
**Pontifícia Universidade Católica do Rio de Janeiro**

# Curto-Circuito: Aplicativo de ensino de programação

**Vinícius Cortat Btechs**

**Orientador: Waldemar Celes**

**Projeto Final de Graduação**

**Centro Técnico Científico – CTC**

**Departamento de Informática**

Curso de Graduação em Engenharia da Computação

Rio de Janeiro, junho de 2022


**Vinícius Cortat Btechs**

# Curto-Circuito: Aplicativo de ensino de programação

Relatório de Projeto Final, apresentado ao programa de engenharia da computação da PUC-Rio como requisito parcial para a obtenção do título de Bacharel em Engenharia da Computação

Orientador: Waldemar Celes

Departamento de Informática

Rio de Janeiro

Junho de 2022

# Resumo

Vinícius Cortat Btechs, Waldemar. **Curto-Circuito: Aplicativo de ensino de programação.** Rio de janeiro, 2022. 28p. Relatório de Projeto Final – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Nos últimos anos tem surgido uma demanda crescente pelo conhecimento de programação em diversas áreas de atuação, mas é somente através de um curso especializado em informática que um jovem tem seu primeiro contato com programação.

Com isso, este projeto teve como objetivo a implementação do Curto-Circuito, um aplicativo feito em Unity que visa ser o primeiro contato com a programação. Através de diversos desafios, a aplicação ensina conceitos como linha de comandos, condicionais e repetições, e mostra que existem diversas maneiras de solucionar um problema, ao mesmo tempo que estimula a busca pela melhor solução.

Palavras-chave

programação, ensino, Unity, aplicação, educação

# Abstract

Vinícius Cortat Btechs, Waldemar. **Curto-Circuito: An application to teach programming.** Rio de janeiro, 2022. 28p. Relatório de Projeto Final – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

In the last few years, the demand for programming knowledge has been growing in many different fields, but it’s only with specific IT courses that someone would have their first contact with programming.

Therefore, this project’s objective was to create Curto-Circuito, an application made in Unity which goal is to be the first contact with programming. Through different challenges the application teaches concepts such as command line, conditionals, and repetition, and shows that there are different ways to solve a problem, as well as stimulates the search for the best solution.

Palavras-chave

programação, ensino, Unity, aplicação, educação

# Sumário

# 1. Introdução

## 1.1 Motivação

A programação vem cada vez mais conquistando protagonismo no mundo em que vivemos, de forma que, hoje, o entendimento dos seus conceitos básicos é fundamental no mercado de trabalho, nas mais diversas áreas de atuação.

Diversas empresas utilizam softwares para resolver problemas e otimizar processos, fazendo com que seus empregados necessitem entender sobre o funcionamento de programas, que ficam cada vez mais complexos. Dessa forma, funcionários de áreas que a princípio não têm relação com a parte de tecnologia, passam a precisar de um conhecimento de programação para utilizá-los.

Assuntos como biologia, estatística e design são exemplos de interdisciplinaridade com informática. O primeiro tem relevância na área da bioinformática, principalmente em assuntos como genética e DNA[1]. O segundo faz uso da linguagem R[2] para estatística computacional. E o terceiro faz uso de linguagens HTML, CSS e Javascript na criação de páginas web.

## 1.2 Definição do problema

A programação não é ensinada no ensino médio na grande maioria dos colégios. Com isso, alunos que se interessam pelo assunto são forçados a buscar esse conhecimento fora da escola, seja em sites ou em cursos extracurriculares, que podem não ser dedicados ao público jovem ou acrescentar um custo financeiro.

Outro problema é a desistência de alunos nos cursos superiores ligados a informática, pois muitos dos alunos que se matriculam em cursos como engenharia ou ciência da computação nunca tiveram contato com a programação, e, ao se depararem com a dificuldade do curso, acabam desistindo logo no primeiro ano. Isso resulta em grandes prejuízos para a faculdade por conta de ociosidade de professores, funcionários e infraestrutura, além dos recursos aplicados pelo aluno que desistiu.

Atualmente, a maneira mais comum para se obter conhecimento de programação é através de cursos online criados por empresas como a Udemy[3] ou através de plataformas de autoensino como a Codeacademy[4], mas esses cursos são, em sua maioria, voltados para o público adulto, deixando jovens adolescentes sem muitas opções.

# 2. Soluções existentes para o ensino de programação voltadas para crianças e jovens

## 2.1 Scratch

Scratch[5] é uma linguagem de programação desenvolvida pelo Instituto Tecnológico de Massachussetts e, atualmente, é muito utilizada na educação de programação focada em crianças. Scratch está presente em mais de 150 países e foi traduzido para 60 línguas, incluindo português.

Scratch é considerado mais acessível por utilizar uma interface gráfica que permite construir programas utilizando blocos, em que cada bloco executa um comando. Estes comandos podem se encaixar livremente para que sejam executados numa determinada ordem.

O objetivo do Scratch é ser uma linguagem feita para ensinar conceitos matemáticos e computacionais, e normalmente são desenvolvidos jogos ou animações. A Figura 1 demonstra um exemplo da tela do Scratch 3.0.

Scratch possui uma biblioteca completa, desde criação de varável a criação de funções, repetições, condicionais, sub-rotinas e eventos. O problema de possuir tantas funcionalidades, é que pode causar resistência a jovens que se deparam com tanta informação.

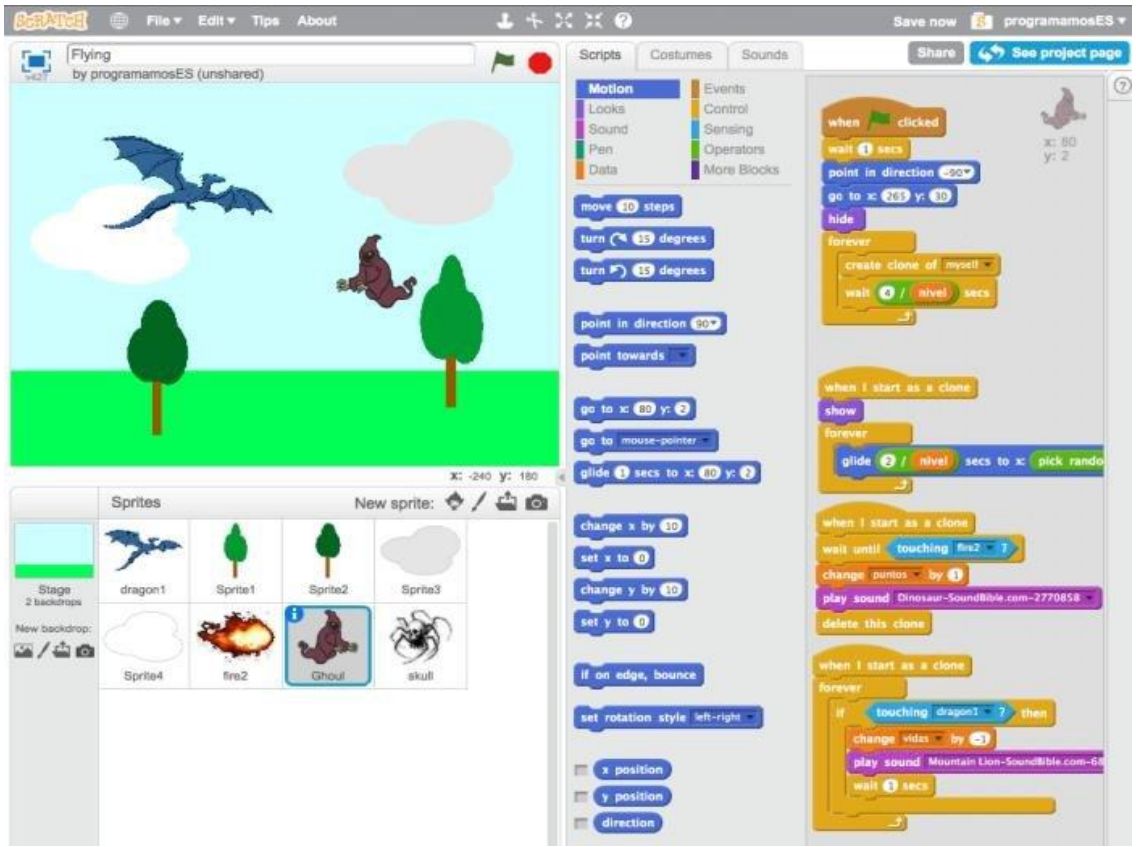


*Figura 1- Exemplo de Código em Scratch*

## 2.2 Lightbot

Lightbot[6] é um jogo em estilo enigma com a proposta de ensinar programação. Nele, o jogador deve controlar um robô por um espaço 3D, levando-o em direção a todos os quadrados azuis, acendendo-os conforme passa por cima deles.

O ensino de programação está na maneira como o robô é controlado e na interface do jogo há um espaço onde é possível arrastar diferentes imagens. Cada imagem disponível representa um comando para o robô executar. Assim, os comandos básicos como andar, girar para direita, girar para esquerda, acender chão e pular representados pelas imagens de uma seta apontando para cima, seta curvada para direita, seta curvada para esquerda, lâmpada e mola, respectivamente.

O jogo também possibilita criar funções que permitem chamar uma mesma sequência de comandos diversas vezes e até mesmo fazer chamadas recursivas, apesar de que é uma recursividade infinita.

Os principais conceitos ensinados pelo Lightbot são sequenciamento de comando, chamada e criação de funções, condicionais e repetições. Seu público-alvo são pessoas que estão entrando em contato com a programação pela primeira vez. Contudo, a dificuldade de cada desafio aumenta progressivamente e rapidamente pode se tornar inacessível para jovens.

A Figura 2 demonstra um exemplo de interface do Lightbot.

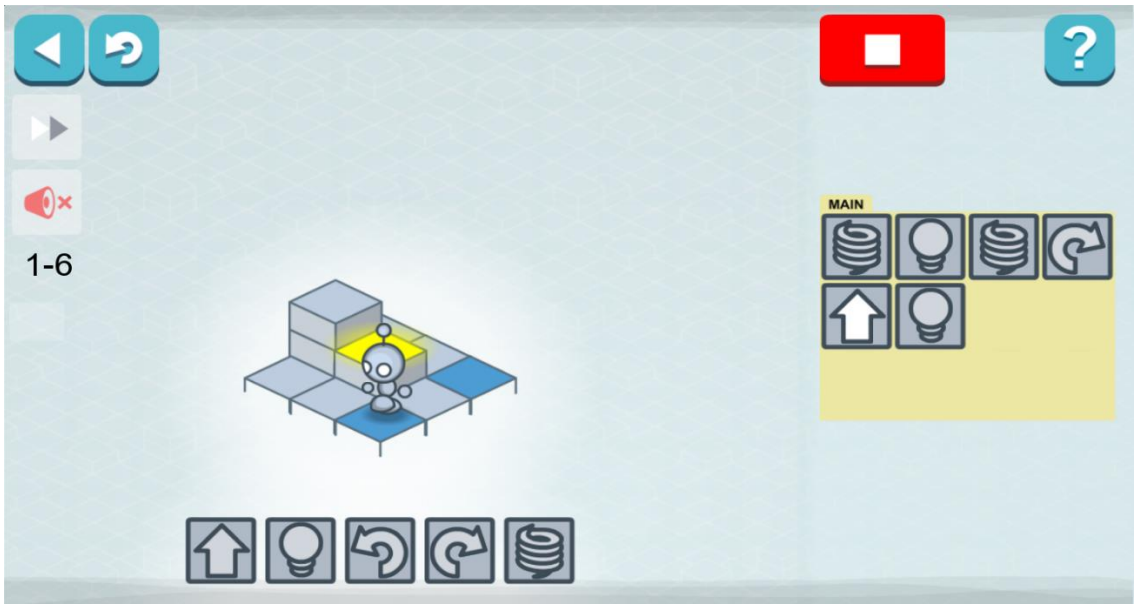


*Figura 1 - Exemplo de interface do Lightbot*

Assim**,** a ideia principal do Lightbot é ensinar programação de maneira “secreta”, isto é, o jogador está aprendendo sobre conceitos de programação através da prática enquanto se diverte, mas não há ensino teórico por trás, de forma que os conceitos são ensinados apenas nas instruções de “como jogar”.

