DESVENDAR A TECNOLOGIA PARA CRIAR TECNOLOGIAS

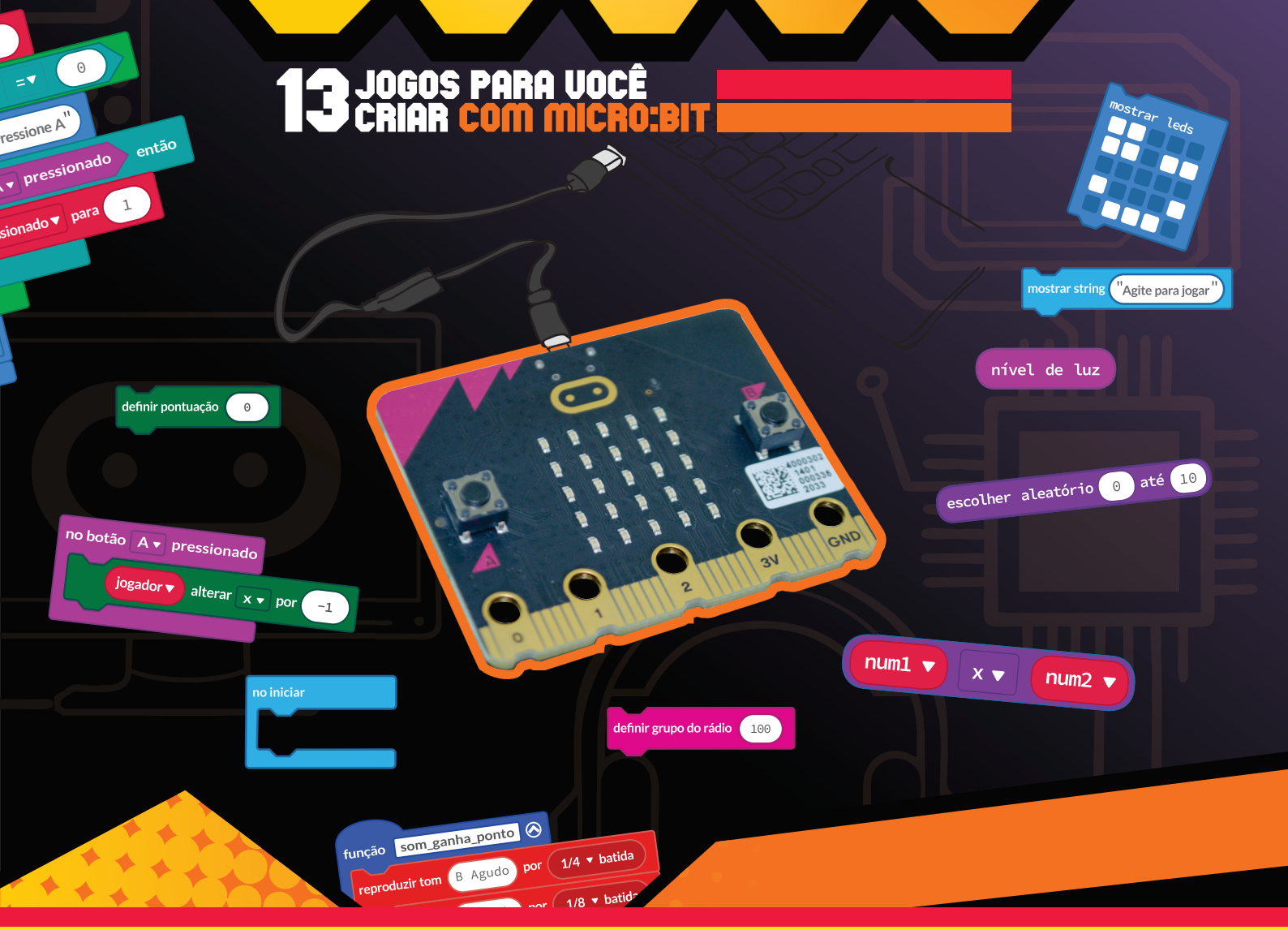
6º ano | 7º ano

ensino fundamental

HacKids editora

PROGRAMANDO JOGOS

13 JOGOS PARA VOCÊ CRIAR COM MICRO:BIT



MAKE PLAY LEARN

WAGNER RODRIGUES

Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)

Rodrigues, Wagner de Paula
Programando jogos com micro:bit / Wagner de Paula Rodrigues. -- Londrina, PR : Hackids, 2021.

ISBN 978-65-996767-0-3

1. Educação - Tecnologia 2. Inovações tecnológicas 3. Jogos eletrônicos - Desenvolvimento 4. Jogos eletrônicos - Programas de computador 5. Tecnologia (Ensino fundamental) I. Título.

21-92997 CDD-372

Ilustrações

Marcas Registradas

Rod Celso Garcia Cid, Km 380, s/n - Campus Universitário - Londrina - PR
Fone e WhatsApp: 43 9 8843-1202

hackidsedu
hackidsedu
hackids.com.br

criado com ♥ pela Hackids®

UMA MENSAGEM PARA VOCÊ

Seja bem-vindo ao mundo da Educação 4.0! Este livro apresenta o universo da programação e da criação por meio de uma série de desafios utilizando a micro:bit, nosso gadget preferido.

Neste livro, você você terá acesso a um conjunto de 13 projetos, nos quais você irá estudar e explorar programação por meio de blocos de uma forma prática e envolvente, construindo games totalmente funcionais.

Nossa principal plataforma de programação para o aprendizado dos conceitos fundamentais da ciência da computação será o MakeCode da Microsoft, que é o primeiro passo para explorar recursos mais avançados com linguagens, como o Python.

Tecnologia não é mágica, porém aprender a criar tecnologia é realmente mágico, e qualquer um pode aprender, especialmente você. Este livro irá ajudá-lo a entender como muitos dos jogos são criados, modificando-os à sua maneira e, assim, criar sua própria magia.

Seja curioso! Divirta-se lendo e construindo!

Wagner, Hackids Editora

Olá, meu nome é Wagner Rodrigues,Ms., fundador da Hackids, Parceiro da Fundação micro:bit para o Desafio Global ***do your :bit***, Educador micro:bit Champion 2021, mestre em Desenvolvimento de Tecnologia, educador certificado Lego Education, envolvido com a cultura maker, ciência e tecnologia, em conjunto com a BBC micro:bit procura produzir e compartilhar projetos que possam ajudar a resolver os problemas e situações da atualidade.

Tenho plena convicção de que a ciência da computação é uma habilidade que pode ser aprendida e explorada por qualquer pessoa, independente de seu gênero, idade, riqueza ou local de habitação. Assim, é nosso objetivo ajudar tantos quantos forem possíveis a conhecer e aplicar a programação como um meio de criatividade e progressão pessoal.

A QUEM ESTE LIVRO SE DESTINA

Este é um livro para você que deseja explorar o universo da ciência da computação. Ele pode ser utilizado como livro paradidático para alunos do ensino fundamental e para aqueles eternos alunos que desejam aprender algo novo por conta própria.

Não é necessário ter experiência em programação, pois todos os projetos possuem versões básicas e versões mais avançadas para que você possa expandir seu aprendizado de programação. A beleza de programar está no fato de podermos criar, seja do zero, seja remixando algo existente. Se você tem uma ideia, você pode sentar na frente de seu computador, sacar o teclado e, após algum tempo, ter algo funcional, um novo projeto para se divertir e compartilhar. Não tenha medo de experimentar, modificar, adicionar, remover elementos, pois os erros trarão muita aprendizagem. Faça suas anotações, explore e separe um tempo para refletir sobre as soluções e sobre os erros.

RECURSOS

Nosso intuito foi elaborar um livro prático, em alguns momentos com instruções detalhadas de programação, em outros, desafios para que você possa resolver e aprender os conceitos de programação. Nos vários exemplos apresentados ao longo do livro, exploramos vários conceitos, como algoritmo, repetições, pilha, fila, variáveis, criação de funções e muito outros elementos tratados na **Base Nacional Comum Curricular** (BNCC).

As seções **Para ir além** e **Indo fundo** presentes nos capítulos foram concebidas para desafiar suas habilidades de programação e todas elas possuem cenários com uma sugestão de solução. Antes de você olhar o código pronto, sugiro que você tente resolver e criar suas próprias soluções. Esta é uma das belezas da programação: podemos codificar soluções diferentes para um mesmo desafio.

Acesse ***https://tiny.one/programandojogos*** para ter acesso aos recursos utilizados neste livro. Nesta página, você encontrará erratas e atualizações. Nós fizemos o melhor para garantir que o livro contenha informações precisas, no entanto, errar é humano. Assim, acesse o link para obter as atualizações mais recentes, caso sintamos tal necessidade.

PALAVRA CHAVE

ORGANIZAÇÃO DOS CAPÍTULOS

O primeiro capítulo apresenta a micro:bit e o Makecode. Neste capítulo, são descritos, de forma rápida e objetiva, os elementos de partida para o uso dos recursos. Do capítulo 2 ao capítulo 14, são abordados os projetos, finalizando com uma breve apresentação sobre como compartilhar seus projetos e, por fim, uma breve descrição do eixo **Pensamento Computacional do Currículo de Tecnologia e Computação** (CIEB) e uma matriz de correlação do conteúdo deste livro em relação ao presente currículo.

Capítulo 1 - Desvendando a micro:bit apresenta as características da micro:bit e do ambiente de programação Makecode, bem como o processo de conexão e transferência da programação entre o Makecode e a micro:bit.

Capítulo 2 - Matemática em todo canto: é hora de explorar a tabuada. Neste jogo, você irá explorar a utilização de variáveis para registro de dados, como números aleatórios. Também dará seu primeiro passo na tomada de decisão e controle de fluxo dos programas.

Capítulo 3 - Sensações - Quente ou frio: neste divertido projeto, que não é propriamente um jogo, você aprenderá a utilizar os sensores internos da micro:bit enquanto se aprofunda na construção de operadores lógicos para controle de fluxo dos programas.

Capítulo 4 - ASTRO:BIT - My little pet: você irá mergulhar no mundo dos emojis e na construção de animações, explorando seu uso na construção de um pet virtual. Tudo isso enquanto aprende um pouco sobre o acelerômetro e outros sensores e entradas da micro:bit. Cuidado para seu pet não ficar com frio ou calor ;)

Capítulo 5 - Dados digitais: construir um dado com a micro:bit é um projeto clássico. Desta vez, você irá explorar recursos matemáticos como aleatoriedade, arredondamento, escolhendo entre projetar algo para um ou dois dados.

Capítulo 6 - Hot wire: você já deve ter visto ou brincado com o jogo labirinto elétrico. Pois bem, é hora de tirar o arame da gaveta e construir este clássico jogo, tudo isto enquanto explora o uso de múltiplas variáveis para registro de falhas e recompensas. E não podemos deixar de lado que serão utilizados os blocos de entrada para controle de ações que ocorrem nos pinos da micro:bit.

Capítulo 7 - Contos fantásticos: neste jogo, inspirado no famoso Rory's Cube, novamente é explorado o recurso de múltiplas variáveis e aleatoriedade, tudo isso para construir uma das ferramentas de narração de histórias mais valiosas para escritores de todas as idades.

Capítulo 8 - Escape vírus. É sempre legal jogar um arcade, melhor ainda é construir o seu. Você será inspirado a construir um projeto inspirado no clássico Space Invaders. Você está sob uma intensa chuva virótica, se der bobeira será contaminado. Ao longo do projeto, repetições, criação de variáveis, criação de sprites, utilização dos blocos especialmente construídos para games são alguns dos itens apresentados neste capítulo.

Capítulo 9 - Pedra, papel, tesoura. Este é um jogo clássico para quem está entrando no mundo da micro:bit, mas aqui gostamos de apimentar os projetos. Será necessário aprofundar na elaboração de rotinas de tomada de decisão, expressões lógicas mais complexas, uso de elementos de aparência, registro de pontuação e a cereja do bolo, a comunicação entre dois micro:bit's. Isso mesmo, este é um projeto que pode ser jogado em duplas, cada um com sua micro:bit.

Capítulo 10 - Pixel memory: no projeto Hot Wire, sua coordenação motora foi colocada a prova, agora queremos ver como anda sua memória e coordenação motora juntas. Você irá explorar a construção de variáveis booleanas, construções melódicas, construção de múltiplas funções para decompor problemas, novos tipos de repetições, manipulação individual dos LEDs do display, aleatoriedade associada às coordenadas x e y. Além de algo incrível da ciência da computação, a recursividade.

