, determinamos uma condição que devemos cumprir e repetimos a ação até completá-la.

3.3.2 SINTAXE

A estrutura que iremos usar hoje é a primeira estrutura de laço de repetição que iremos ver. O *while* é uma estrutura que é usada de modo semelhante ao *if*, junto a sua declaração deve ser dada uma condição a ser verificada, ao fim de cada execução do bloco *while* a condição é verificada e se verdadeira o bloco é executado novamente. Vide exemplo abaixo:

Figura 26 – Demonstração de xintaxe

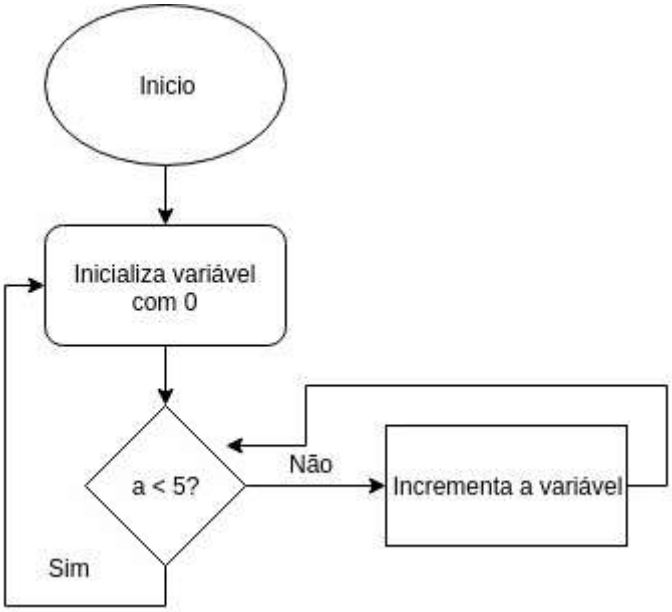
```
void setup() {

int a = 0;
}

void loop() {

 while(a < 5)
 {
 a = a + 1;
 Serial.println("Dentro do While");
 delay(500);
 }

 a = 0;

 Serial.println("Fim do While !");
 delay(500);
}
```

Fonte: Produção do próprio autor

3.3.3 FLUXOGRAMA

A estrutura que usamos para representar estruturas condicionais como *if* é a mesma que será usada para laços, o que vai mudar é apenas o fluxo do programa. A Figura 12 representa o código que foi mostrado na seção acima.

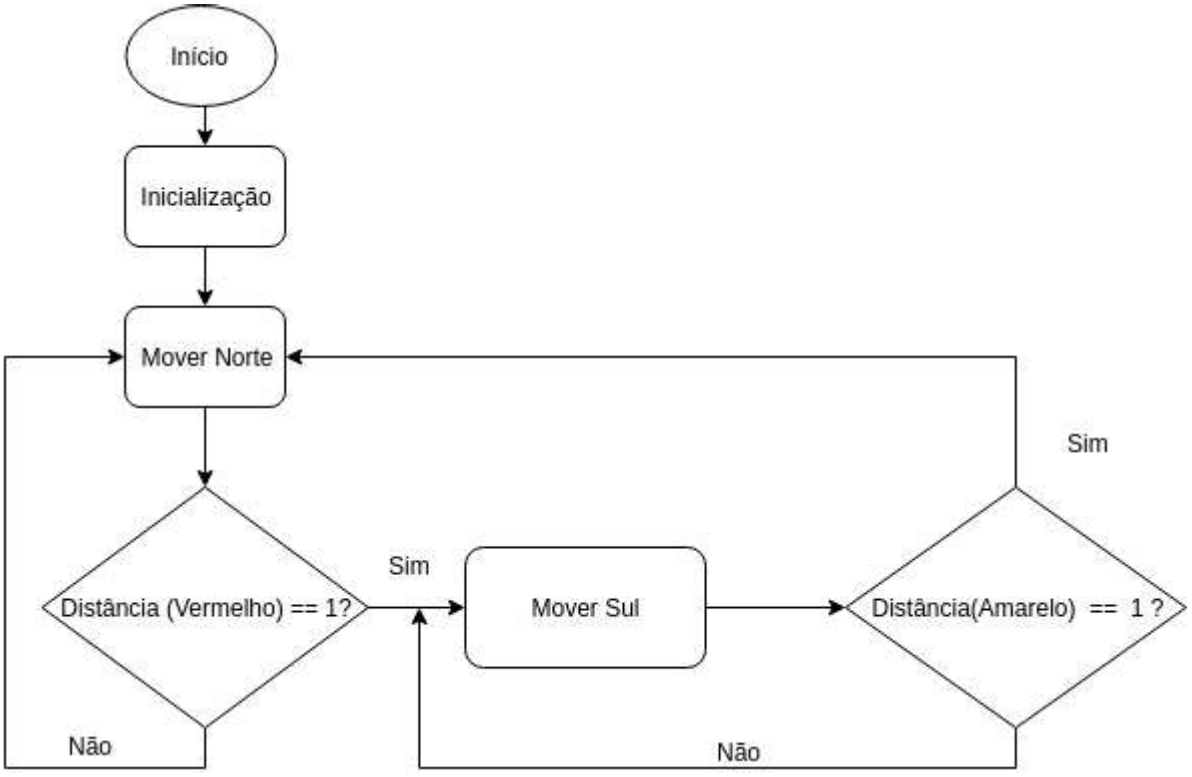
Figura 27 – Fluxograma de programa exemplo do laço while.