# 3. Proposta

A proposta deste projeto foi desenvolver uma aplicação de ensino de programação similar ao funcionamento do Lightbot, mas com desafios mais simples de serem resolvidos, tornando-o mais acessível ao público jovem.

Para a criação desta aplicação foi utilizado o Unity[7], uma plataforma de desenvolvimento em tempo real que possibilita a criação de jogos, animações, efeitos especiais e modelagem 3D. A linguagem utilizada para criar scripts no Unity[8] é o C#[9]. A Figura 3 demonstra um exemplo da interface do Unity durante a criação de um jogo.

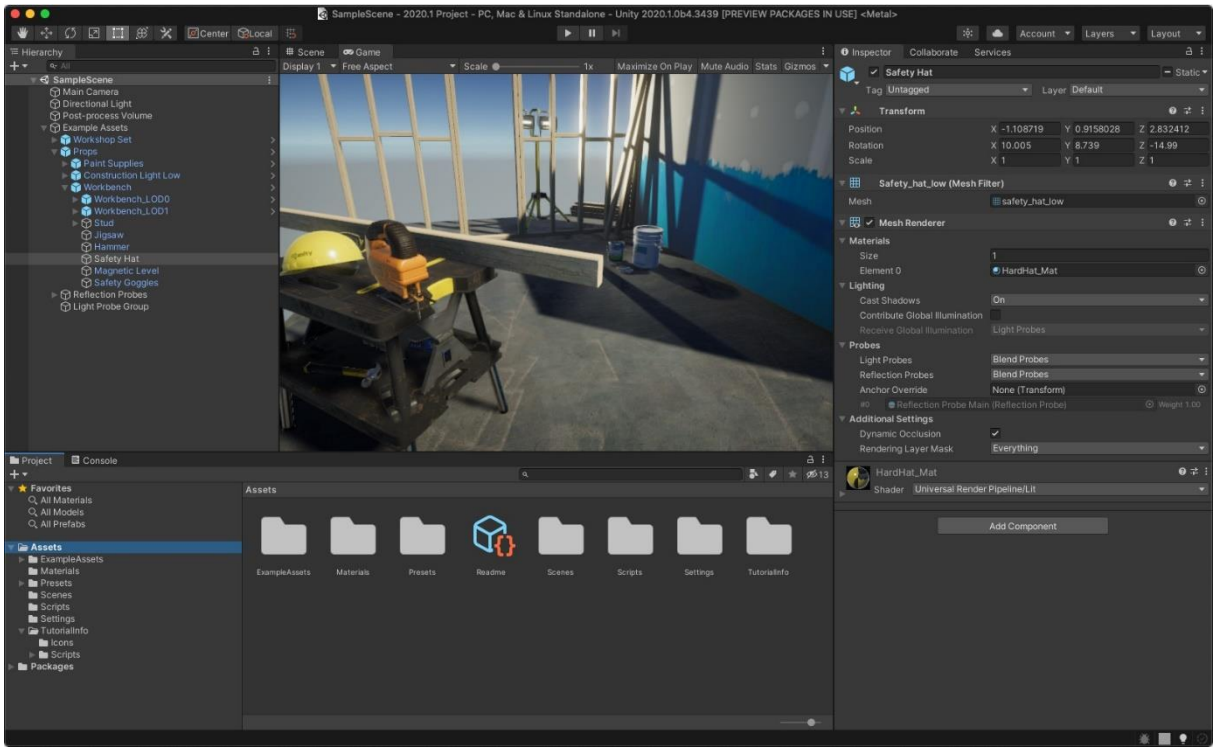


*Figura 3 - Exemplo de interface do Unity*

# 4. Atividades realizadas

Neste projeto foi criado o aplicativo Curto-Circuito[10]: uma aplicação de ensino de programação focada no aprendizado através da resolução de desafios lógicos.

## 4.1 Características da aplicação

O aplicativo não ensina diretamente sobre programação. No entanto, o conceito é abordado em temas como sequência de comandos, condicionais e repetições, mas apenas como instruções de como solucionar os desafios. Uma das propostas é que o aplicativo seja o primeiro contato com conceitos de programação, assim como o Lightbot ensina a programar de maneira “secreta”.

A aplicação é composta por diversos desafios, de maneira que todos eles possuem múltiplas soluções. Uma delas indicada como a solução ideal. Dessa forma mesmo que o usuário não consiga solucionar perfeitamente certo desafio, ainda será possível continuar com os próximos.

O público-alvo são jovens adolescentes ou pessoas que ainda não tiveram seu primeiro contato com a programação. Com isso em mente, a interface foi pensada para ser simples e minimalista. Os controles são apenas o botão esquerdo do mouse e o uso de texto foi reduzido com a utilização símbolos.

O aplicativo pode ser baixado apenas para Windows, mas existe portabilidade para WebGL[11]. Então através de um navegador compatível[12], é possível ser usado por usuários de Linux e MacOS.

## 4.2 Fluxo de tela

Para melhor entendimento do fluxo do programa, A figura 4 mostra o fluxo de telas, que são apresentados nas secções subsequentes.

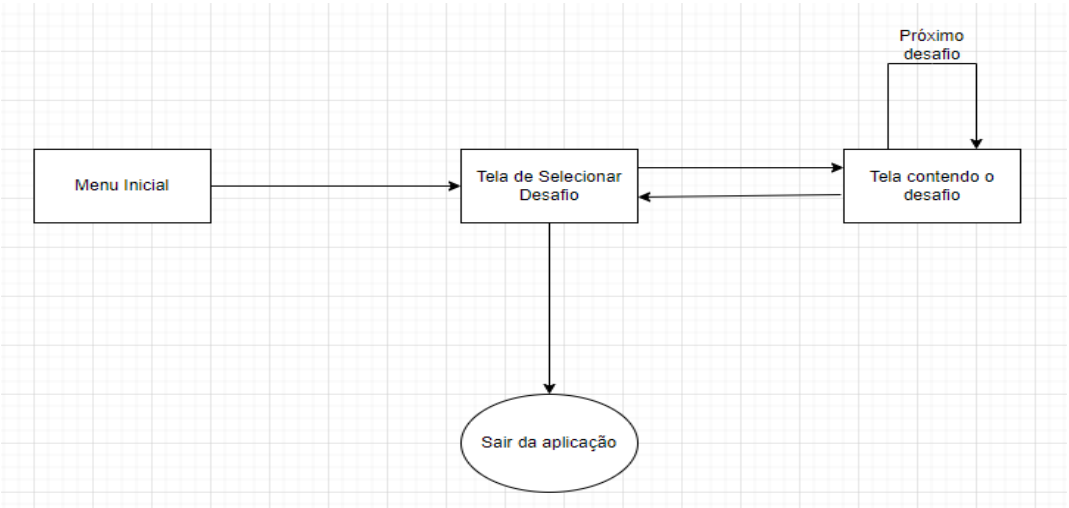


*Figura 4 – Fluxo de tela*

### 4.3 Tela de Selecionar Desafio

Nesta tela, o usuário pode escolher qual desafio quer resolver. No primeiro momento, somente o desafio 1 está disponível; ao ser resolvido, o desafio 2 se torna acessível, seguindo esta ordem até que todos os 15 desafios sejam desbloqueados.

Os desafios possuem três status: “Incompleto”, “Melhor solução não encontrada” e “Melhor solução encontrada”, representados pelas cores vermelho, amarelo e verde, respectivamente. Para liberar o próximo desafio, não é necessário encontrar a melhor solução, mas é necessário achar uma solução.

O aplicativo salva o progresso do usuário, e nesta tela também se encontra um botão para resetar seu progresso, além do botão para fechar o programa.

Na Figura 5 pode-se observar um exemplo de tela de selecionar desafio.

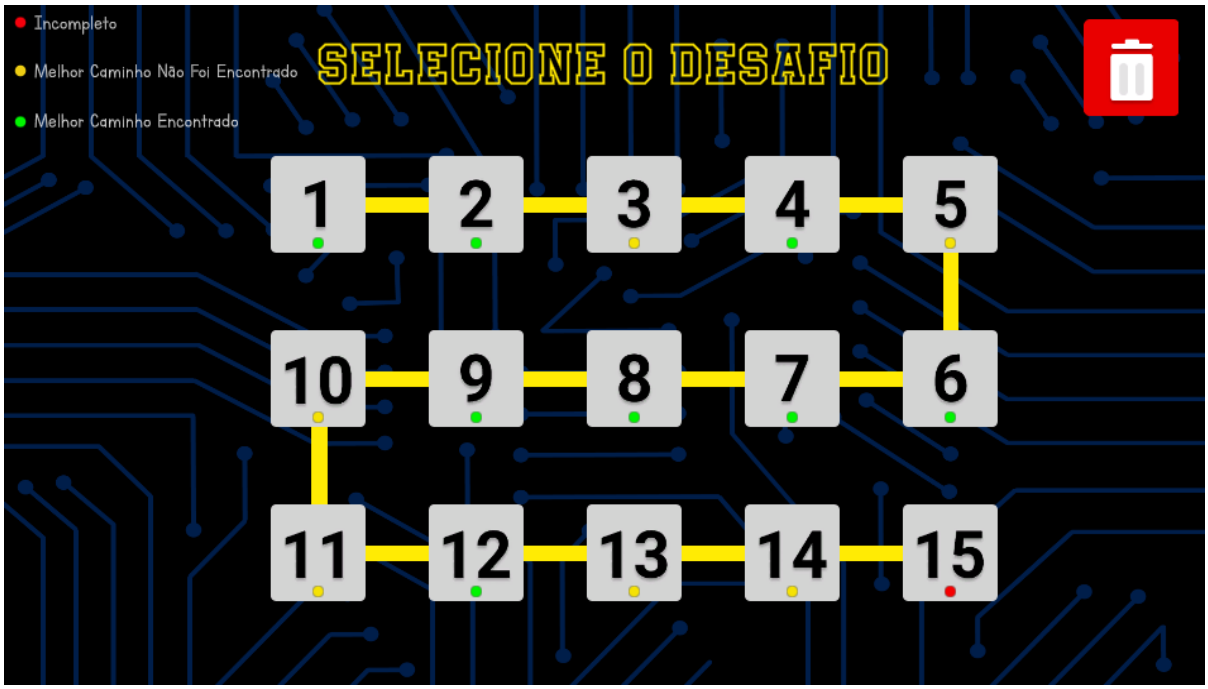


*Figura 5 – Tela de Selecionar Desafio*

### 4.4 Tela de Desafios

A seguir será explicado o funcionamento de cada tipo de desafio. Cada um deles ensina um tópico diferente sobre conceitos de programação.

### 4.4.1 Objetivo do Desafio

Os desafios têm como base o comportamento de um circuito. Este é composto por uma fonte e lâmpadas a serem acesas.

Para cada desafio haverá um circuito diferente. Estes circuitos sempre possuem uma fonte na mesma posição, com um lado positivo, indicando o começo, e o lado negativo, indicando o fim. Duas lâmpadas, cuja posições mudam de acordo com o desafio, devem ser “alimentadas”. O objetivo é controlar a eletricidade que sai do lado positivo da fonte por cada vértice do circuito de modo que ambas as lâmpadas sejam acessas, e chegar no lado negativo da fonte com o menor circuito possível. Na Figura 6 ilustra um exemplo de desafio.