Capítulo 11 - Campo minado: você tem apenas dez segundos... isso mesmo, apenas 10 segundos para desarmar a bomba. Aqui será explorada a comunicação serial da micro:bit para aprender e entender a coleta de dados de sensores internos da micro:bit. O acelerômetro será seu joystick, suas condicionais ficarão mais complexas, aprenderá a usar operadores AND (E) e usar o conceitos de vidas para seu sprite (jogador). Mas não para por aí, ainda terá desafios para construção de animações e o uso de recursos de brilho para melhorar a aparência e jogabilidade.

Capítulo 12 - Pandemia: Missão de resgate. Se você estava preocupado ao construir o Escape Vírus, agora temos variações dos vírus, enquanto em Escape Vírus tínhamos apenas um vírus, agora eles sofrerão mutações, variações letais e não letais. Você irá explorar repetições de forma muito mais aprofundada, sistema de pontuação e vidas baseados em regras, movimento através das coordenadas (x e y), criação de funções, padrões melódicos diferentes para as ações do jogo e estabelecimento de níveis de dificuldade.

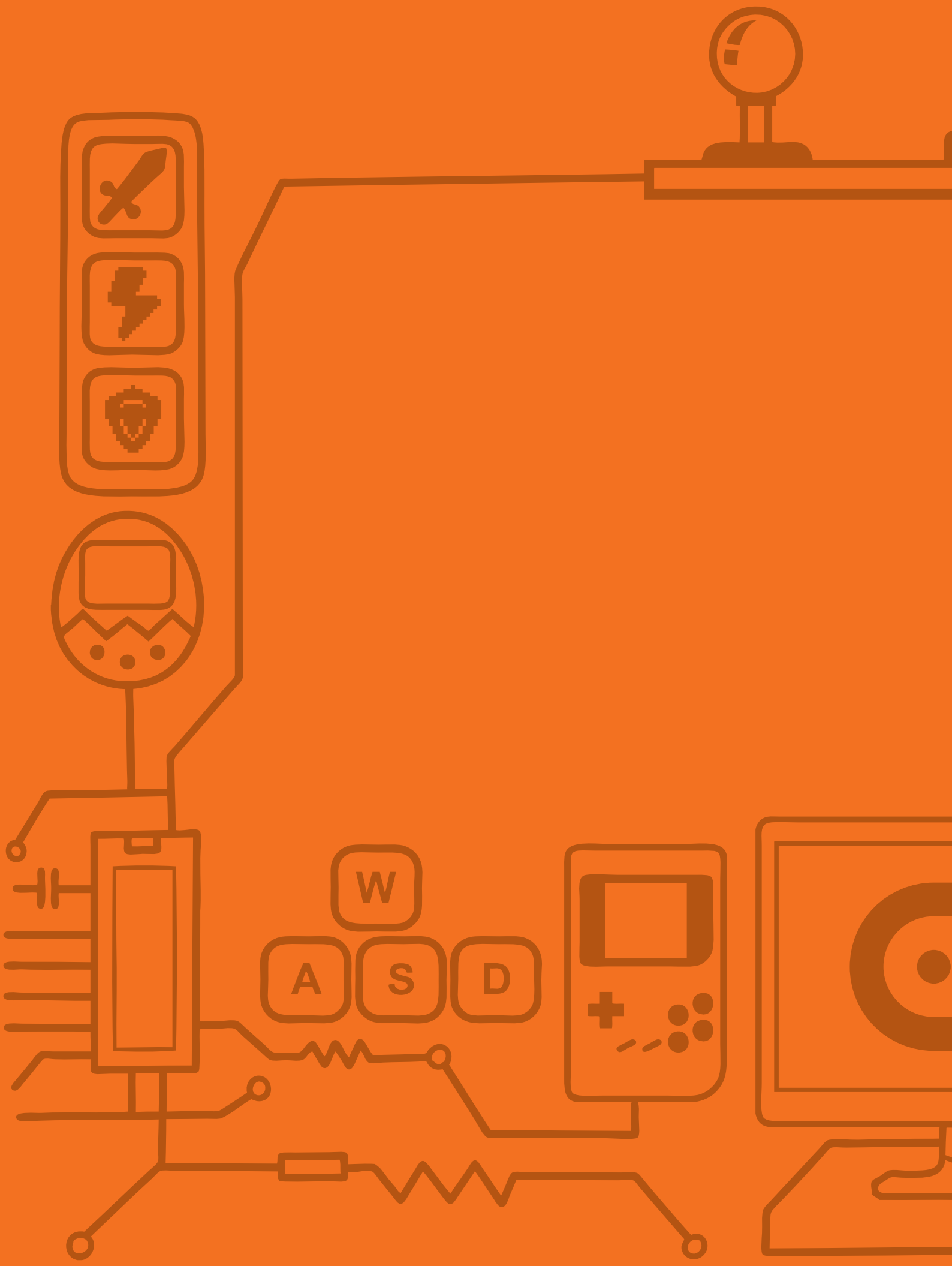
Capítulo 13 - bit:repeat - Jogo da Imitação. Simon diz... inspirado no jogo Genius, é hora de construir um clássico dos jogos, aqui chamado de bit:repeat. Neste projeto, você irá explorar o uso de listas e filas, processamento de strings, criação de variáveis do tipo string e, novamente, o acelerômetro será seu joystick. Iniciará com uma versão com apenas quatro elementos a serem repetidos até chegar ao terceiro projeto, em que não terá limite na quantidade de elementos a serem repetidos e ainda com controle de velocidade.

Capítulo 14 - magic:bit - Jogo de Adivinhação. Vamos construir um pequeno jogo para que você tenha uma ideia de como algoritmos diferentes para o mesmo problema podem ter desempenhos completamente diferentes. Não é mágica, é matemática!

Capítulo 15 - Compartilhar. Neste apêndice, nosso objetivo é apresentar uma forma de compartilhar seus projetos com seus amigos e comunidade.

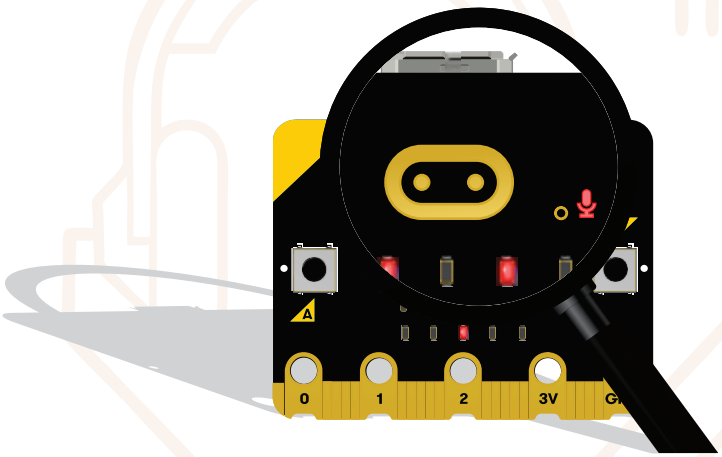
Apêndice - Pensamento Computacional. Neste apêndice nosso objetivo é apresentar o Currículo de Tecnologia e Computação (CIEB), sua correlação com a Base Nacional Comum Curricular e como o conteúdo deste livro está conectado a esta matriz.

SUMÁRIO


W
A
S
D

DESVENDANDO A MICRO:BIT 01

Neste capítulo introdutório, aprenderemos sobre as características e funcionalidades básicas da micro:bit e como programá-lo utilizando o Makecode.



MICRO:BIT

POR DENTRO DO HARDWARE

É muito comum ouvir questionamentos sobre como conceituar o que é a micro:bit e para que ela serve.

Assim, é importante garantir o entendimento sobre a micro:bit. Ele é um "cérebro" que pode sentir, controlar e reagir ao mundo real de várias maneiras.

As criações não precisam ser complexas, fazer parte do processo é tão divertido quanto a parte de codificação. Alguns projetos terão apenas um sensor e uma saída (como um contador de tráfego), outros terão programas mais complexos (como um robô de controle remoto). Sua beleza está em sua simplicidade e versatilidade.

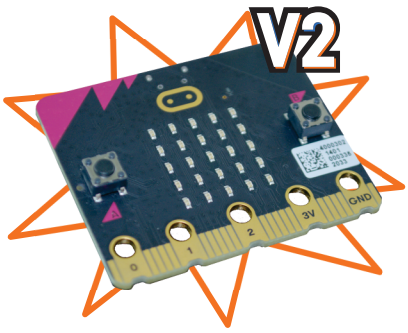
A chave para desbloquear todo o poder da micro:bit é compreender os blocos de programação ou o código que a micro:bit irá utilizar, além de aprender como combinar os blocos para resolver problemas de forma criativa. É importante ter em mente que há muito a ser aprendido. Por meio de experimentos, brincadeiras e desafios, nós progredimos muito mais rápido.

A BBC micro:bit é uma placa compacta baseada no microcontrolador Nordic **nRF51822** (v1) ou **nRF52833** (v2), ou seja, é um pequeno computador com LED´s, botões e sensores. É um gadget incrível.

Possui vários sensores embutidos: temperatura, luz, campo magnético em 3 eixos e aceleração em 3 eixos, possui Bluetooth LE, alguns botões de ação e uma matriz de LED 5x5.

Além disso tudo, possui 3 entradas/saídas analógicas/digitais fáceis de usar e muito mais entradas/saídas acessíveis com extensões de hardware. Possui comunicação Bluetooth e também é possível usá-lo conectado a um computador através de um cabo microUSB para USB.

No Reino Unido, elas foram distribuídas para crianças em idade escolar na expectativa de inspirar uma nova geração de codificadores. Muitas outras placas inspiradas na BBC micro:bit estão começando aparecer, o que leva ao aumento de sua popularidade!



A micro: bit possui:

uma matriz 5 x 5 de LEDs (diodos emissores de luz);
dois botões (A e B);
um acelerômetro (muito útil para saber em que direção está);
um magnetômetro (yes, isso é como uma bússola);
um sensor de temperatura;
microfone (**v2**);
alto-falante (**v2**);
Bluetooth (para conversar com outros dispositivos);
pinos para conectar-se a outros dispositivos/recursos externos, como telas, motores, sensores, botões, robôs e muito mais!

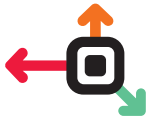
BOTÕES

Botões A e B, são botões programáveis que podem executar comando ou ações ao serem programados. Quando você pressionar um dos botões, ele completa um circuito elétrico. Com a micro:bit, podemos detectar o acionamento de qualquer um deles separadamente ou em conjunto, o que nos permite construir um programa para agir sobre isto.

SUSPENSÃO

Pressionar o botão de reset na nova micro:bit redefinirá a micro: bit e executará seu programa novamente desde o início. Se você mantê-lo pressionado, o LED vermelho de energia apagará. Quando o LED de energia apagar, solte o botão e sua micro:bit estará no modo de espera para economia de energia. Use isto para fazer suas baterias durarem mais. Pressione o botão novamente para ativar sua micro:bit.

ACELERÔMETRO



Ele pode detectar o grau de inclinação na micro:bit e medir as mudanças na velocidade da micro:bit. Converte informações analógicas para o formato digital que podem ser utilizadas em programas para a micro:bit. O dispositivo também pode detectar alguns padrões de ações, como por exemplo, agitar, inclinar, entre outros.

LED´s

Os LED´s do display podem ser ligados ou desligados para exibir imagens ou pictogramas. O Makecode para micro:bit fornece diferentes blocos para uso. Você pode ativar ou desativar os LEDs e criar suas próprias imagens.

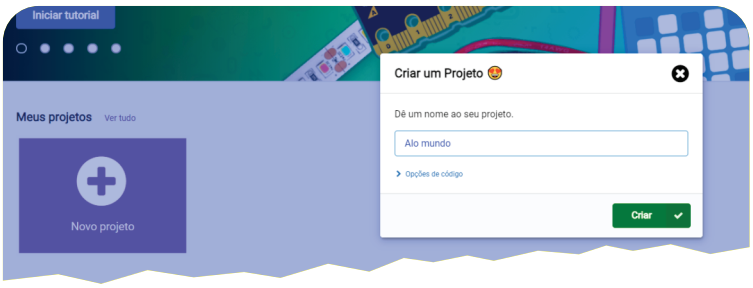
TOQUE

Os pinos grandes (0,1,2) e o logotipo (V2) na micro:bit podem ser configurados para saber quando estão sendo tocados ou pressionados.
A micro:bit V2 também pode usar um modo de toque diferente, que você pode configurar em seu programa.
O **toque resistivo** funciona detectando uma mudança na resistência quando um sinal elétrico passa por um material condutor como parte de um circuito.
O **toque capacitivo** funciona detectando mudanças no campo elétrico de um capacitor usando um dedo como condutor. Ele será acionado quando seu dedo tocar o pino ou se aproximar dele. O toque capacitivo não exige que você faça uma conexão de aterramento como parte de um circuito, portanto, você pode apenas tocar a micro:bit com um dedo.