Fonte: Produção do próprio autor.

Compreendido este fluxograma, a Figura 13 apresenta o fluxograma do desafio desta aula, ele deverá ser implementado usando a estrutura *while* que foi apresentada nesta aula.

Figura 28 – Fluxograma do desafio da aula exemplo.



Fonte: Produção do próprio autor.

3.3.4 CONFIGURAÇÃO INICIAL

A luz azul irá começar numa posição média entre uma luz vermelha e uma luz amarela. O código abaixo garante esta configuração inicial. Para a solução é necessário usar a estrutura *while* visto que durante toda a execução o programa sempre volta para o mesmo ponto do código, indicando uma estrutura repetitiva. Não há nenhuma indicação de contadores ou algo que indique quantas vezes executar determinada ação, portanto a Figura 29 mostra a inicialização.

Figura 29 – Fluxograma do desafio da aula exemplo.

```
//Função que é executada apenas uma vez ao ligar o Arduino
void setup()
{
 //Inicializa com a cor azul na posição 4,0
 tabuleiro.inicializar(4,0);
 //Ilumina em vermelho na posição 7,0 e amarelo na posição 2,0
 tabuleiro.setPosicao(7,0,"Vermelho");
 tabuleiro.setPosicao(2,0,"Amarelo");
}
```

Fonte: Produção do próprio autor

3.3.5 SOLUÇÃO PROPOSTA

O código abaixo apresenta uma solução usando o laço while. O aluno deve ser incentivado a encontrar sua própria solução, portanto a solução abaixo deve servir apenas como um guia para o professor e não um manual ao aluno.

Figura 30 – Resolução proposta.

```
//Inclui a biblioteca SilverLogic no programa
#include <SilverLogic.h>

//Cria o objeto do tipo SilverLogic com luminosidade 20
SilverLogic tabuleiro = SilverLogic(20);

//Função que é executada apenas uma vez ao ligar o Arduino
void setup()
{
 //Inicializa com a cor azul na posição 4,0
 tabuleiro.inicializar(4,0);
 //Ilumina em vermelho na posição 7,0 e amarelo na posição 2,0
 tabuleiro.setPosicao(7,0,"Vermelho");
 tabuleiro.setPosicao(2,0,"Amarelo");

}
```

Fonte: Produção do próprio autor

4 CONCLUSÃO

O dispositivo se mostrou eficiente e cumpriu seu papel em ser um dispositivo genérico. Visto as nova diretrizes de ensino ensino e as tendências mundiais de um ensino mais voltado a habilidades e não a conhecimento específicos, o produto neste trabalho descrito se apresenta como uma plataforma versátil e atual que pode inspirar futuros professores a incluir novas maneiras não só no ensino médio como também no ensino superior, de ensinar programação; todavia mesmo ainda considerando as matérias convencionais, este projeto tem aplicações para o ensino de matemática visto que álgebra é um tema rico em lógica, podendo abordar temas como progressões aritméticas, álgebra básica, lógica booleana, entre outros. A estrutura da matriz ainda pode ser usada para uma representação visual de autômatos celulares, estrturas de física aplicada para a determinação de padrões genéticos. A programação vem se demonstrando uma habilidade essencial no novo mercado de trabalho que vem se formando onde *big data* e outras tecnologias tem ganhado cada vez mais espaço.

Quanto a melhorias futuras, o projeto após bastante testado e aprovado em aulas práticas, pode ter uma versão com uma placa eletrônica dedicada a ele com o uso de menos fios. Outra possível melhoria seria a implementação de uma interface web inteligente para gerar os códigos necessário para uma configuração inicial desejada. Ainda como possível ponto de melhoria futura é a integração de mais dispositivos de interface para a introdução de outras tecnologias em sala de aula como conectividade bluetooth, wifi, integração com aplicativos mobile, entre outros.

REFERÊNCIAS

COPI, I.M. **Introdução à lógica**. 2. ed. São Paulo: Mestre Jou, 1978. 19 p.

OLIVEIRA, E. D. Tecnologia e educação. In: ENCONTRO DE PESQUISADORES DO PROGRAMA DE PÓS-GRADUAÇÃO EM EDUCAÇÃO: CURRÍCULO, 11., 2013, São Paulo. **Anais…** São Paulo: Pontifícia Universidade Católica de São Paulo, 2013.

ALVES, R.M. et al. Uso do hardware livre arduino em ambientes de ensino-aprendizagem. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA NA EDUCAÇÃO, 2012, Rio de Janeiro. **Anais…** Rio de Janeiro: Universidade Federal do Rio de Janeiro, 2012.

ZANETTI, H.A.P. et al. Prática de ensino de programação de computadores com robótica pedagógica e aplicação de pensamento computacional. In: CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 4., 2015, Maceió. **Anais…** Maceió: Instituto Federal de Alagoas, 2015.

YUANLEI SEMICONDUCTORS. **Datasheet LED 5050**. Disponível em: <http://www.yuanlei-led.com>. Acesso em: 15 set. 2018.

REVOLED – TECNOLOGIA EM ILUMINAÇÃO. **Aprenda a calcular a potência ideal de fonte de alimentação para ligar fitas LED**. Disponível em: <https://www.revoled.com.br/>. Acesso em: 12 ago. 2018.