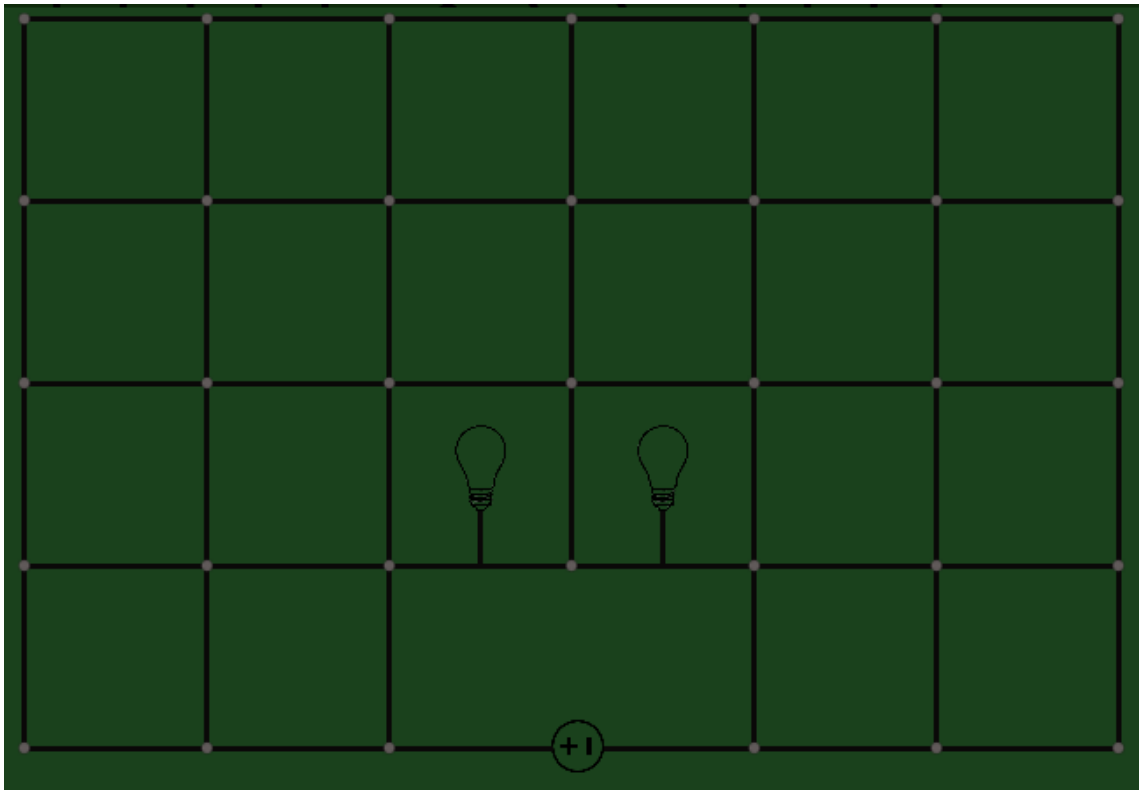

*Figura 6 – Desafio 1*

### 4.4.2. Controles

O usuário controlará a eletricidade através de 4 comandos básicos: esquerda, direita, cima e baixo, representados por uma seta apontando para a respectiva direção.

Primeiramente, o usuário deve montar a sequência de comandos desejada para solucionar o desafio, e ao pressionar o botão que representa “ligar”, uma animação vai caminhando a eletricidade comando por comando.

Se ambas as lâmpadas forem ligadas e a eletricidade chegar no lado negativo da fonte, então uma tela de êxito aparecerá para o usuário. Essa tela

de êxito também indica se o usuário obteve sucesso em percorrer o menor caminho, e o leva para o próximo desafio. Se o objetivo não for alcançado, o usuário terá que tentar novamente.

Na Figura 7 ilustra o mesmo desafio da Figura 6, dessa vez com a interface inteira a mostra e com o desafio solucionado.

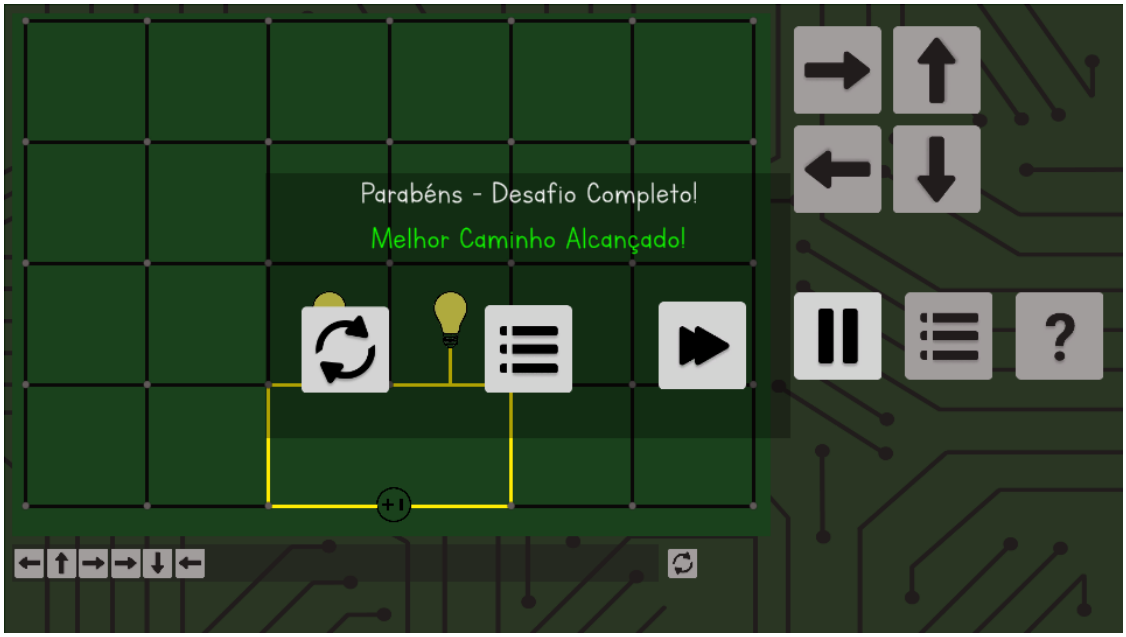


*Figura 7 – Desafio 1 com solução*

Existem três tipos de desafios diferentes, cada um deles foca em um conceito de programação, sendo eles execução e sequência de comandos, condicional e repetições.

### 4.4.3 Tipo I – Execução de Comandos

No início do primeiro desafio, o usuário será apresentado um manual de instruções, mostrando qual o é objetivo do problema e quais os controles da aplicação. A instrução é mostrada na Figura 8.

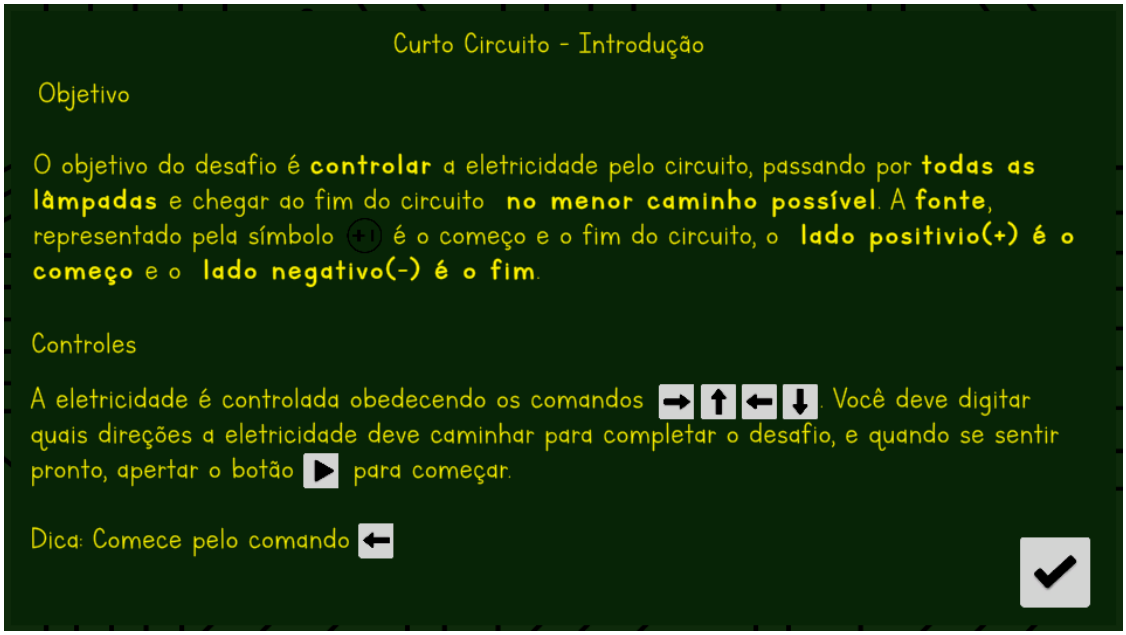


*Figura 8 – Instrução dos primeiros desafios*

O primeiro tipo de desafio tem o objetivo de treinar a lógica sequencial, progressivamente aumentando o número de comandos necessários para completar o desafio. Nesta parte, é bem prático achar o menor caminho e na maioria das vezes o usuário deve conseguir encontrar na primeira tentativa. A partir do desafio 5, algumas das arestas são removidas. Encontrar o menor caminho se torna uma tarefa mais complexa. Assim, o foco da aprendizagem passa a não ser somente resolver o enigma, mas também encontrar a melhor solução.

### 4.4.4 Tipo II – Condicionais

Ao chegar no desafio 10 entramos no tema de condicionais. Uma aresta condicional é adicionada ao circuito. Ela é representada por uma linha pontilhada, e o usuário será apresentado a um novo manual de instruções explicando como a aresta funciona (Figura 9).

Com essa nova aresta, o desafio precisará ser completado duas vezes usando uma só sequência de comandos. Na primeira vez, no lugar da aresta pontilhada ficará uma aresta normal e, na segunda vez, ela será removida.

Além de uma nova aresta, um novo botão de comando pode ser encontrado e com ele é possível criar duas pequenas sequências de comandos. Uma delas para caso a aresta condicional for uma aresta normal e outra quando a aresta for removida. A ideia é ter o funcionamento similar ao de um *if* e *else*.

A Figura 10 ilustra a criação de uma sequência de comandos condicional enquanto a Figura 11 mostra um exemplo de desafio com a aresta condicional.