A micro:bit pode parecer simples a princípio, mas, com sua criatividade e imaginação, você pode criar e codificar muitas coisas. Você poderá construir um relógio digital, usar os botões para ativar um cronômetro, criar um game, construir uma bússola digital ou um acelerômetro digital para colocar na sua bike ou em seu patinete... são inúmeras as possibilidades.

Aqui deu certo, consegui acesso... abri meu navegador, digitei o endereço... assim que abriu, cliquei em **Novo projeto**...

Vamos dar um nome ao projeto, por exemplo, **Alô mundo**...



Que tal explorar o ambiente do Makecode!!! Acesse as categorias, observe os blocos disponíveis... O MakeCode tem um simulador embutido, significa que você não precisa realmente de uma micro:bit para aprender como usá-la.

CONSTRUIR

Para apresentar uma mensagem de texto, ou seja, uma string, você irá precisar de um bloco que represente esta condição. Explore as categorias em busca de um bloco que possua estas características.

Você deve ter notado que todas as vezes que criamos um novo projeto, dois blocos já são adicionados automaticamente, os blocos **no iniciar** e **sempre**.

O bloco **no iniciar** irá executar tudo que estiver dentro dele uma vez, ao iniciar a micro:bit. Já o bloco **sempre** é onde você irá colocar os blocos que serão executados constantemente, logo após a execução do bloco **no iniciar**. Isso quer dizer que o bloco **sempre** fica em um loop, que será encerrado quando a placa for desligada.

Assim que você finalizar sua programação, você pode testar e validar o resultado usando o simulador que fica do lado esquerdo da tela

Agora que você testou no simulador virtual, você pode enviar sua programação para a micro:bit; se ainda não deu um nome ao seu projeto, é o momento de fazer. Em seguida, é hora de enviar a programação clicando no botão **Baixar**.

Um passo importante é sincronizar o Makecode com a micro:bit, de forma que, ao clicar no botão Baixar, sua programação seja enviada diretamente a ela.

PARA IR ALÉM
DO BLOCO AO PYTHON

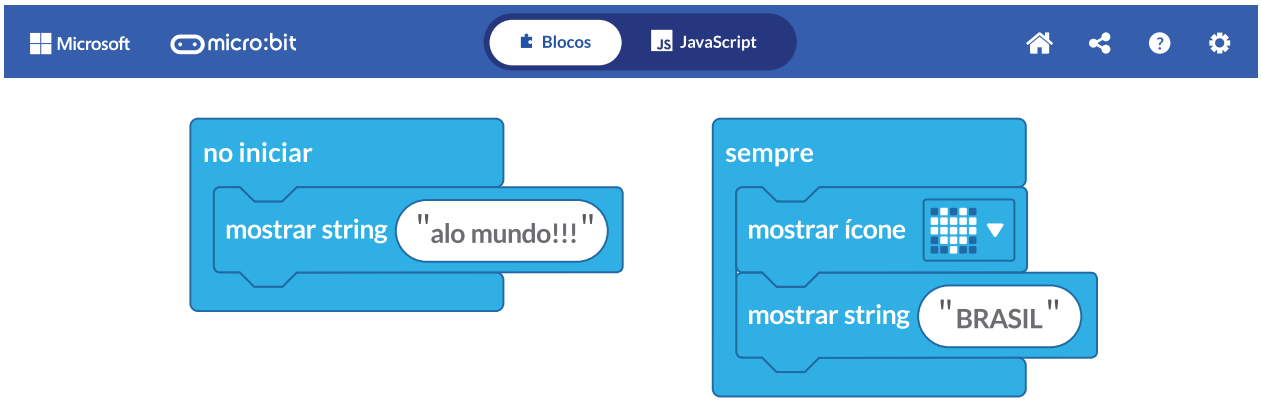
Um elemento muito importante a ser explorado no Makecode é sua capacidade de permitir a programação de múltiplas maneiras. Você pode programar utilizando blocos ao mesmo tempo em que vai se familiarizando com a programação textual, seja utilizando JavaScript, seja utilizando o Python.

Python é uma linguagem simples, porém muito expressiva; ela é amplamente usada, desde a programação de operações de script simples até complexas aplicações na Web.

Aprender Python com MakeCode permite que você comece com programas simples enquanto alterna entre blocos e até mesmo JavaScript, se desejar. Posteriormente, você pode avançar para conceitos mais complexos, como matrizes e classes. À medida que você desenvolve proficiência no editor Python, pode escrever um código muito complexo, desta forma, você estará no caminho certo para se tornar um verdadeiro codificador. :)

Para iniciantes, familiarizar-se com programação textual pode ser assustador. A programação em bloco permite o encadeamento lógico das sequências de maneira visual, e uma das maneiras mais fáceis de mergulhar na codificação com Python é converter um design em programação com blocos para Python (textual). Iremos utilizar o exemplo a seguir, em que temos a programação para apresentação de uma mensagem no display do microbit.

Não deixe de construir esta programação para você validar o experimento.



No menu suspenso, você encontrará diferentes opções para visualizar seu código. Além da familiar visualização em Blocos, você também pode escolher uma visualização em Texto, que representa seu código na linguagem Python, como pode ser observado abaixo.

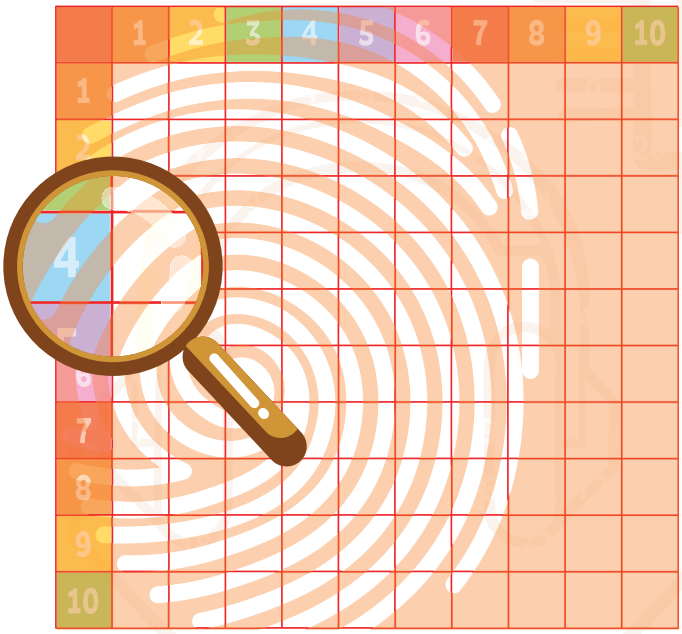

```
basic.show_string("alo mundo!!!")

def on_forever():
    basic.show_icon(IconNames.HEART)
    basic.show_string("BRASIL")
basic.forever(on_forever)
```

O mais legal é que mover seu código em blocos para Python não é uma via de mão única, ou seja, não quer dizer que ao mudar não poderá mais visualizar em blocos; pelo contrário, se você mantiver a compatibilidade da linguagem, você pode voltar a visualizar em bloco e vice-versa.

MATEMÁTICA EM TODO CANTO 02

A Matemática está em toda parte, então é hora de explorar. Neste projeto, por meio de um jogo totalmente dinâmico, você irá explorar a tabuada.
Você programará a micro:bit para construir problemas simples de matemática, além de apresentar as respostas, quando necessário, para te ajudar a praticar a tabuada.



O X DA QUESTÃO

A HISTÓRIA DOS DADOS

Ao aprender a tabuada, todos nós encontramos várias barreiras, e o objetivo neste projeto é tornar este aprendizado agradável e divertido. Uma solução é construir e transformar o processo de aprendizagem em um game. Neste sentido, um jogo muito comum é o uso de dois dados, porém você pode fazer o seu próprio jogo muito mais tecnológico.

Os dados e seus precursores são os instrumentos de jogo mais antigos conhecidos pelo homem. Sófocles relatou que os dados foram inventados pelo lendário grego Palamedes durante o cerco de Tróia, enquanto Heródoto afirmou que eles foram inventados pelos lídios nos dias do rei Atys.

Ambas as "invenções" foram desacreditadas por numerosos achados arqueológicos que demonstram que os dados foram usados em muitas sociedades anteriores.

Os precursores dos dados eram dispositivos mágicos que os povos primitivos usavam para lançar a sorte para adivinhar o futuro.

Os prováveis precursores imediatos dos dados foram objetos feitos de ossos do tornozelo de ovelhas, búfalos ou outros animais, às vezes com marcações nas quatro faces. Esses objetos ainda são usados em algumas partes do mundo.

Nos tempos gregos e romanos, a maioria dos dados eram feitos de osso e marfim; outros eram de bronze, ágata, cristal de rocha, ônix, azeviche, alabastro, mármore, âmbar, porcelana e outros materiais.

Dados cúbicos com marcações praticamente equivalentes às dos dados modernos foram encontrados em escavações chinesas de 600 AC e em tumbas egípcias que datam de 2000 AC.

Os primeiros registros escritos de dados são encontrados no antigo épico sânscrito, o Mahabharata, composto na Índia há mais de 2.000 anos. Os dados piramidais (com quatro lados) são tão antigos quanto os cúbicos; esses dados foram encontrados com o chamado **Jogo Real de Ur**, um dos jogos de tabuleiro completos mais antigos já descobertos, que remonta à Suméria no terceiro milênio AC.

Foi só no século 16 que os jogos de dados foram submetidos à análise matemática - pelos italianos Girolamo Cardano e Galileo, entre outros - e os conceitos de **aleatoriedade** e **probabilidade** foram concebidos. Até então, a atitude predominante era que os dados e objetos semelhantes caíam da maneira que caíam por causa da ação indireta de deuses ou forças sobrenaturais.

fonte: britannica.com

VARIÁVEIS

Programas de computador processam informações. Algumas das informações que são inseridas, armazenadas e usadas em um programa de computador têm um valor constante, o que significa que elas não são alteradas durante o curso do programa.

Um exemplo de constante em matemática é **PI** (π) porque π tem um valor que nunca muda.

Na maioria das linguagens de programação, uma variável é uma "caixa", um contêiner que contém informações que você pode acessar. Você pode abrir a caixa para descobrir o que está dentro ou até colocar outra coisa na caixa. De certa forma, a memória humana e a memória do computador são semelhantes. Confiamos em nossa memória para lembrar de todos os tipos de coisas.

Por exemplo, enquanto estamos assistindo a um jogo de basquete, iremos lembrar a pontuação atual. Agora, na linguagem do computador, essa é realmente uma variável que contém um número como 43 e muda ao longo do tempo à medida que mais cestas são feitas.

Meu sabor favorito de sorvete? Essa é outra variável que possui um valor como flocos.

As variáveis podem conter números, frações e frases.

Afinal, o que é uma variável?

Quando estamos programando, uma variável tem um nome e contém um valor . Pense em nossa analogia anterior de uma variável como sendo uma caixa. Se você rotulou a caixa como Brinquedos e colocou um cubo mágico dentro dela, em termos de programação, Brinquedos é o nome da variável e cubo mágico é o valor.

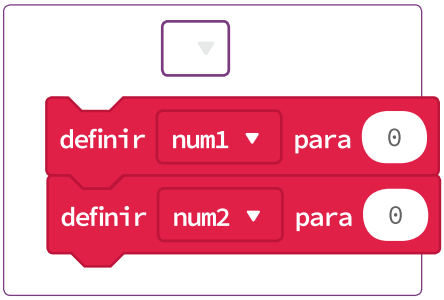
Vejamos três tipos de dados diferentes que podem ser armazenados - Números (numbers), Strings e Lógicos (booleans):

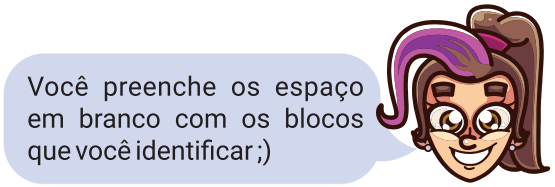
Variáveis do tipo **string** são variáveis usadas para representar texto. As cadeias contêm grupos de caracteres, como uma palavra ou uma frase. Uma maneira fácil de pensar sobre sequências de caracteres na vida real é pensar em como seu cérebro armazena informações. Pense em uma pessoa como uma variável. Um nome é simplesmente um valor que seu cérebro usa para identificar essa pessoa.