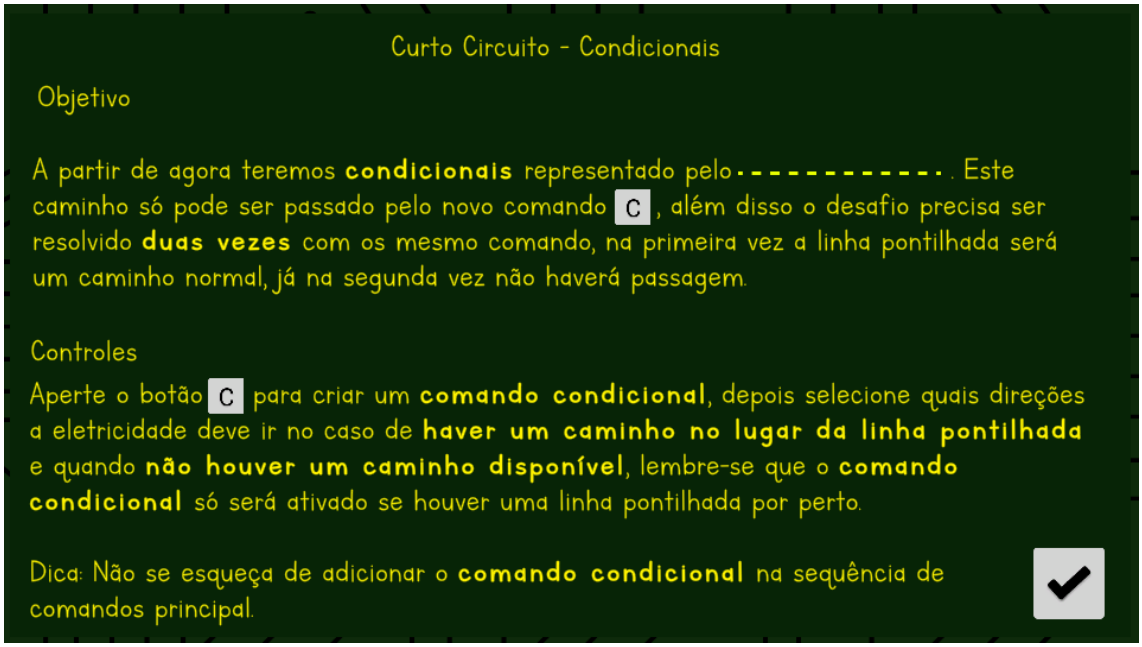


*Figura 9 – Instrução de como resolver desafios com condicionais*

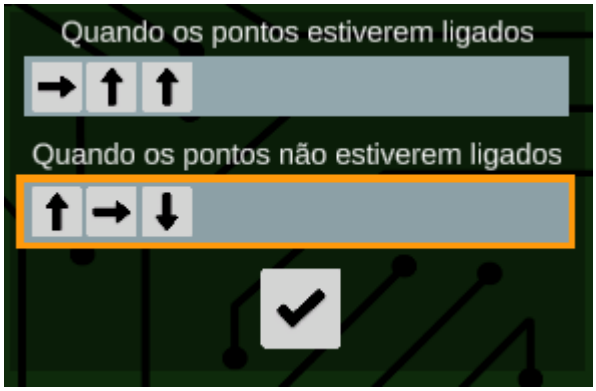


*Figura 10 – Criação de uma condicional*

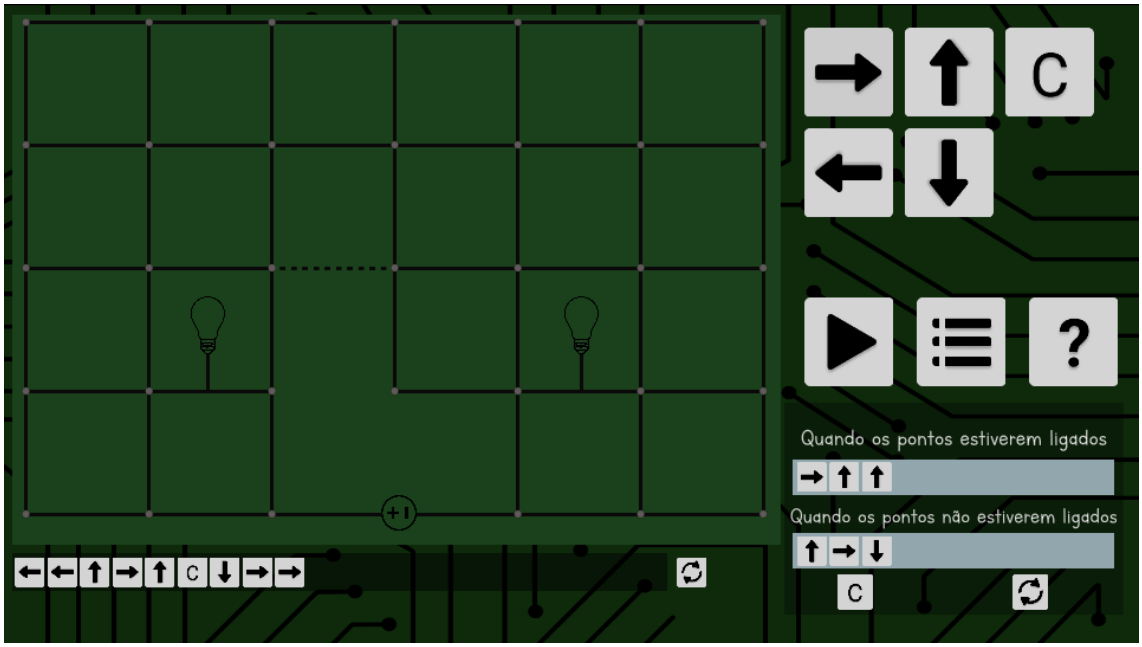


*Figura 11 – Exemplo de desafio utilizando condicional – Desafio 10*

Para solucionar desafios com condicionais é necessário que a eletricidade chegue à linha pontilhada, em seguida chamar o comando condicional, se for a primeira volta, haverá um caminho no lugar da linha pontilhada e a primeira sequência de comandos será chamada. Se o usuário obtiver sucesso em acender as lâmpadas e chegar no fim do circuito, mesmo que não tenha sido a melhor solução, o circuito começará do início, dessa vez removendo a aresta pontilhada e lendo a segunda sequência de comandos do comando condicional. Com ambas as voltas completas, a tela de êxito aparece, para obter a melhor solução neste desafio, é necessário que as duas voltas tenham o menor circuito possível.

### 4.4.5 Tipo III – Repetição

A partir do desafio 13, é adicionado um novo manual de instruções (Figura 12) contendo informações sobre repetição de comandos, além de um novo botão para permitir criar repetições. Os desafios de repetição são desafios já vistos anteriormente. O objetivo é mostrar ao usuário que é possível resolver o mesmo problema com ainda menos comandos quando apresentado a nova ferramenta.

Para fazer uso da repetição, primeiramente deve-se criar uma repetição, que é, resumidamente, uma pequena sequência de comandos. A diferença é que ao ser adicionada à linha de comando principal, é requisitado que o usuário indique quantas vezes ele quer que esta sequência de comandos se repita, sendo possível criar 4 repetições distintas por desafio.

Um exemplo de desafio é ilustrado abaixo na Figura 13, com a criação da repetição na Figura 14 e a entrada pedindo número de repetições na Figura 15.

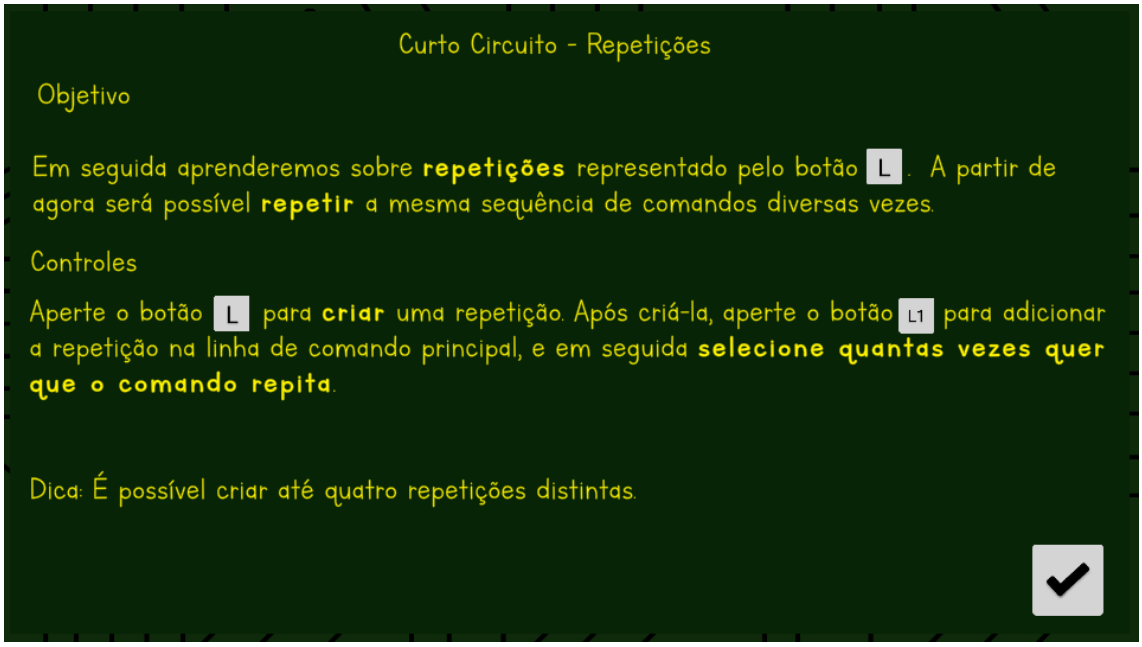


*Figura 12 – Instruções para usar uma repetição*

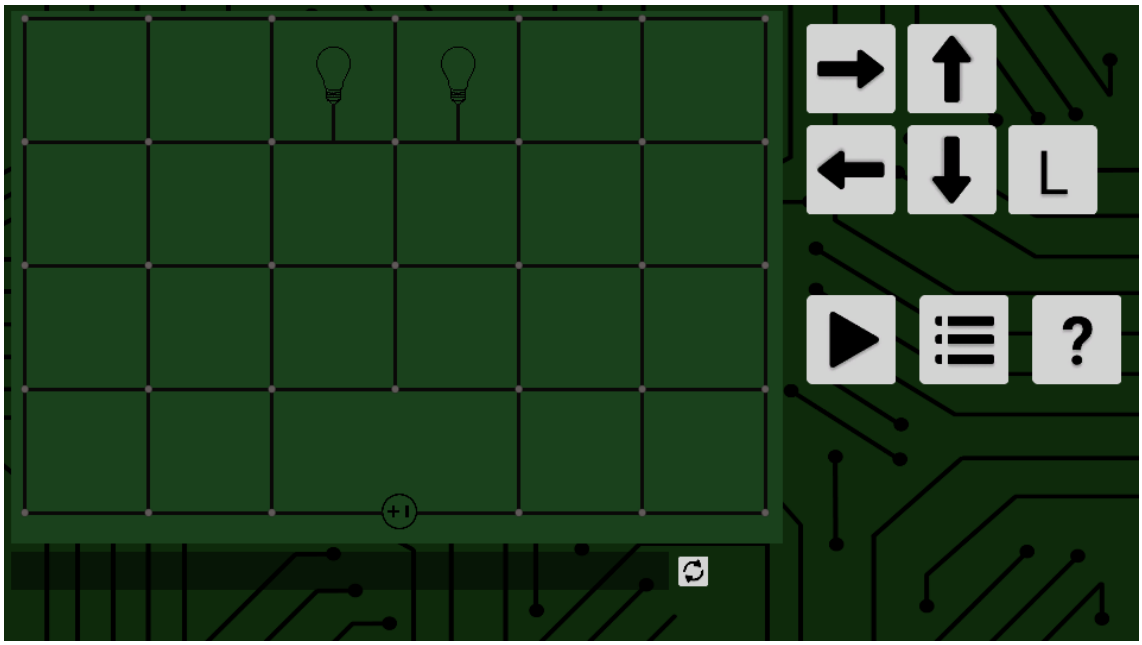


*Figura 13 – Exemplo de desafio com repetição – Desafio 13*

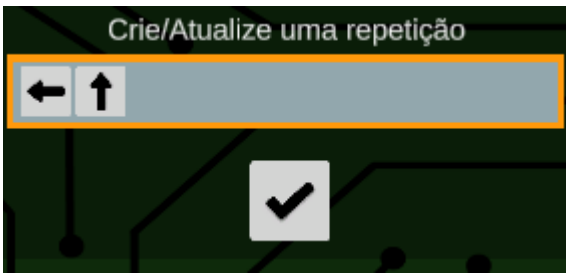


*Figura 14 – Criação de uma repetição*

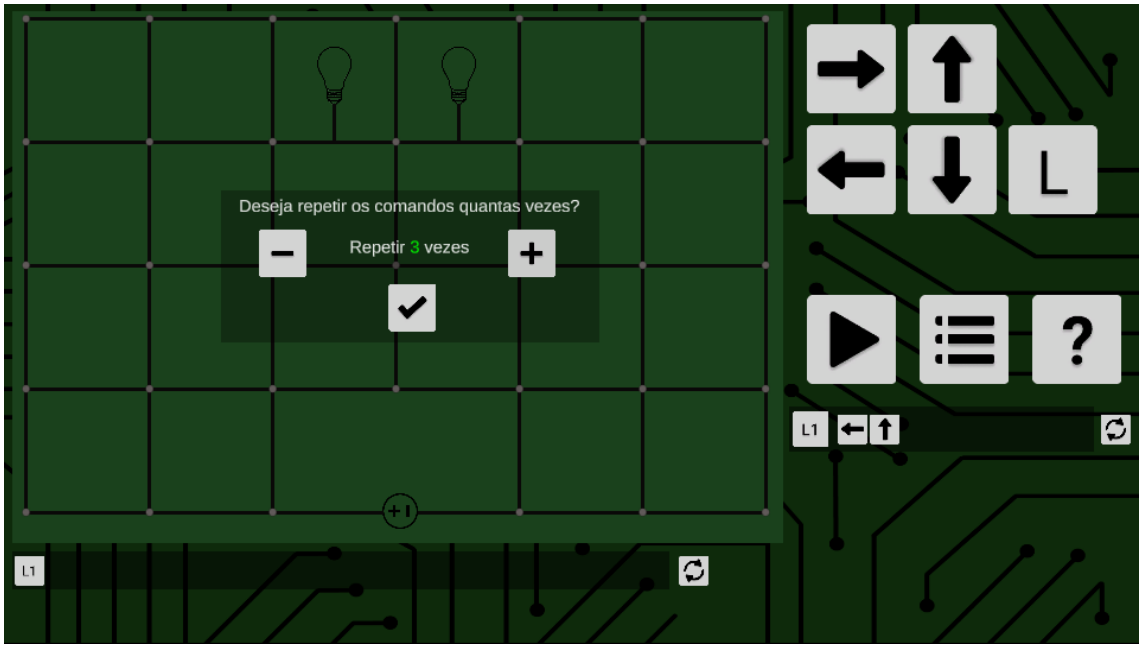


*Figura 15 – Inserindo número de repetições*

Para solucionar desafios com repetição, é necessário analisar onde um conjunto de comandos seria repetido duas ou mais vezes consecutivas e em seguida criar uma repetição que reproduza esta sequência, a chamando na linha de comando principal.

### 4.5 Arquitetura da aplicação

O aplicativo foi projetado para que seja fácil a criação de novos desafios. Dentro do Unity, temos *cenas*, é nelas que é montado a aplicação. Cada uma delas é independente da outra. Neste projeto, cada *cena* é um desafio. Além disso, também existem *prefabs* que são um conjunto de um ou mais *gameobjects*, sendo estes componentes básicos como quadrados, imagens e textos, transformados em uma classe.

Todos os componentes que compõe um desafio são formados por um único *prefab.* Assim, para criar um desafio novo basta criar uma *cena* nova, incluir uma nova instância do *prefab*, e modificar alguns valores como posição das lâmpadas e qual aresta é removida. O resto funcionará como deve, este processo todo leva em torno de 5 a 10 minutos dando grande oportunidade de expandir o número de desafios com certa facilidade.

A arquitetura do desafio tem como base o diagrama na Figura 16.

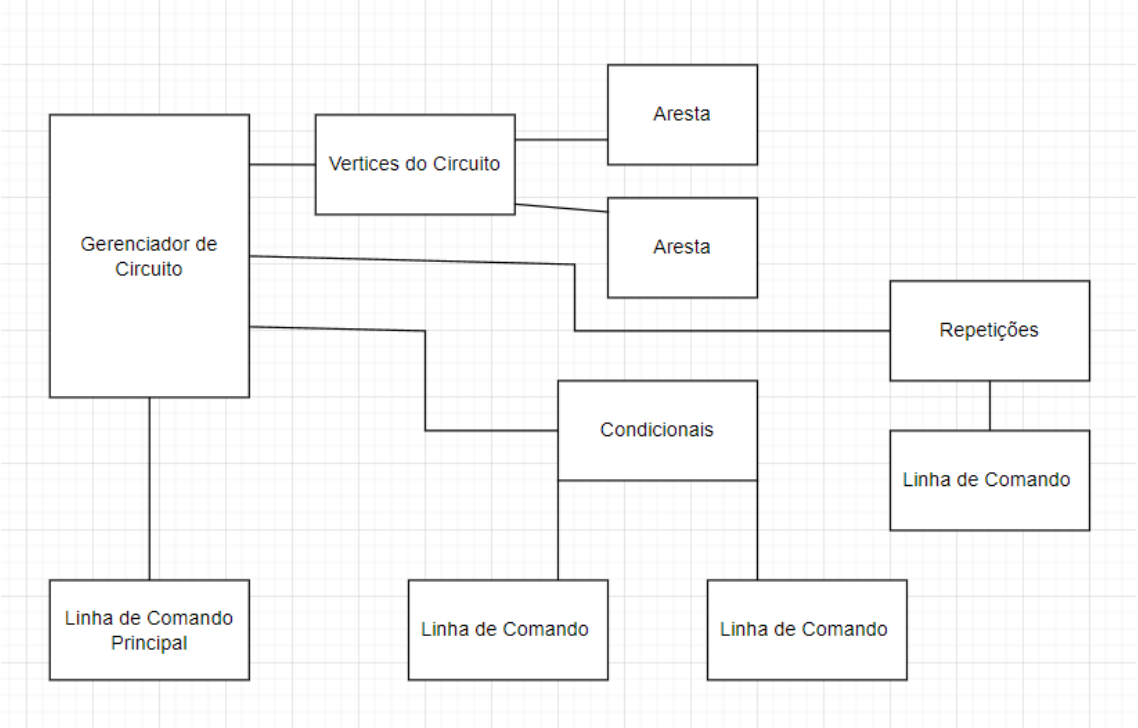


*Figura 16 – Diagrama da arquitetura do desafio*

Existem 6 classes principais na montagem de cada desafio. Gerenciador de circuito, vértice, aresta, linha de comando, condicional e repetição.

O gerenciador cuida da comunicação central entre todos os componentes, ele recebe todos os eventos de botão apertado, envia os comandos apertados para a linha de comando principal que gera a imagem e inclui um novo elemento na lista.

A execução dos comandos também é responsabilidade do gerenciador que possui método assíncrono, que a cada meio segundo verifica qual o comando sendo lido, qual vértice que se encontra a eletricidade, e se o objetivo foi concluído.

O gerenciador não interage diretamente com nenhuma aresta, ele somente tem acesso aos vértices do circuito e cada vértice possui de 2 até 4 arestas. Quando a eletricidade passa por uma aresta nova, o gerenciador envia uma mensagem para o vértice atual, que em seguida envia a mensagem para a aresta que está sendo caminhada, trocando o status dela para ativada e iluminando a aresta.

Quando uma condicional está sendo criada, todos os eventos dos botões das setas são enviados através do gerenciador para a linha de comando da condicional até que conclua sua criação. Funciona da mesma maneira para repetições.

Devido a complicações com o comportamento da condicional de precisar executar o circuito duas vezes, foi necessário criar uma cópia do método de execução dos comandos para garantir que tudo funcione como devido.

As repetições são apenas listas de comandos, então quando uma repetição é chamada, basta pausar a execução da linha de comando até que a lista inteira da repetição tenha sido lida.

# 5. Resultados

O aplicativo foi criado com sucesso e dispõe de diversos desafios que ensinam sobre conceitos de programação. Mas para verificar sua eficácia é necessário que seja testado por jovens sem conhecimento de programação, fazendo um levantamento se os conceitos foram aprendidos com sucesso.

É esperado que os 10 primeiros desafios ensinem sobre execução e sequência de comandos, ao mesmo tempo que estimulam o raciocínio lógico para resolução de problemas.

O importante do aprendizado dos desafios de condicional, é entender que independentemente da condição atual, os comandos antes e depois da condicional precisam funcionar para ambos os casos.

Com repetições, o foco é refazer desafios anteriores, dessa vez disponibilizando de uma ferramenta diferente. Mostrando ser possível executar o mesmo caminho com um número menor de comandos.

## 5.1 Soluções de desafios

A seguir será apresentada a lista de desafios criados, e algumas soluções serão discutidas mencionando o que deve ser aprendido com cada um.

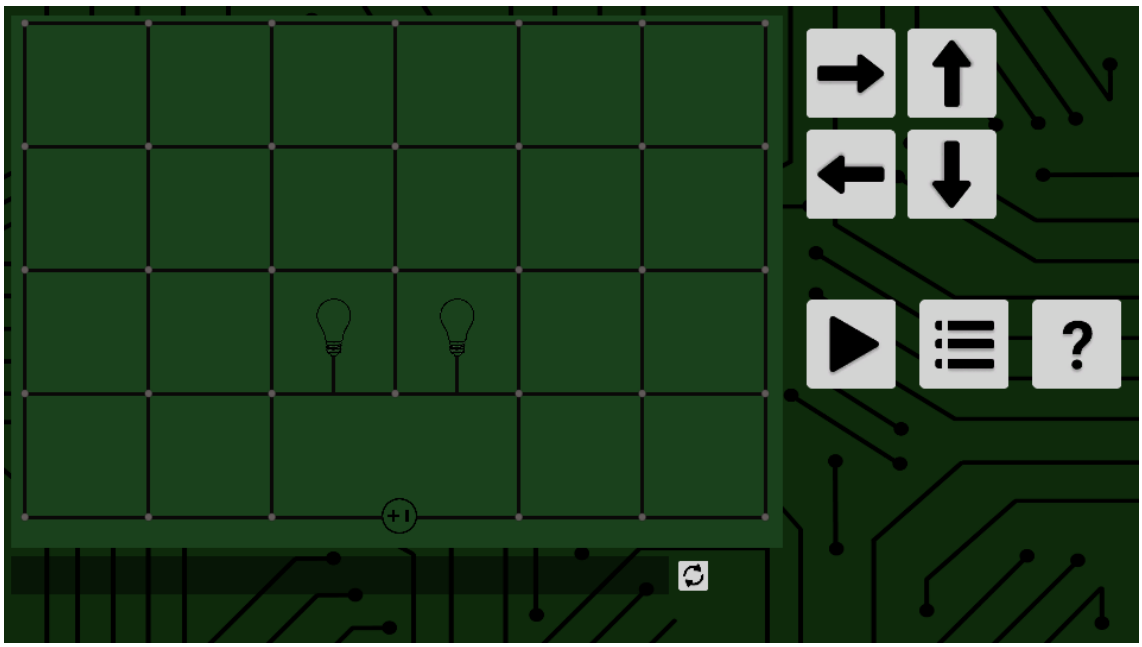

*Figura 17 – Desafio 1*

O primeiro desafio tem como solução o menor circuito possível, o objetivo disso é que o usuário se acostume com o funcionamento do aplicativo servindo como um tutorial.

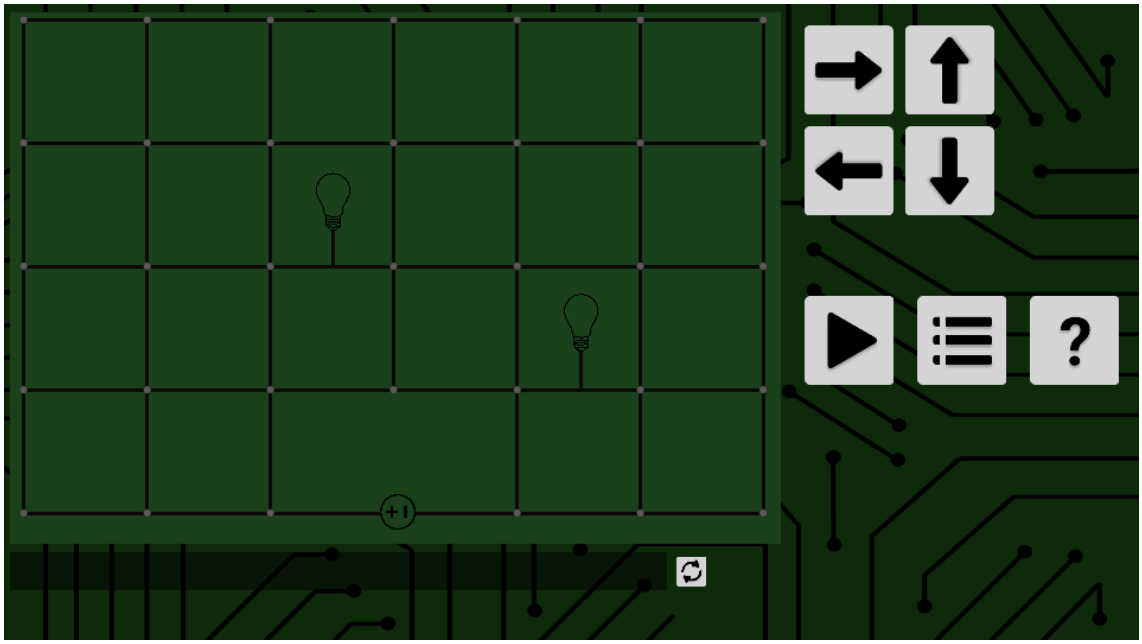

*Figura 18 – Desafio 2*

O segundo desafio desloca uma das lâmpadas na vertical e a outra na horizontal aumentado levemente a complexidade, mas ainda servindo como introdução.