Variável **numérica** é um dos tipos de dados mais simples. Podemos armazenar diferentes tipos de números, números inteiros como 5 ou -3. Para armazenar valores fracionários, precisamos usar o formato decimal como 3.14 (usamos . (ponto) ao invés de , (virgula))

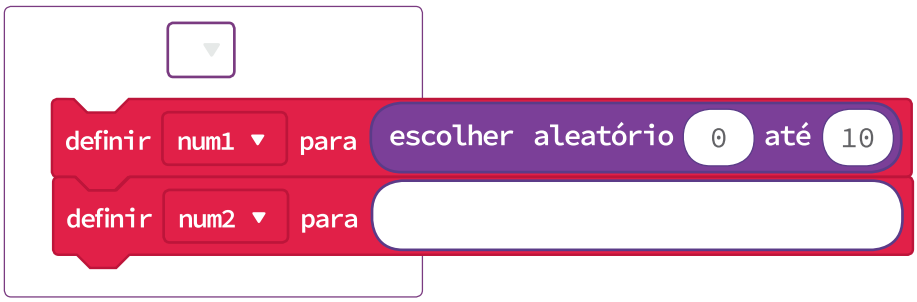
Uma variável **lógica** ou **booleano** contém o valor verdadeiro ou falso. Booleanos são valores verdadeiros - eles são um tipo de dados que pode representar um dos dois estados: verdadeiro ou falso. Os booleanos respondem a perguntas de sim ou não e as funções booleanas são usadas para executar testes. Em um videogame, você pode testar se um jogador entrou em um obstáculo ou se sua saúde é igual a cinco.

4 Próximo passo, agora você precisa localizar um bloco que tenha a funcionalidade de executar uma ação ao pressionar o botão A. Pesquise dentro da categoria **Input** (entrada) um bloco que tenha estas características. Quando você identificar este bloco, arraste para a área de programação e adicione dentro dele blocos definir **num1** e **num2**.

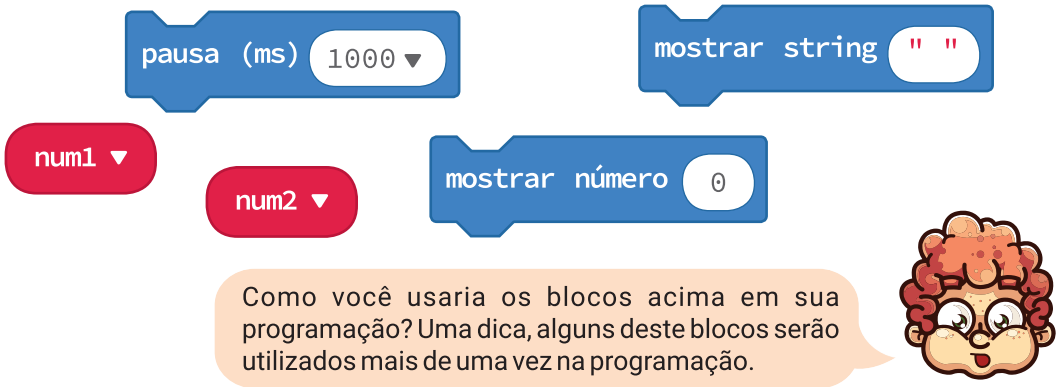




Nosso programa deverá definir um número aleatório entre 0 e 10 para as variáveis **num1** e **num2**. Em qual categoria você acredita que este bloco está localizado?



5 Com a programação feita até o momento, ao pressionar o botão A, as variáveis **num1** e **num2**, serão definidas com números aleatórios de 0 a 10 quando o botão A for pressionado.
Agora, você deve concluir a programação adicionando blocos que mostrem os valores das variáveis na matriz de LEDs, além de algumas mensagens quando o botão A for pressionado para que o usuário possa ver a pergunta.



SENSAÇÕES QUENTE OU FRIO 03

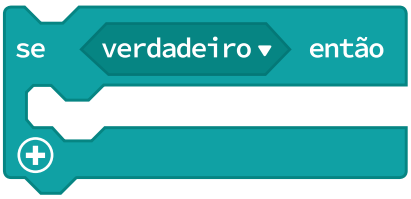
Neste capítulo, iremos iniciar a exploração dos recursos internos da micro:bit. O objetivo é criar um sensor de temperatura e para isto você aprenderá novos blocos no MakeCode.

2 Você já adicionou um conjunto de blocos que mostra o valor da temperatura continuamente, agora precisamos criar uma regra lógica, algo que possa ser representado por meio de programação o seguinte algoritmo:

```
se temperatura < 18
    mostre mensagem ''muito frio!!!''
se temperatura > 24
    mostre mensagem ''muito quente!!!''
senão
    mostre mensagem ''na medida!!!''
```

Na categoria Lógica, você encontrará vários blocos que nos ajudam a construir as regras acima.

Como você usaria os blocos desta categoria para resolver o final deste desafio?

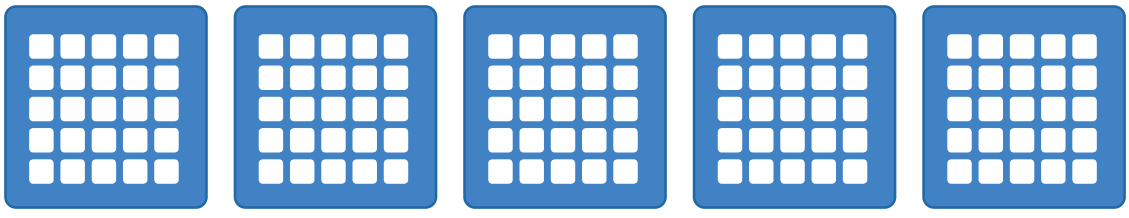


Operadores lógicos
- = **IGUAL a**
- ≠ **NÃO é igual a**
- < **MENOR que**
- ≤ **MENOR ou IGUAL a**
- > **MAIOR que**
- ≥ **MAIOR ou IGUAL a**

E então, conseguiu? Muito bem! Compartilhe com seus amigos os seus resultados.

E se você utilizar o botão A para mostrar a temperatura e o botão B para mostrar o nível de luz?

Que tal você medir a temperatura, mostrando um gráfico na matriz de LEDs utilizando cinco imagens para muito frio, frio, na media, quente e muito quente? Seja curioso e divirta-se!!!



muito frio! **frio!** **na medida!** **quente!** **muito quente!**

Estamos indo para nosso quarto capítulo, vamos dar uma olhada em tudo que exploramos nestes capítulos iniciais

Algumas coisas que você explorou e aprendeu:

- Conhecer as partes que compõem a micro:bit;
- Criar programas básicos usando MakeCode;
- Elementos básicos de programação textual (python);
- Dar nomes e salvar seus programas;
- Fazer upload de seus programas para sua micro:bit;
- Explorar vários blocos, de diversas categorias, em seus programas;
- Utilizar blocos extras, além dos desafios básicos;
- Utilizar blocos da categoria Matemática;
- Explorar em seus programas as Entradas (Input) para diversificar as ações;
- Criar e testar variáveis em soluções de problemas;
- Utilizar números aleatórios e combiná-los para criar uma útil aplicação;
- Utilizar lógica computacional para tomada de decisão em uma aplicação;
- Explorar o uso de entradas variadas e diferentes formas de saída.

É hora de continuar explorando novas possibilidades utilizando os blocos de programação e sensores para criar novas soluções ou melhorar as que você já criou.

Que tal melhorar o projeto da tabuada do capítulo 2 utilizando outra micro:bit para manter a pontuação de dois jogadores. Você poderá transferir a pontuação entre as micro:bits utilizando os blocos da categoria **Rádio**.

Ou, estender as funcionalidades do projeto de temperatura. Fazer algo como: se estiver muito quente acionar algum dispositivo ao invés de apenas escrever uma mensagem que está quente demais. Será que podemos ligar um dispositivo externo com a micro:bit?

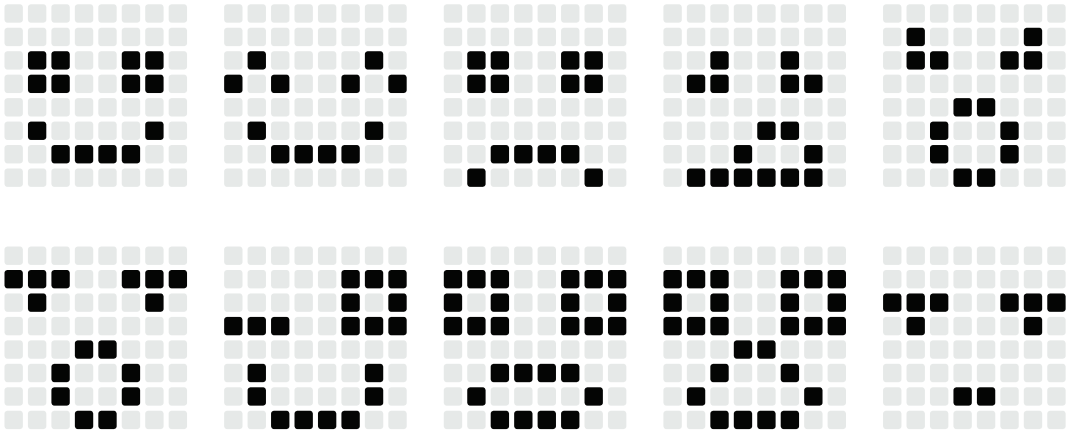
Pesquise o que é uma solenóide e como ela pode ser utilizada para ligar ou desligar máquinas.

Como funciona a fechadura elétrica
https://tiny.one/fechadura

ASTRO:BIT MY LITTLE PET 04

Neste capítulo, desbravaremos o uso das entradas da micro:bit para criar um pet digital, usando expressões de emojis que serão apresentadas na tela como saída. Você recebeu a tarefa de criar um animal de estimação digital que possa brincar e fazer companhia às pessoas enquanto elas permanecem no hospital.

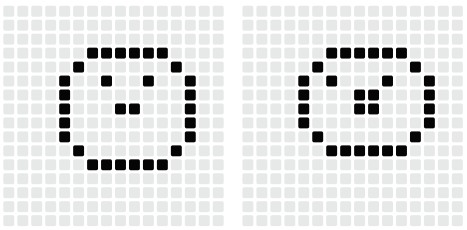
Confira as expressões que usam matriz 8×8, maiores que a 5×5 encontrada na micro:bit, em que muitos olhos expressivos têm 3 pixels de largura:



Isto nos levará a alguns desafios neste encontro.

Para ajudar na busca de soluções e inspirações, podemos voltar no tempo e observar os brinquedos com displays de baixa resolução, incluindo um dos meus favoritos: o Tamagotchi. O Tamagotchi original de 1996 tinha uma tela ou matriz de 32×16.

Observe na figura abaixo que a boca é simétrica e tem apenas 2 pontos de largura.



Uma grande vantagem da micro:bit é que você tem o poder de incorporá-lo da forma que desejar, ou seja, podemos usar o display da micro:bit como parte de um personagem.

Assim, diferentemente do Tamagotchi, onde a tela exibe o corpo do personagem e sua expressão facial, podemos usar a matriz de LEDs da micro:bit apenas para transmitir as expressões enquanto envolve o microcontrolador em uma forma personalizada que define o corpo do personagem. O estilo geométrico da micro:bit funciona muito bem para transformar a micro:bit em um personagem divertido.

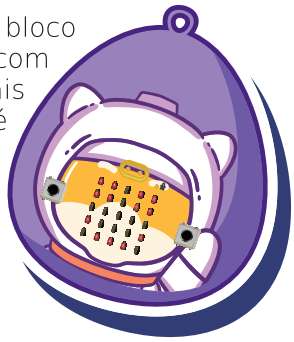
astro:bit^{++} (Hackids)

George Green's

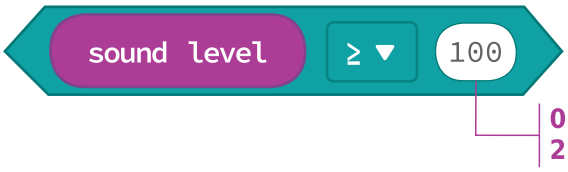
PARA IR ALÉM

Que tal tentar unir o bloco **pausa (ms)** com o bloco **escolher aleatório** para criar intervalos de tempos com valores variados? Seu pet virtual ficará muito mais imprevisível. Lembre-se de que o valor do bloco pausa é em milissegundo, então 1000 e 1050 é uma diferença muito pequena e pouco perceptível.

Alguns arranjos matemáticos poderão te ajudar a obter resultados interessantes.



A micro:bit v2 possui um microfone embutido, então você pode explorar este recurso para dar um susto em seu pet virtual e criar uma expressão para esta emoção. Na categoria **Input** (Entrada), há um bloco chamado **sound level**. Utilizar este bloco em uma condição **SE** pode trazer resultados interessantes a seu projeto. Experimente!!!



0 - silêncio
255 - som alto

Você conhece algum equipamento ou dispositivo que utiliza a detecção de um ruído ou som para ativar alguma ação?

DADOS DIGITAIS 05

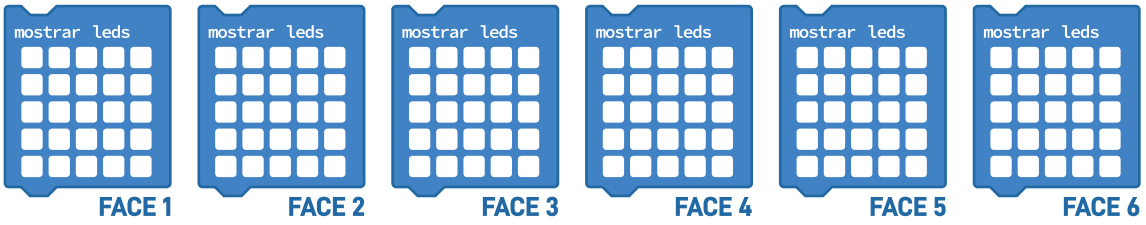
Neste capítulo, você irá construir um programa para apresentar um ou dois dados de 6 faces do display da micro:bit. Isso mesmo, separe o jogo de tabuleiro de que você mais gosta e chame seus amigos. Agora você poderá lançar incríveis dados digitais!

DESAFIO 1

Antes de você iniciar seu algoritmo, é hora de pensar como representar as faces de um e dois dados no display de LEDs. Utilize as imagens abaixo para esboçar como os dados serão apresentados.

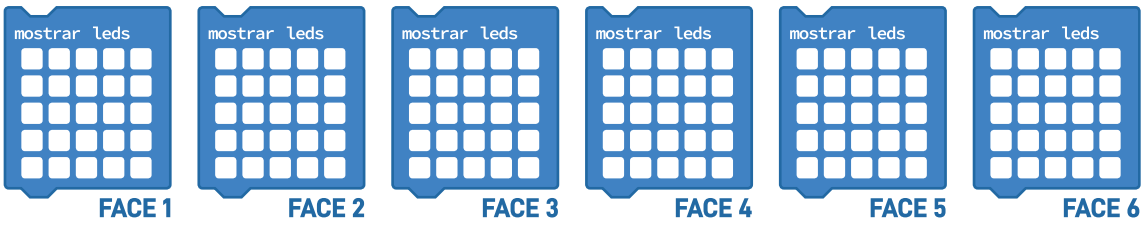
MODO: 1 DADO

DADO 1

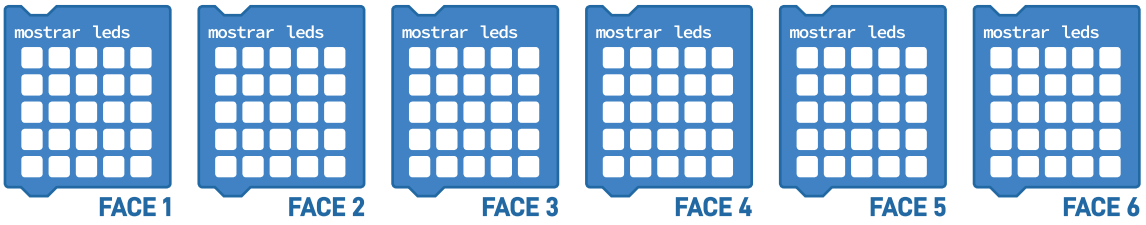


MODO: 2 DADOS

DADO 1

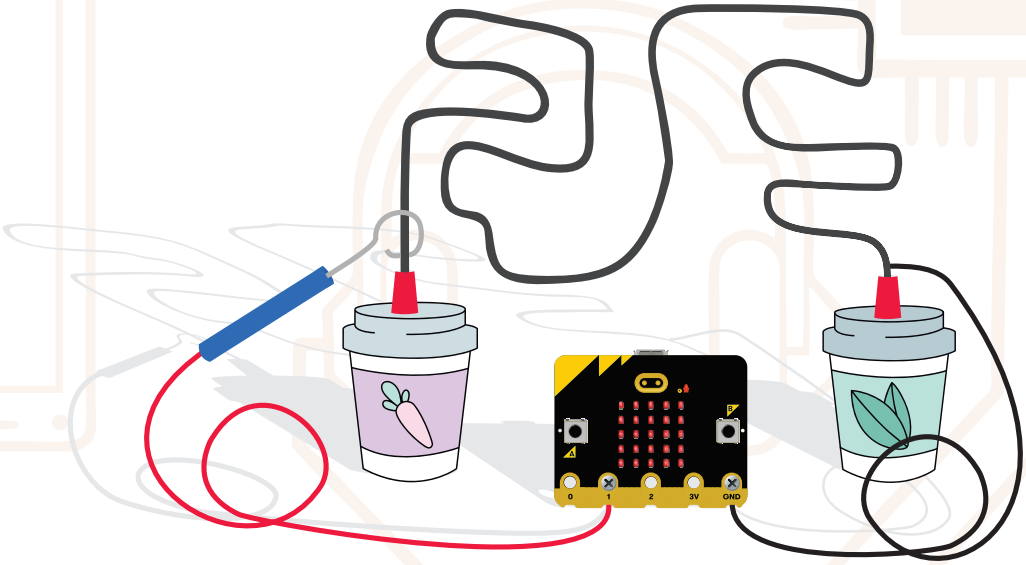


DADO 2



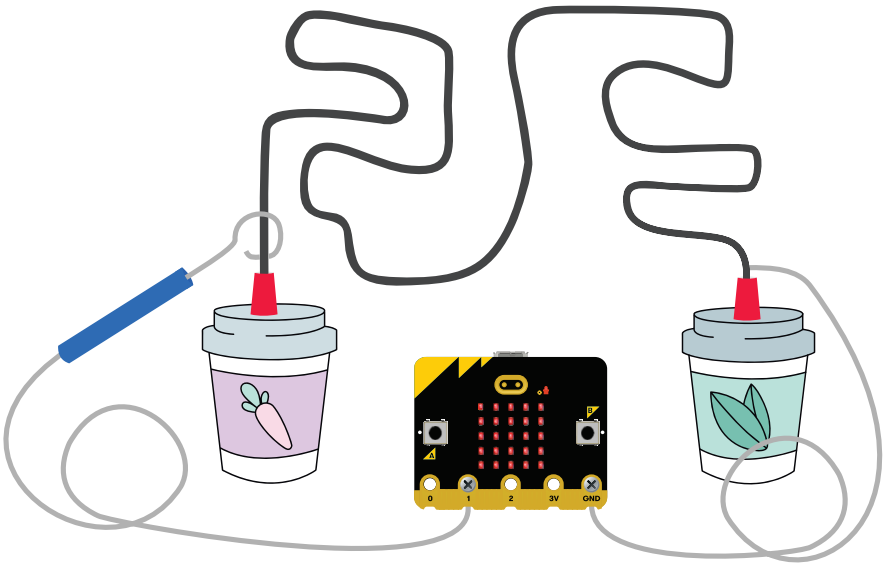
HOT WIRE 06

Neste capítulo, está reservado um daqueles projetos clássicos, você irá construir um jogo para testar suas habilidades motoras e se você tem mãos firmes. Você deve terminar a pista de obstáculos sem tocar no fio de aço dentro do limite de tempo. É melhor ter controle e não ficar ansioso enquanto joga. Jogue por sua própria conta e risco!



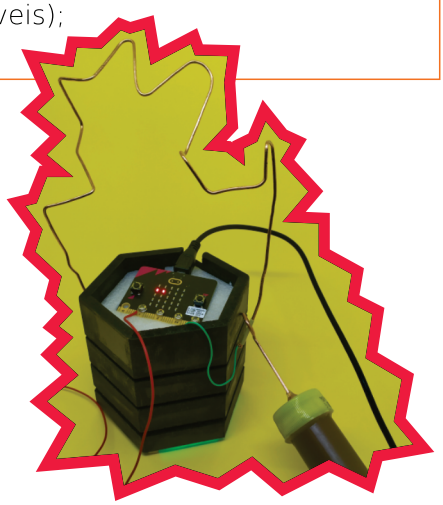
NERVOS DE AÇO

Cuidado, o fio está quente! Neste jogo de habilidade, você deve tentar guiar um anel de arame ao longo do fio sem tocá-lo. Se o fio for tocado, um sinal sonoro é ouvido e o jogo deve ser reiniciado.



Para a construção deste projeto, você precisará de alguns materiais e poderá montar conforme sua inspiração, pois não há uma regras para a montagem. A lista abaixo possui alguns materiais que podem ser úteis:

- micro:bit;
- garras de jacaré ou fios para conectar a micro:bit ao anel e ao percurso;
- aproximadamente 50 cm de fio metálico rígido (arame, fio de cobre, fio desencapado de cabo de rede...);
- cano PVC com 10cm a 12cm para a construção do cabo (opcional);
- 2 tampinhas de refrigerante PET (opcional);
- cola quente (opcional);
- fita isolante (opcional).
- parafusos e porca - 3mm (opcional);
- papelão (opcional);
- caixa para a base (por exemplo, copos descartáveis);
- alto-falante (opcional).



DESAFIO 2

Que tal adicionar alguns extras ao seu jogo? Uma das coisas mais legais em programação é a liberdade que temos em criar ou mudar as regras, adicionando ou eliminando elementos da programação. Agora que você já resolveu o primeiro desafio, você pode adicionar alguns extras.

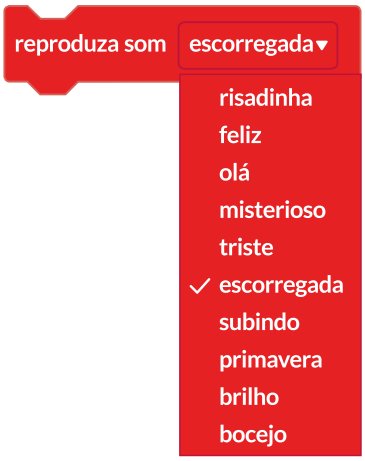
Nossa sugestão:

- ☑ reproduzir um som ao iniciar e ao pressionar o botão A;
- ☑ reproduzir um som, diferente do anterior, a cada falha.

Não deixe de realizar suas anotações sobre como você adicionaria estes recursos a seu projeto. Mantenha sempre documentado seu diário de bordo, pois estas informações ajudam a resolver novos desafios.

V1

V2



CONTOS FANTÁSTICOS 07

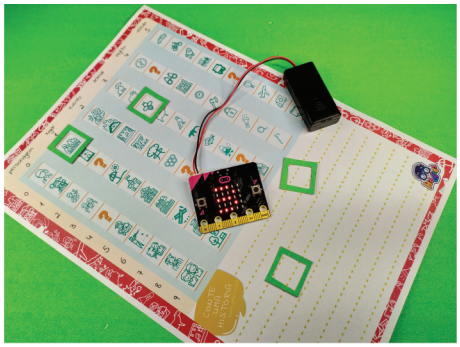
Novo capítulo e, desta vez, o projeto mistura elementos físicos e digitais. Você irá construir uma das ferramentas de narração de histórias mais valiosas para escritores de todas as idades. Elas inspiram os escritores a ter ideias interessantes para histórias . E, com uma pitada de aleatoriedade, nunca se sabe que combinação de ideias obterá quando você agitar sua micro:bit.