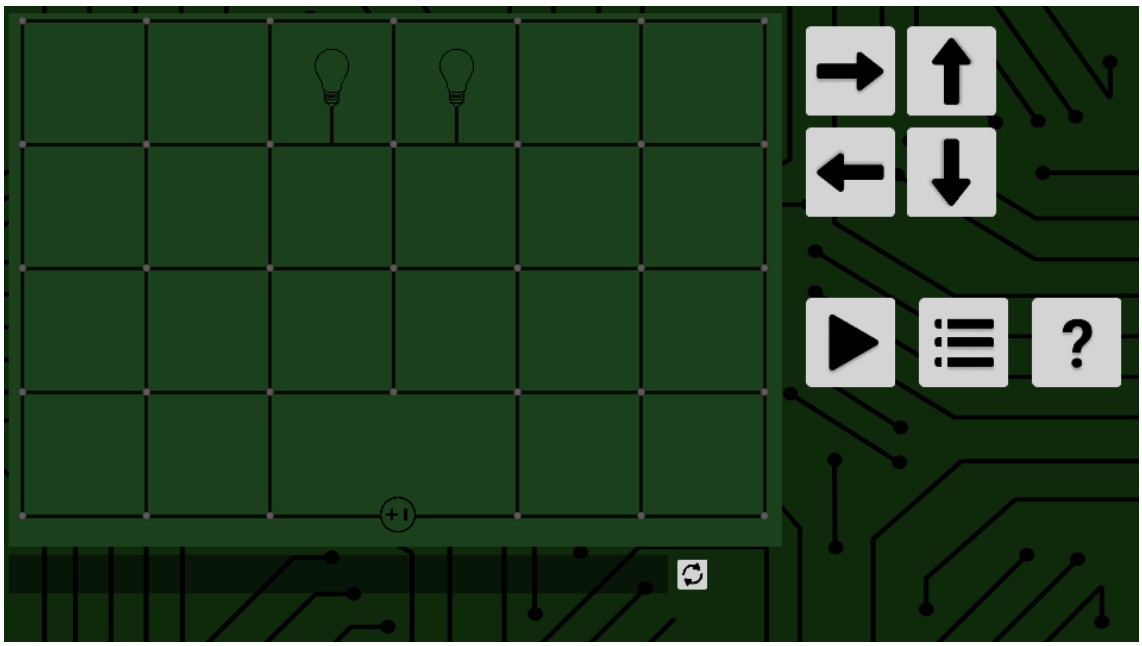

*Figura 19 – Desafio 3*

O desafio 3 possui uma solução com três chamadas consecutiva dos mesmos comandos, e será revisitado novamente no desafio 13.

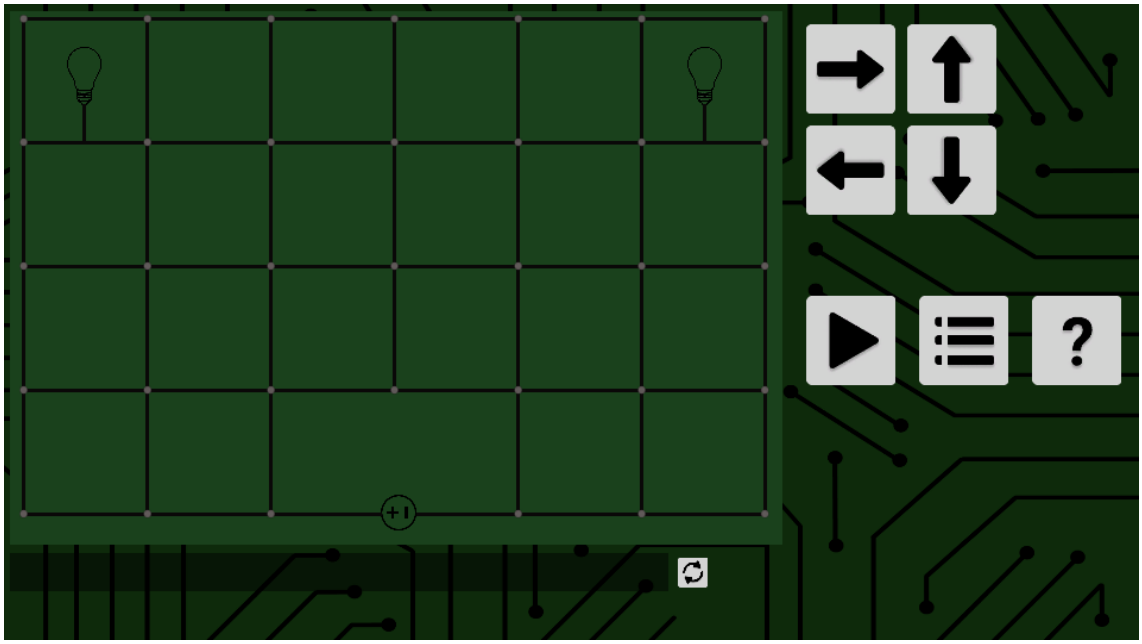

*Figura 20 – Desafio 4*

Assim como o desafio 3, o desafio 4 tem como solução a chamada consecutiva de 4 diferentes comandos 5 vezes e será revisitado no desafio 14.

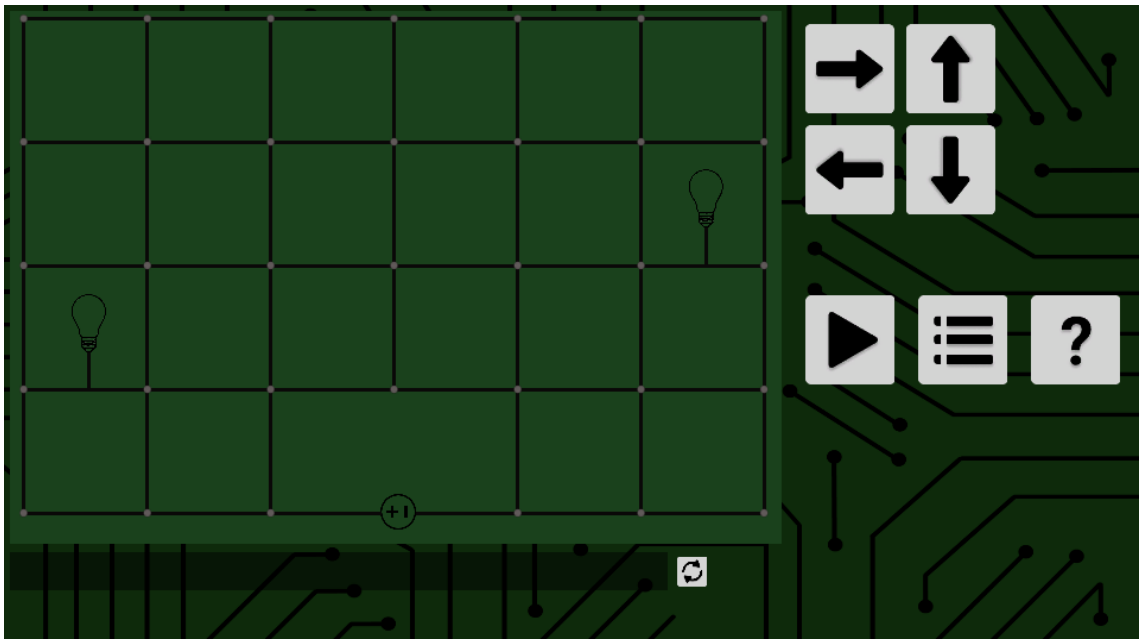

*Figura 21 – Desafio 5*

O desafio 5 não tem uma solução especial, é apenas mais um exercício para se acostumar com os comandos.

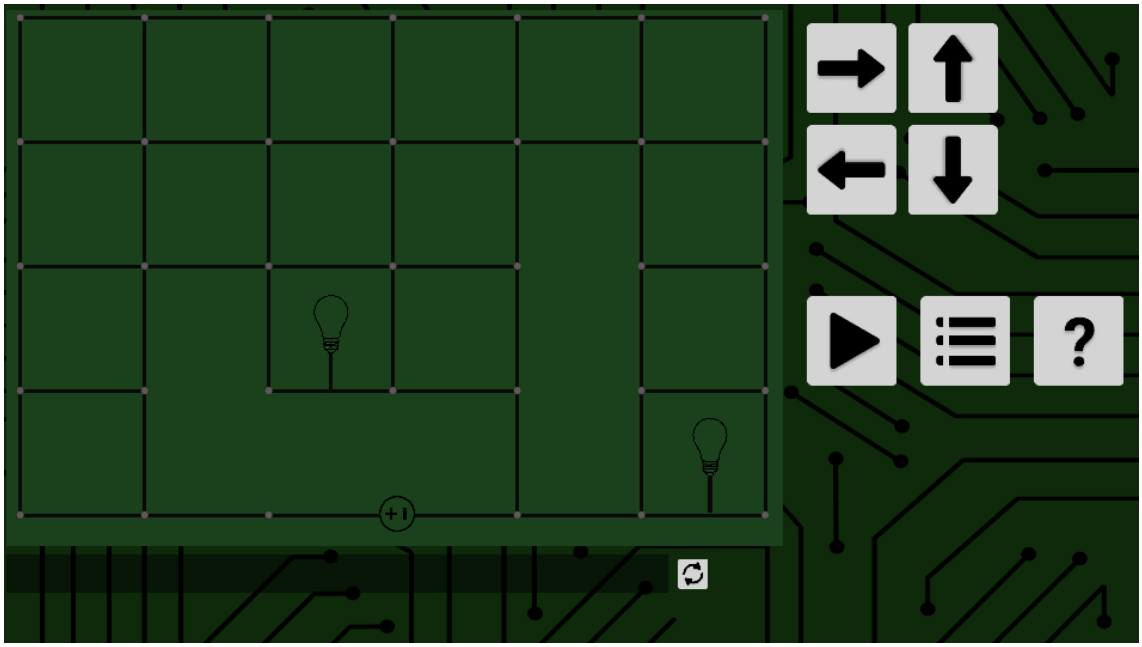

*Figura 22 – Desafio 6*

O desafio 6 tem algumas de suas arestas removidas do circuito, é necessário dar mais voltas, e assim, executar mais comandos para chegar na melhor solução.

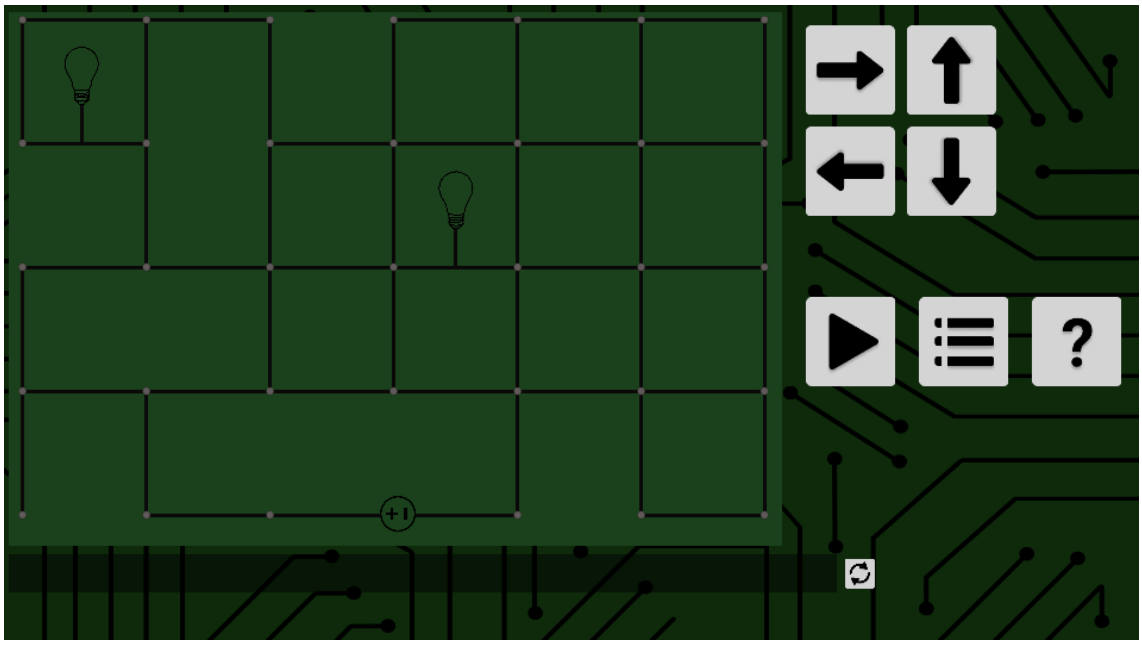

*Figura 23 – Desafio 7*

No desafio 7 uma das lâmpadas é isolada no canto, com poucas arestas em volta algumas soluções estão disponíveis para o usuário, mas o desafio de verdade se torna encontrar qual delas é a melhor.

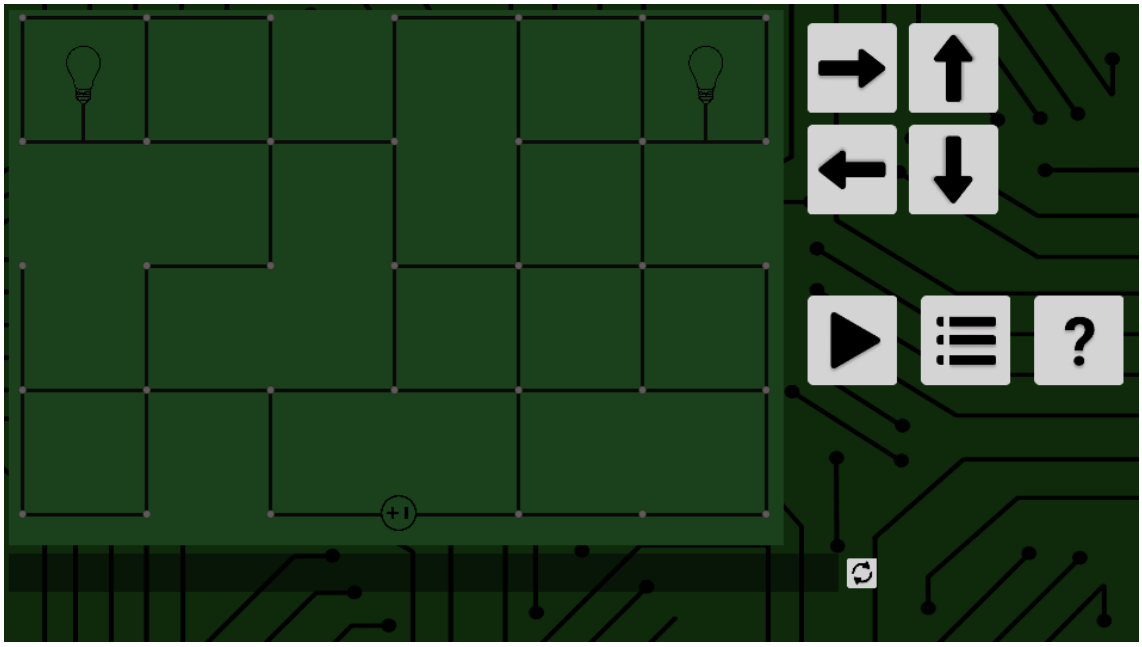

*Figura 24 – Desafio 8*

No desafio 8, ambas as lâmpadas ficam isoladas, não há uma solução especial, apenas um aumento na complexidade do desafio.

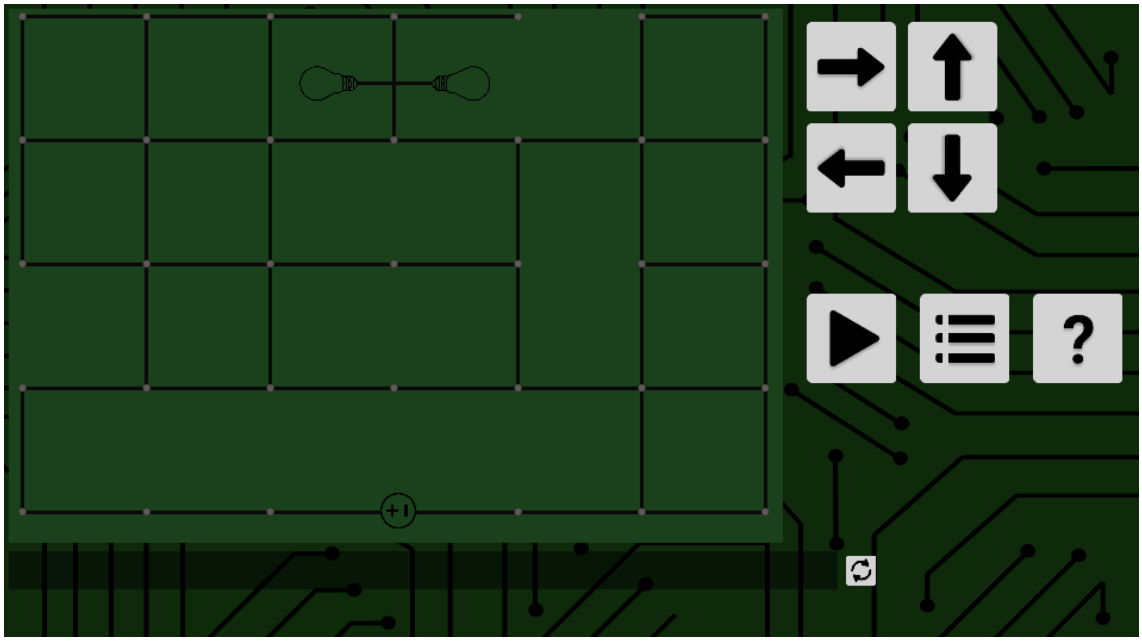

*Figura 25 – Desafio 9*

O desafio 9 é o único que contêm ambas as lâmpadas ligadas na mesma aresta ao mesmo tempo de ser o único com lâmpadas ligadas em uma aresta vertical, não possui uma solução especial.

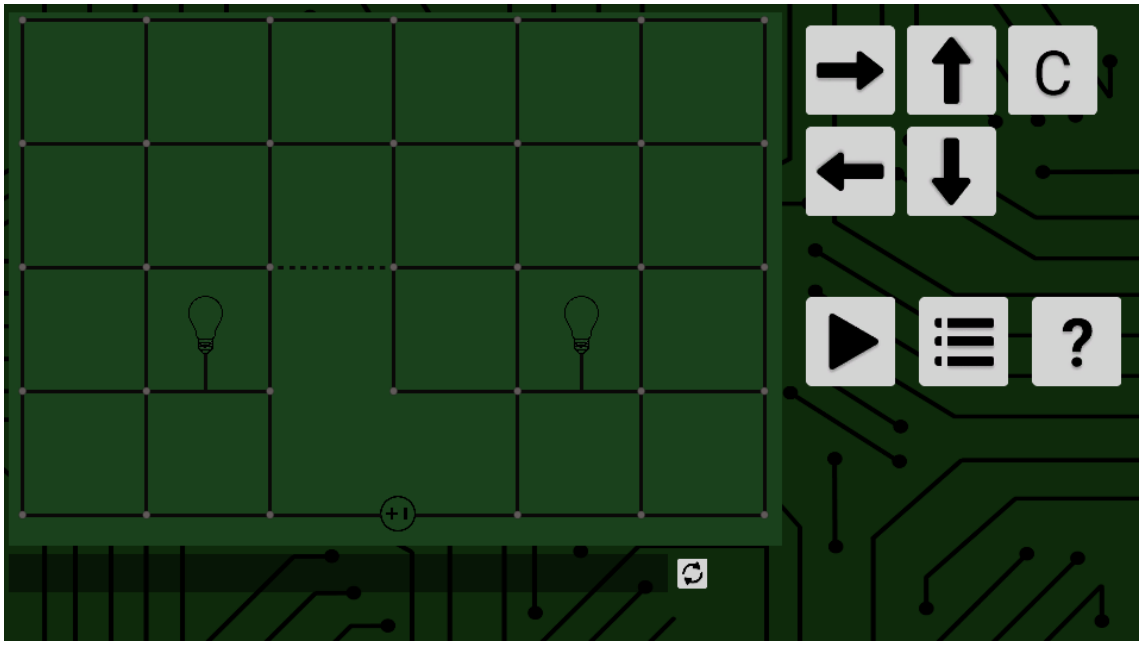


*Figura 26 – Desafio 10*

Com o desafio 10, é introduzido os conceitos de condicional. Este desafio serve como tutorial e sua solução é bem simples, seguir em linha reta no caso de ter aresta no lugar da aresta pontilhada e quando não houver, dar a volta por cima.

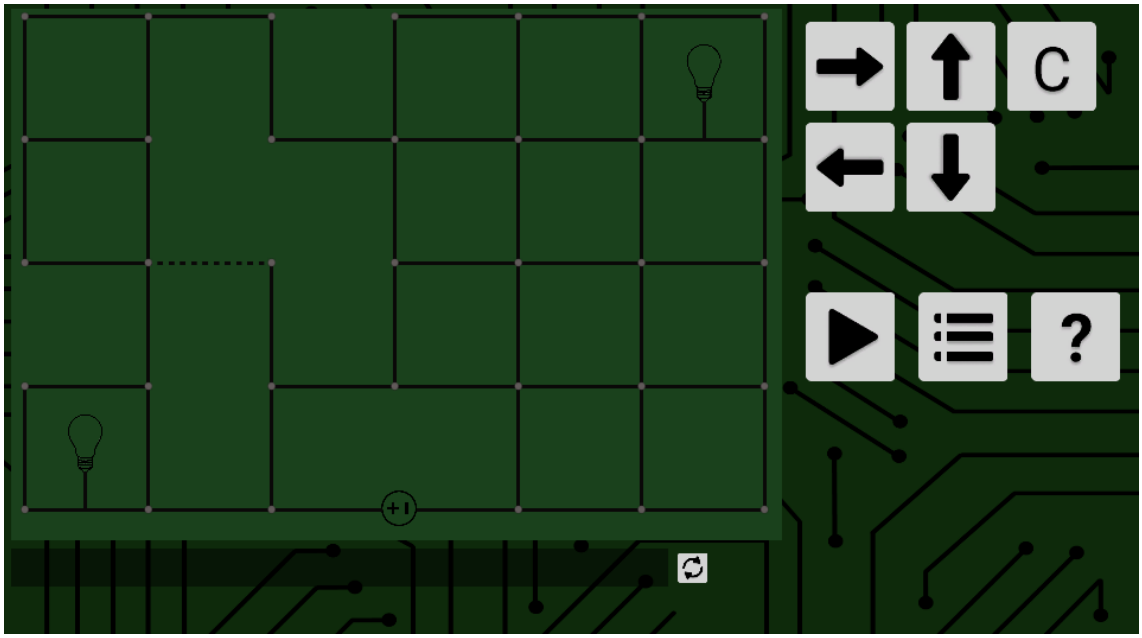


*Figura 27 – Desafio 11*

O desafio 11 já aumenta a dificuldade do circuito, o que deve ser ensinado com exercícios de condicional é que existem comandos antes da condicional e comandos que vem depois da condicional, ou seja, independentemente de qual condição está sendo testada, a eletricidade deve estar no mesmo lugar quando a condicional acabar para que o resto dos comandos faça sentido.