DESAFIO

Requisitos do protótipo

- ☑ Ao iniciar a micro:bit, exibir uma imagem ou animação que faça referência ao jogo;
- ☑ Sempre inicia com as variáveis zeradas;
- ☑ Quando pressionado o botão A, é definida aleatoriamente a coluna de categoria e o item da categoria (linha);
- ☑ Se a coluna já tiver sido escolhida, é necessário pressionar o botão A para escolha de nova alternativa;
- ☑ Quando pressionar o botão Botão A+B, zera as variáveis para começar nova história.

Para inspirar, ou se quiser remixar, servir como ponto de partida para seu projeto, tiramos algumas fotos e fizemos um vídeo do nosso **Conte uma história com micro:bit** que criamos aqui no HacKids Labs.



https://tiny.one/[redacted]

https://tiny.one/[redacted]

Era uma vez...

Em cada rodada, um jogador deve começar clicando no Botão A para selecionar a coluna de categoria e o elemento dela. A partir dos personagens, lugares, eventos, construa sua história usando as palavras representadas pelas imagens.

Será necessário criar algumas variáveis para controlar as categorias e elementos selecionados. Para a montagem do protótipo, além da micro:bit, você pode utilizar alguns materiais que você já tenha disponível, como cartolina e materiais para colorir.
Todos os ícones que usamos para representar os elementos foram coletados em sites da Internet, porém você também pode desenhar seus próprios elementos.

Dicas de três sites com ícones super legais para seus projetos! 😎😍

🔥🔥🔥 https://thenounproject.com/icons/
🔥🔥 https://www.freepik.com/
🔥 https://freeicons.io/

MERGULHANDO FUNDO

DEFININDO CATEGORIA E ITENS ALEATÓRIOS

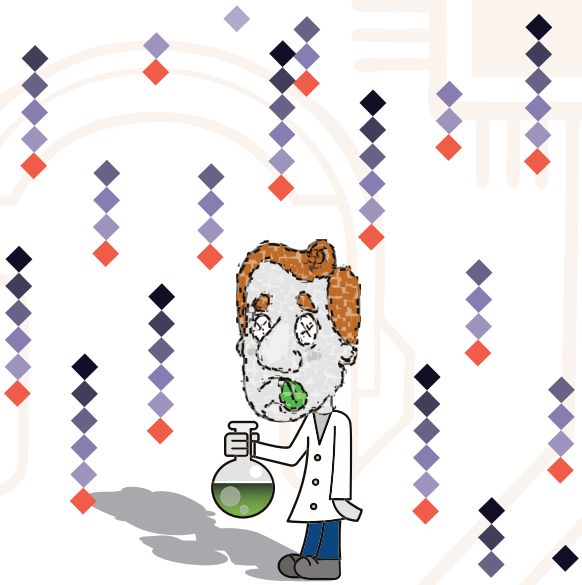
Kelly, nestes dois projetos, as colunas eram definidas sequencialmente. E se pudéssemos definir apenas 4 ou 5 colunas aleatórias para a construção de histórias?

Caio, é possível! Se criar variáveis, uma para cada coluna e mais alguns controles na programação, é algo possível sim. Fica como desafio para ir fundo na lógica de programação.

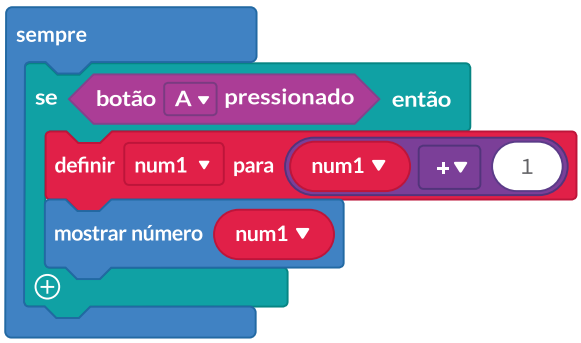
Vou começar a trabalhar no projeto!!!

ESCAPE VIRUS 08

Desta vez, a inspiração vem de algumas décadas atrás. Você irá construir um projeto inspirado no clássico arcade Space Invaders. Você irá criar um jogo totalmente funcional, que te levará a explorar e aprender mais à medida que vamos avançando.



O exemplo abaixo faz com que a micro:bit aguarde uma ação do usuário (pressionar Botão A) dentro de um loop **Sempre**.
Somente ao pressionar o Botão A é adicionado 1 à variável **num1**. Com algumas alterações, este código poderia ser o ponto de partida para um cronômetro.



Python

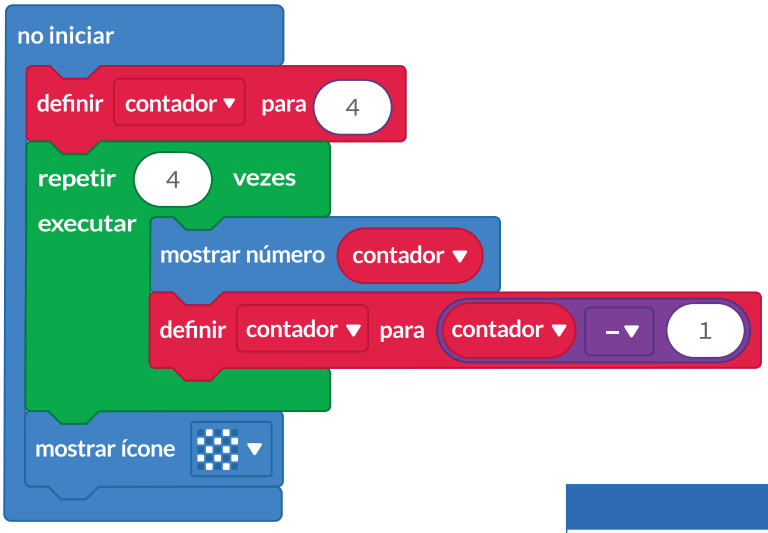
```
num1 = 0

def on_forever():
    global num1
    if input.button_is_pressed(Button.A):
        num1 = 0 + 1
        basic.show_number(num1)
basic.forever(on_forever)
```

Imagine que você precise fazer um contador regressivo para a largada de uma corrida, ou para medir o tempo que alguém terá disponível para apresentar uma resposta para um quiz. O bloco **repetir ... vezes** poderá controlar a apresentação no display do valor da variável contador.

No exemplo abaixo:

- ☑ ao iniciar, é definido o valor 4 para a variável contador;
- ☑ em seguida, entra no bloco repetir, em que será repetido 4 vezes o seguinte:
  - ☑ mostrar o número que está armazenado na variável contador;
  - ☑ define o valor de contador para **contador - 1** e volta a repetir até que finalizem as quatro repetições;
  - ☑ ao sair da repetição, mostrar o ícone quadriculado.



Python

```
contador = 4
for index in range(4):
    basic.show_number(contador)
    contador = contador - 1
basic.show_icon(IconNames.CHESSBOARD)
```

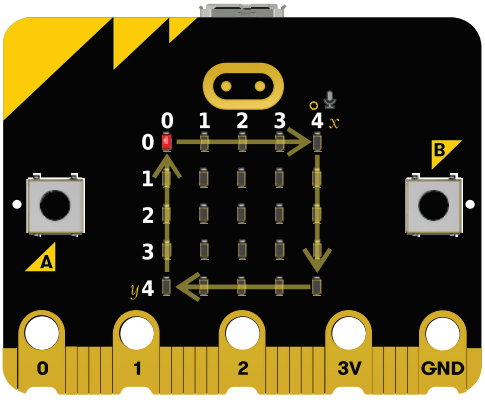
DESAFIO 1

Seu desafio é fazer nosso ator/objeto, denominado **sprite**, percorrer um quadrado. Para isto você irá explorar os recursos de repetições além de utilizar blocos e técnicas de produção de games.

> Um **sprite** é um objeto gráfico bi ou tridimensional que se move em uma tela sem deixar traços de sua passagem.
> Os **sprites** foram inventados originalmente como um método rápido de animação de várias imagens agrupadas numa tela.

Requisitos do projeto

- ☑ Quando iniciar, precisamos posicionar nosso sprite no canto superior esquerdo do diplay;
- ☑ ao pressionar Botão A repetir 4 vezes:
  - ☑ mova o sprite quatro LEDs para direita;
  - ☑ vire a direita por 90º;



Utilize os blocos abaixo para construir sua solução no Makecode.

Python

```
sprite = game.create_sprite(0, 0)
for index in range(4):
    sprite.move(4)
    sprite.turn(Direction.RIGHT, 90)
    basic.pause(500)
```

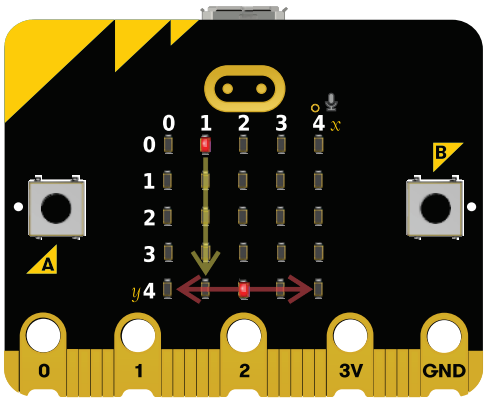
ESCAPE VIRUS

SINOPSE DO JOGO

Um evento terrível acaba de ocorrer no laboratório, a explosão de alguns frascos lançou para o teto toda a substância. Cuidado, esta substância viscosa pingando do teto vai aniquilar quem tocá-la. Você e seus colegas terão de trabalhar juntos se quiserem sobreviver a este ataque virótico. Cuidado, você tem apenas uma vida!!!

Requisitos do projeto

- ☑ Quando iniciar, sua pontuação iniciará em zero;
- ☑ Quando iniciar a posição do jogador será na coluna central (**x=2**);
- ☑ o **VIRUS** iniciará sempre em uma coluna (**x**) aleatória, descendo até a base (**y=4**);
- ☑ o ator/sprite **JOGADOR** se moverá somente na linha 4 (**y=4**);
- ☑ ao pressionar Botão A o **JOGADOR** se moverá para a esquerda;
- ☑ ao pressionar Botão B o **JOGADOR** se moverá para a direita;
- ☑ o ator/sprite **VIRUS** se moverá de cima para baixo, ou seja, sempre iniciando em **y=0**;
- ☑ cada vez que **VIRUS** tocar a base (**y=4**) a pontuação aumenta em 1;
- ☑ se **VIRUS** tocar em **JOGADOR**, será fim de jogo;
- ☑ se fim de jogo, mostrar no display da micro:bit a pontuação obtida;
- ☑ pressione A+B para jogar novamente.



PEDRA, PAPEL, TESOURA 09

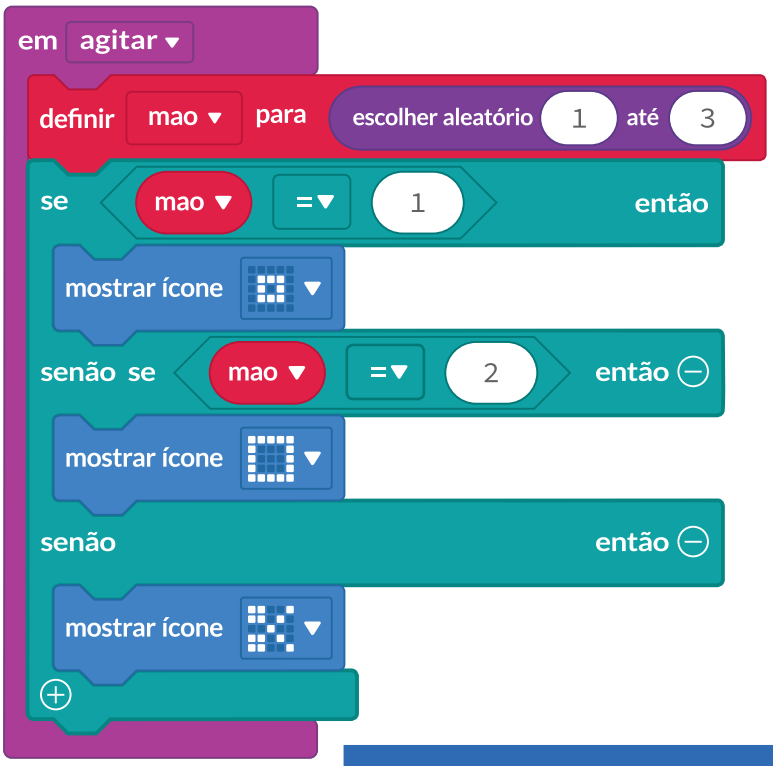
Neste capítulo, você irá explorar um jogo mais complexo: irá envolver lógica para determinar o quem é o vencedor. É o clássico pedra-papel-tesoura ou jon ken pon. Este é um projeto onde serão necessários duas micro:bits para que você possa jogar uma partida com seus amigos.