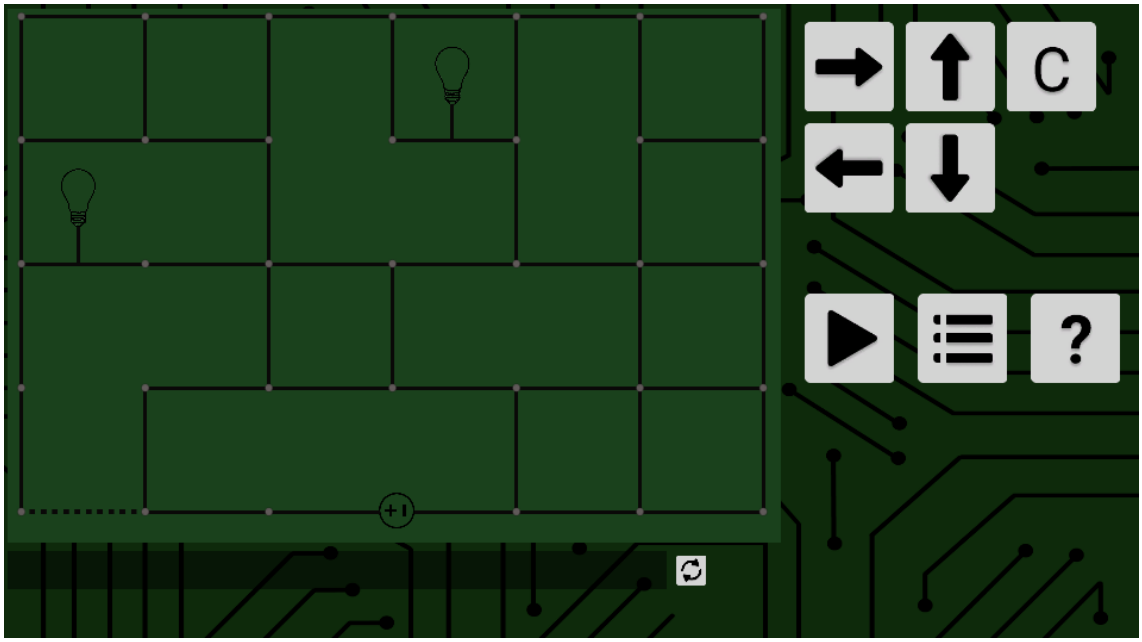


*Figura 28 – Desafio 12*

O interessante do desafio 12 é que o caminho que parece ser o melhor na primeira volta faz com que os comandos depois da condicional não funcionem para a segunda volta. A solução deste desafio tem o objetivo de voltar o foco para, primeiramente, encontrar uma solução que funcione e em seguida reconstruir os comandos procurando pelo menor caminho.

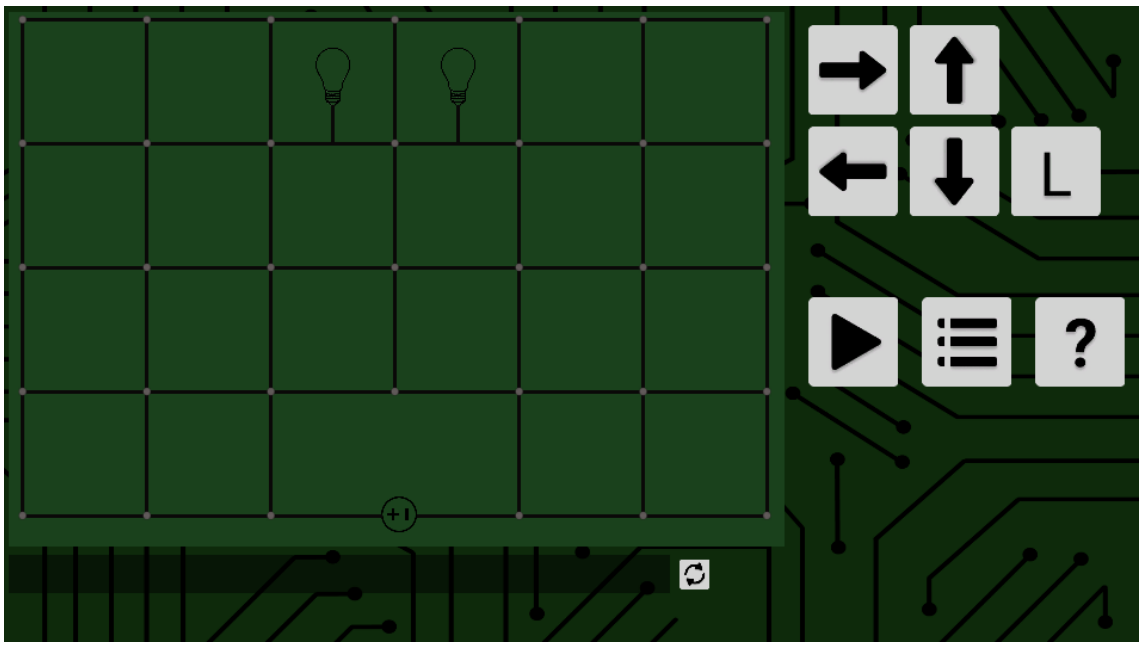


*Figura 29 – Desafio 13*

O desafio 13, como já mencionado, é idêntico ao desafio 3. Mas dessa vez é permitido usar repetições para resolver o problema, o menor caminho deste desafio é diferente do desafio 3, é esperado que o usuário use repetições para chegar na solução, apesar de não ser obrigatório. É possível solucionar usando apenas repetições

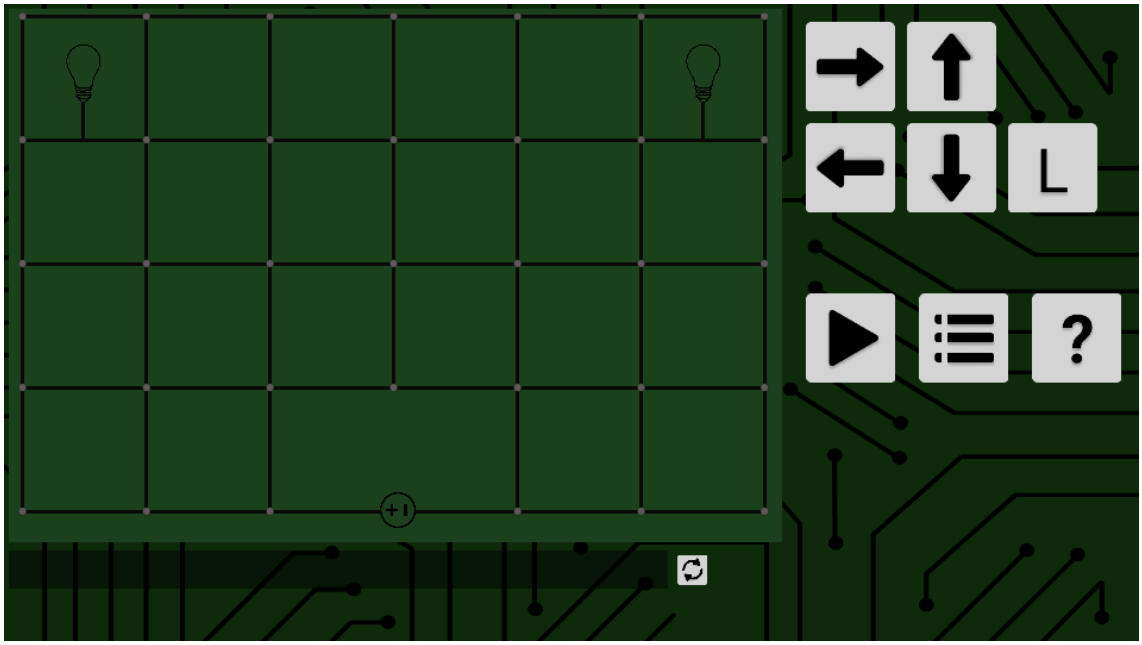


*Figura 30 – Desafio 14*

O desafio 14 é idêntico ao desafio 4, dessa vez a melhor solução inclui o uso de repetições. A melhor solução é obtida usando apenas repetições.

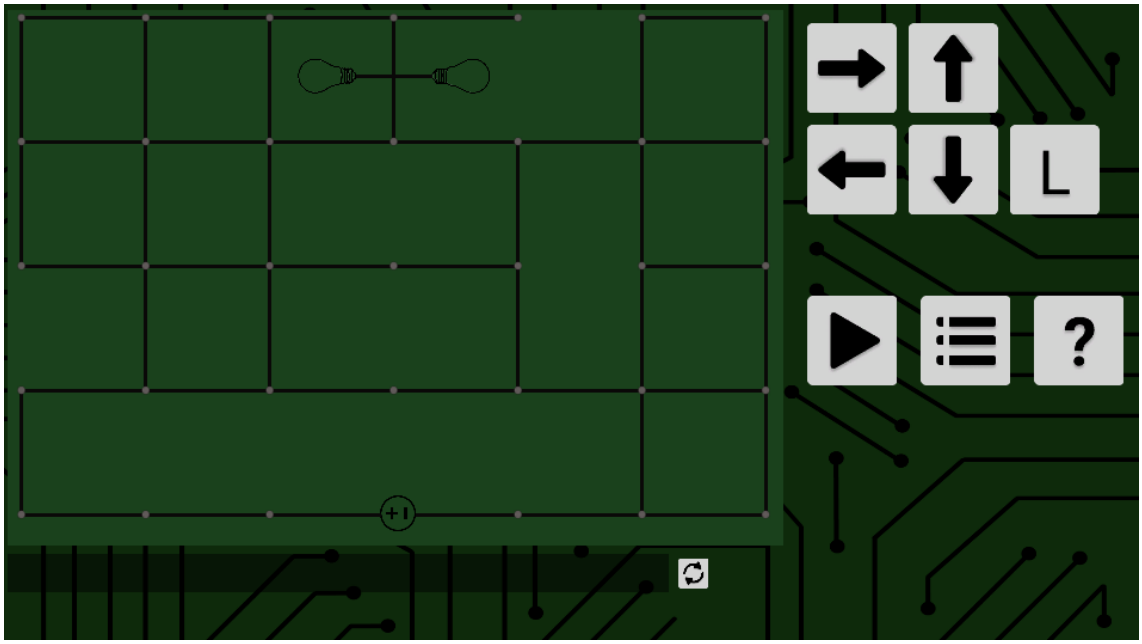


*Figura 31 – Desafio 15*

O desafio 15 é idêntico ao desafio 5, o objetivo é revisitar um problema antigo, dessa vez podendo usar uma ferramenta nova, e assim obter um resultado mais eficiente.

# 6. Conclusão

## 6.1 Acesso ao aplicativo

A aplicação está pronta com um número total de 15 desafios, 5 de sequência de comandos, 4 de sequência de comandos com algumas arestas removidas, 3 de condicional e 3 de repetição.

Ela está disponível no site itch.io[13], onde é possível jogar no próprio navegador em qualquer sistema operacional ou baixando a versão para Windows.

## 6.2 Próximos passos

Existem várias funcionalidades que não eram necessárias para a conclusão do aplicativo, mas ajudariam na experiência do usuário. Algumas delas sendo poder visualizar qual comando está sendo executado. Um botão de ajuda que dê melhores dicas para cada desafio e até mostre a solução, caso requisitado. Além disso, poderia ter sido feito uma pesquisa de cores mais acessíveis para deficientes visuais.

Como dito anteriormente, a aplicação pode ser expandida com novos desafios com facilidade, apesar de que criar a teoria por trás de cada desafio é um processo mais demorado.

# 7. Referências Bibliográficas


[1] Como a bioinformática auxilia no diagnóstico de doenças genéticas na Genomika. Disponível em: https://genomika.einstein.br/blog/como-a-bioinform%C3%A1tica-auxilia-no-diagn%C3%B3stico-de-doen%C3%A7as-gen%C3%A9ticas-na-genomika/. Acesso em 20 de junho de 2022

[2] Acerca da linguagem R. Disponível em: https://www.r-project.org/. Acesso em 20 de junho de 2022

[3] Acerca da Udemy. Disponível em: https://www.udemy.com/pt/. Acesso em 10 de junho de 2022

[4] Acerca da Codeacademy. Disponível em: https://www.codecademy.com/catalog. Acesso em 10 junho de 2022

[5] Acerca do Scratch. Disponível em: https://scratch.mit.edu/about. Acesso em: 11 de junho de 2021.

[6] Acerca do Lightbot. Disponível em: https://lightbot.com/. Acesso em: 07 de junho de 2022.

[7] Acerca do Unity. Disponível em: https://unity.com/pt. Acesso em: 11 de junho de 2021.

[8] Documentação do Unity. Disponível em: https://docs.unity3d.com/. Acesso em: 29 de abril de 2022.

[9] Acerca do C#. Disponível em: https://docs.microsoft.com/pt-br/dotnet/csharp/tour-of-csharp/. Acesso em junho de 2021.

[10] Site onde se encontra o aplicativo criado. Disponível em: https://viniciuscortat.itch.io/curto-circuito, Acesso em: 05 de junho de 2022.

[11] Acerca do WebGL. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/API/WebGL_API. Acesso em 05 de junho de 2022.

[12] Compatibilidade com WebGL. Disponível em: https://phet.colorado.edu/pt_BR/webgl-disabled-page. Acesso em 05 de junho de 2022

[13] Acerca do Itch.io. Disponível em: https://itch.io/. Acesso em: 03 de junho de 2022.