JOGOS CLÁSSICOS

O jogo Pedra, Papel, Tesoura é muito conhecido! No mundo todo, você encontrará pessoas que já jogaram, não importa o nome pelo qual ele é conhecido, todos nós o jogamos na nossa infância e na vida adulta também, repetidas vezes..

Ele é um jogo que parece muito justo, mas há uma desvantagem, que vem da velocidade dos truques do movimento da sua mão, ou do seu oponente. Você já deve ter observado isso.

A construção deste jogo com a micro:bit já é um clássico e é realmente muito simples, como você pode observar no código abaixo.



Python

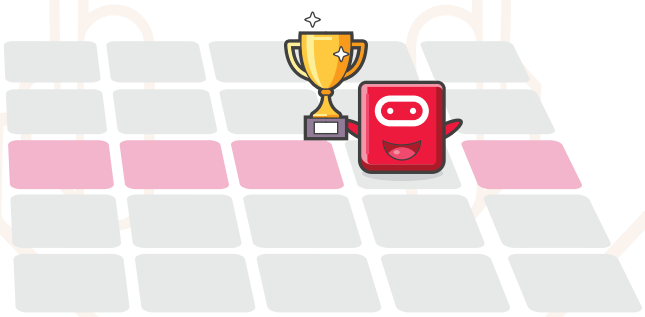
```
mao = 0

def on_gesture_shake():
    global mao
    mao = randint(1, 3)
    if mao == 1:
        basic.show_icon(IconNames.SMALL_SQUARE)
    elif mao == 2:
        basic.show_icon(IconNames.SQUARE)
    else:
        basic.show_icon(IconNames.SCISSORS)
input.on_gesture(Gesture.SHAKE, on_gesture_shake)
```

Com o programa acima, podemos jogar contra a micro:bit ou até mesmo, com um amigo(a), cada um com sua micro:bit, o que é bem divertido, porém podemos explorar ainda mais este jogo simples e transformá-lo em um jogo com muito mais recursos.

PIXEL MEMORY 10

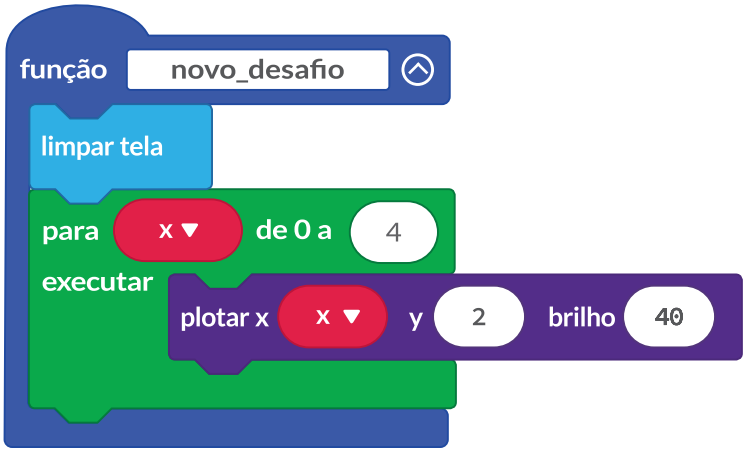
Neste capítulo, você irá explorar a criação de um jogo que irá desafiar sua memória e coordenação. Este jogo será dividido em várias funções menores. Para obter o aumento progressivo na dificuldade, serão utilizados conceitos de recursividade.
É melhor não piscar!



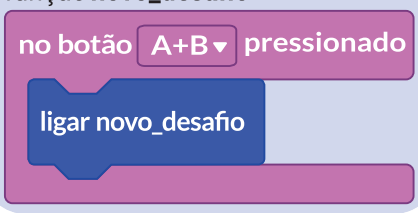
4. A categoria LED possui uma quantidade enorme de blocos que nos permitem manipular a matriz de LEDs. Como queremos ligar todos os LEDs da linha central, adicione o bloco **plotar x ... y ... brilho ...** dentro do loop **para**.

Para ligar um a um cada LED da linha central (eixo y = 2), arraste a variável **x** para dentro do bloco **plotar x**.

Utilizamos um brilho menor para os LEDs: o brilho vai de 0, sem brilho algum, até a maior intensidade, 255.



Para testar, você pode adicionar temporariamente este blocos à sua programação, ao **pressionar o botão A+B**, será executada a função **novo_desafio**



Agora é adicionar uma falha... um LED apagado ou sem brilho, que deverá ser memorizado durante o jogo.

Um ponto importante aqui: podemos simplesmente reduzir o brilho (apagar) de um LED, porém precisamos armazenar a posição deste LED para posteriormente comparar esta posição com o LED que o jogador marcar durante o desafio. Para isto, novamente iremos recorrer às variáveis.

5. Crie uma variável chamada **falha_x**; nela ficará armazenada a posição no eixo x da falha a ser lembrada pelo jogador.

CRIANDO FUNÇÕES

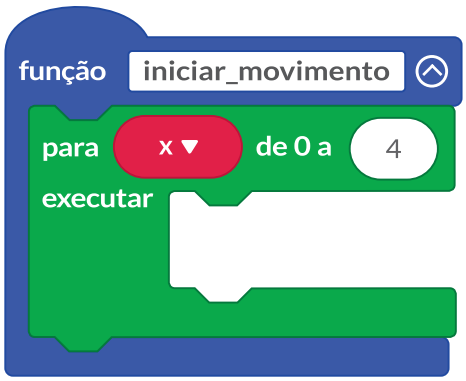
INICIAR_MOVIMENTO

Após programar o desafio e a contagem regressiva, a próxima ação do jogo será iniciar o movimento para que, no momento correto, o jogador pressione o botão B para indicar o local da falha. Podemos perceber, pela narrativa dos LEDs na linha central, que eles serão ligados um a um, com um intervalo entre eles, que é a velocidade estabelecida para o jogo.

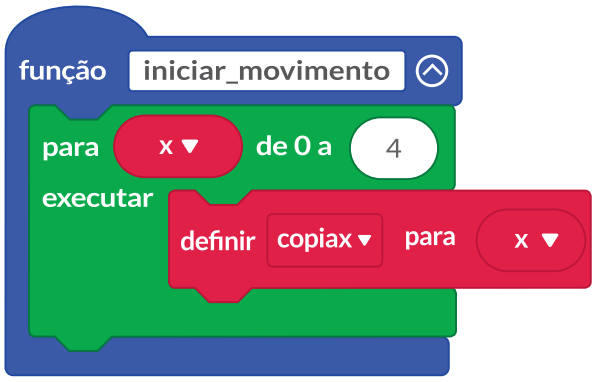
No momento em que o jogador apertar o botão B, precisaremos que seja possível identificar a posição em que o LED da falha se encontra. Para isto, iremos recorrer novamente ao uso de variáveis.

1. Acesse a categoria Funções, que está em avançado, e crie uma nova função com o nome **iniciar_movimento**;

2. Em seguida, adicione um bloco (loop) **para índice de 0 a 4**.O objetivo deste loop, novamente será acender os cinco LED da linha do meio. Substitua o nome da variável índice para **x**: não é obrigatório, mas ajuda a realizar a leitura e analisar a programação. Durante o loop, a coordenada **x** irá valer, 0, 1, 2, 3 e 4.

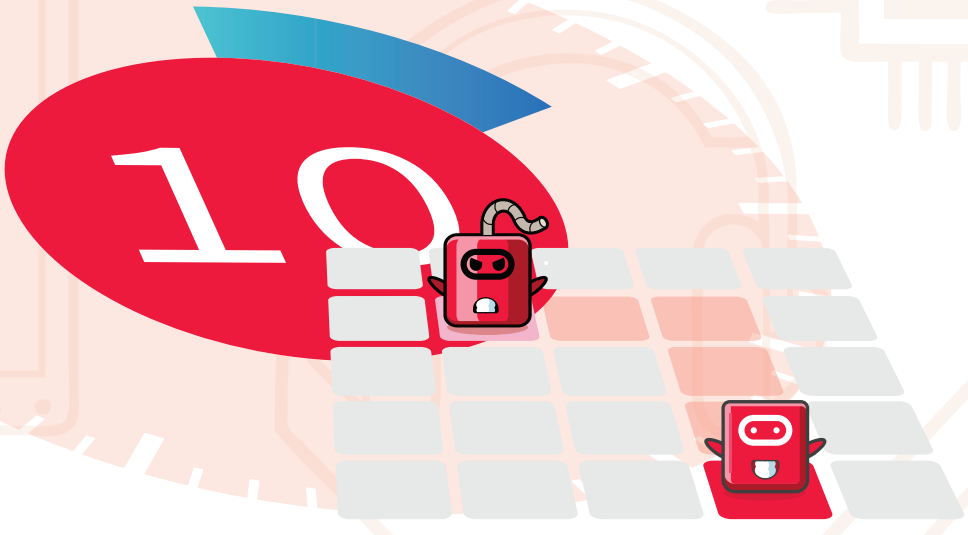


3. Precisamos de uma variável de apoio que armazene a posição da coordenada x, então crie uma variável; pode chamá-la de **copiax**. Esta variável receberá o valor de **x**. Iremos utilizar esta variável quando programarmos as ações ao pressionar o botão B.



CAMPO MINADO 11

Dez segundos... você tem apenas dez segundos para chegar até a bomba para desarmá-la. Neste capítulo, você irá desenvolver um game que irá testar suas habilidade e explorar os recursos do acelerômetro que está dentro da micro:bit.



MEDINDO O MOVIMENTO

ACELERÔMETRO

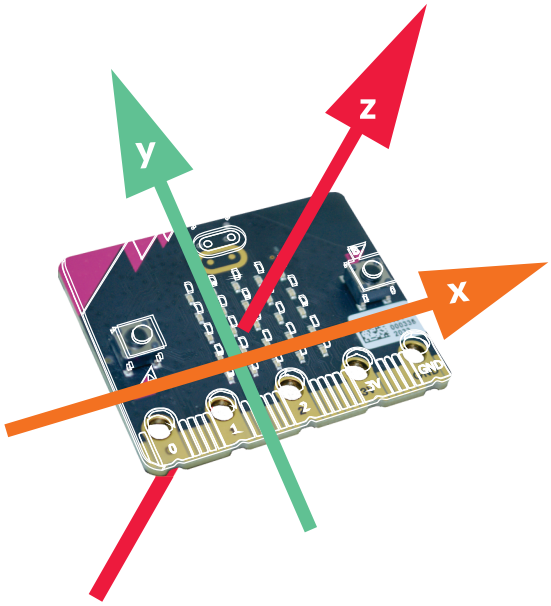
Um chip acelerômetro mede as forças que atuam sobre um minúsculo peso em cada uma das três dimensões.

Imagine a micro:bit repousando sobre uma mesa, a dimensão x é da esquerda para a direita; a dimensão y da frente para trás e a dimensão z dentro e fora da mesa (para cima e para baixo).

Se a micro:bit estiver repousando de forma perfeitamente plana, as forças nas dimensões x e y serão zero. A força agindo no eixo z terá um valor de cerca de -1000 devido à atração da gravidade. Se você movê-la, inclinando a parte de trás da micro:bit para cima, a força na dimensão y aumentará um pouco.

São essas pequenas mudanças na micro:bit que permitem que você use um acelerômetro para detectar a orientação da micro:bit, já que a gravidade estará sempre agindo na mesma direção, para baixo.

Há muitos projetos que podem explorar este recurso, desde um alarme de movimento, um medidor de pedaladas na bike, e claro, no jogo que será abordado, porém, primeiro iremos fazer um pequeno teste para você verificar como este sensor atua e como os valores são obtidos.



A micro:bit, com todos os seus sensores, como temperatura, giroscópio, luminosidade, campo magnético e intensidade sonora, transformam este pequeno dispositivo em um grande laboratório de física. Este potencial pode ser ampliado com o uso de placas de extensão como a **hack:bit**, permitindo a conexão de uma vasta gama de sensores externos, motores, display, etc...

Quais experimentos podem ser feitos?

Com sua micro:bit:, você pode explorar projetos que envolvem a aceleração da gravidade, aceleração de elevador, escada rolante, drones, foguetes, pêndulo simples, rotações sobre um eixo fixo, impulso, colisões unidimensionais, intensidade luminosa, batimentos e muito mais.

Neste projeto, iremos explorar este recurso para mover o ator (sprite) para alcançar a bomba a ser desarmada.

MEDINDO O MOVIMENTO

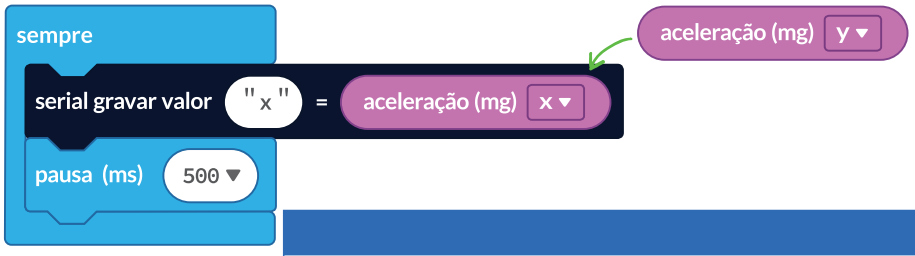
PORTA SERIAL

Iremos utilizar um recurso muito útil do Makecode que nos permite visualizar os dados coletados pelos sensores da micro:bit em tempo real. Para isto faremos o seguinte:

Mantenha a micro:bit conectada no computador por meio do cabo USB. Uma vez que você atualize a micro:bit com o programa, o Makecode irá receber os dados do acelerômetro, que os envia como dados seriais para seu computador.

Construa este simples programa abaixo e envie para sua micro:bit. O bloco **serial gravar valor x = 0** se encontra na categoria **Serial** em **Avançado** e tem a função de dar instruções a micro:bit para enviar os dados pela porta serial; neste caso, a conexão USB.

No exemplo abaixo, estão sendo enviados, a cada 500 milissegundos, os dados de aceleração do eixo x.



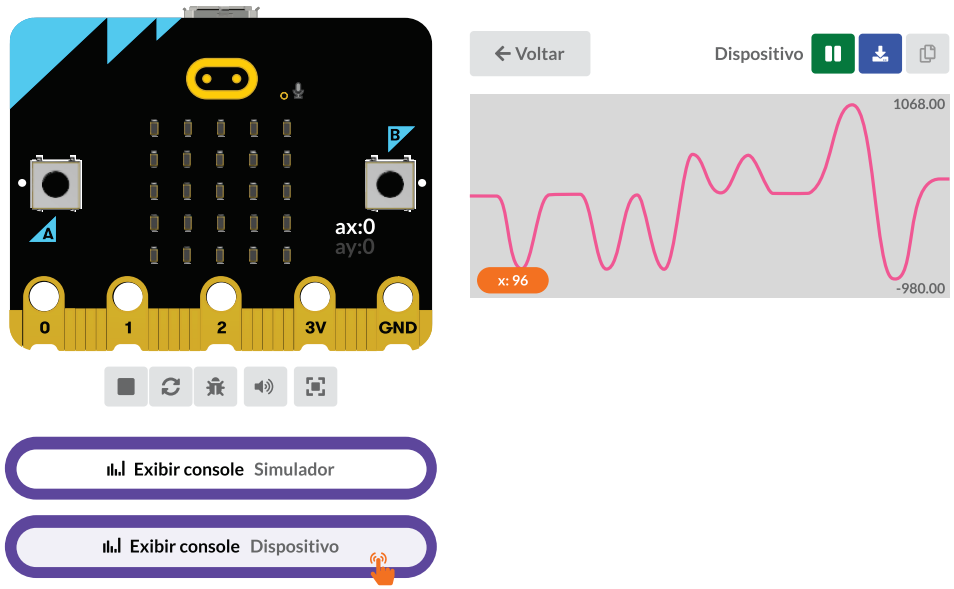

```
Python
def on_forever():
    serial.write_value("x", input.acceleration(Dimension.X))
basic.forever(on_forever)
```

Assim que você mover a micro:bit no eixo x, dados serão enviados ao computador e podem ser visualizados clicando no botão **Exibir console Dispositivo** que surgiu no simulador. Conforme você realiza os movimentos, você perceberá a mudança dos valores ocorrer enquanto é criado um gráfico em tempo real.

Experimente mudar o **eixo x** para **eixo y** e verifique os resultados.

Faça movimentos lentamente, além de movimentos mais bruscos. Observe as leituras...

Experimente remover o bloco pausa para que a leitura seja sem pausa entre uma e outra.



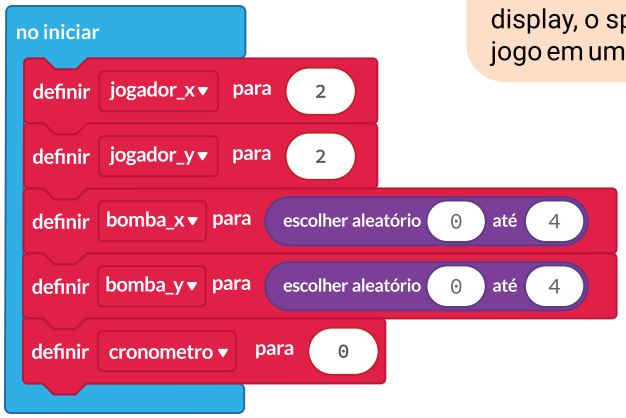
INIMIGO

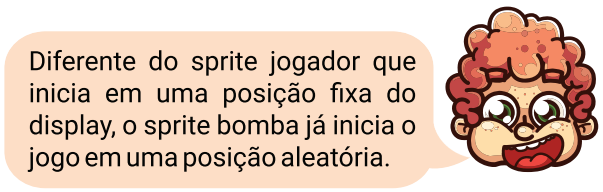
A BOMBA

Agora que você tem um sprite jogador em funcionando plenamente, é hora de colocar um inimigo no jogo. Para isto, teremos um sprite que aparecerá em qualquer um dos 25 LEDs, que não terá habilidades de movimento, pois é uma bomba. A ameaça está no fato de que haverá um temporizador (cronômetro). O jogador tem cerca de 10 movimentos para posicionar seu sprite em cima da bomba. Se for bem sucedida, a bomba é desativada e desaparece do visor. Uma nova bomba aparecerá imediatamente em um local diferente. Se o jogador não conseguir desarmar uma das bombas no tempo, ele será eliminado pela explosão e o jogo termina.

Com os requisitos acima, podemos afirmar que precisaremos de mais variáveis em nosso projeto, além das variáveis **jogador_x** e **jogador_y** já definidas anteriormente.

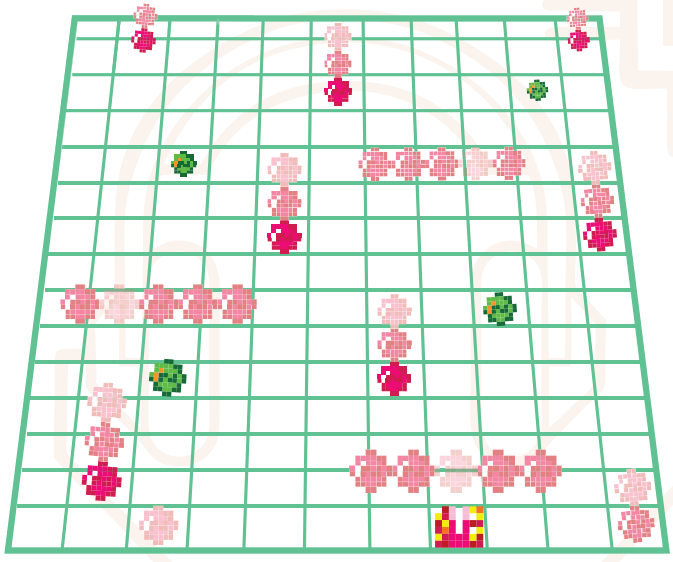
Para o controle das coordenadas da bomba, podemos criar **bomba_x** e **bomba_y**; para o temporizador, definimos **cronometro**.





PANDEMIA MISSÃO DE RESGATE 12

O vírus escapou e iniciou sua mutação! No jogo, você encontrará novas formas diferentes e velocidade variável de contágio, porém você identificou uma oportunidade de obtenção de pílulas que ajudam a manter sua vida. Prepare-se para vencer este inimigo!



BIT:REPEAT JOGO DA IMITAÇÃO 13

Neste capítulo, você irá construir um jogo que irá desafiar sua memória! É chegada a hora em que os melhores jogadores são aqueles com a melhor memória.

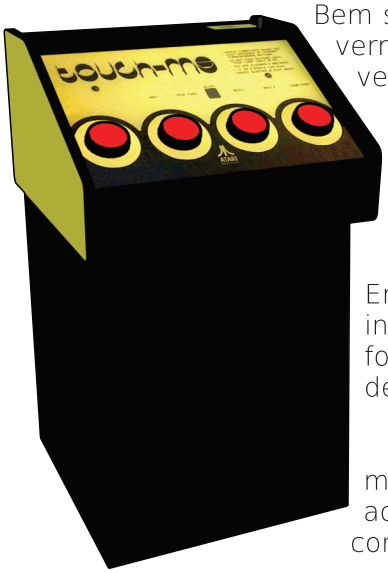
SIMON, O GÊNIO

Um dos primeiros jogos eletrônicos portáteis foi chamado Simon. Curiosamente, não tinha nem tela, não era ligado em uma televisão, mas era fascinante vê-lo criar uma série de tons e luzes para que o jogador repetisse a sequência. O jogador tinha a disposição quatro botões enormes, coloridos em vermelho, amarelo, azul e verde, dispostos em um círculo. Cada botão, ao ser pressionado, acendia sua respectiva lâmpada e cada um tinha uma nota musical particular associada a ele.

Muitas pessoas não sabem, mas a jogabilidade do Simon é baseada no arcade Touch Me da Atari de 1974. A partir dele, em 1978, a Atari desenvolveu o Touch-Me portátil, porém não fez o mesmo sucesso que o Simon.

Simon difere da versão portátil do Touch Me porque os LEDs eram todos da mesma cor e os sons que produziam não eram muito agradáveis. O lançamento desta versão portátil foi tarde demais - todos pensavam que o jogo da Atari era simplesmente um clone que buscava lucrar com o sucesso repentino de Simon , e não o contrário.

O funcionamento do Simon, é da seguinte maneira: o jogo inicia mostrando aleatoriamente uma das cores, por exemplo, vermelho. O botão vermelho acenderia e tocaria seu tom por um segundo. A tarefa do jogador era imitar o computador, pressionando o mesmo botão.



Bem simples e fácil, porém, na próxima rodada, o computador aciona vermelho novamente, seguido por outra cor aleatória, por exemplo, verde.

Então, cabe ao jogador pressionar vermelho e depois verde. Terceira rodada: vermelho, verde, vermelho. Quarta rodada: vermelho, verde, vermelho, azul. Rodada cinco: vermelho, verde, vermelho, azul, azul. E assim por diante.

Em pouco tempo, o jogo se torna bastante difícil. O que é interessante sobre a construção deste tipo de jogo é que ele nos força a pensar em uma abordagem diferente para o algoritmo e design.

Como não temos cores diferentes, nem quatro botões na micro:bit, usaremos outros recursos: acelerômetro em substituição aos quatro botões e quatro ícones diferentes para representar as cores.

Para as notas musicais, será mais fácil de resolver. Você pode seguir as notas musicais que desejar ou os tons do próprio Simon, que foram projetados para serem sempre harmônicos, não importando a ordem da sequência. Suas notas musicais se referem a tríade Lá Maior.

PARA IR ALÉM

E se colocássemos algum efeito visual no jogo?

Caio, que tal colocar um efeito para ir desaparecendo a seta gradualmente, até apagar? Nós estamos apenas ligando e depois limpando a tela.

Ed, Isso pode ficar bem legal. Podemos ir diminuindo o brilho, o maior valor é 255. Até chegar em 255 pode demorar muito se subtrair 1!!!
Talvez a Kelly tenha uma solução

Caio, podemos usar uma função chamada **MAP**. Este bloco **remapeia** um intervalo numérico para outro intervalo que você especificar.

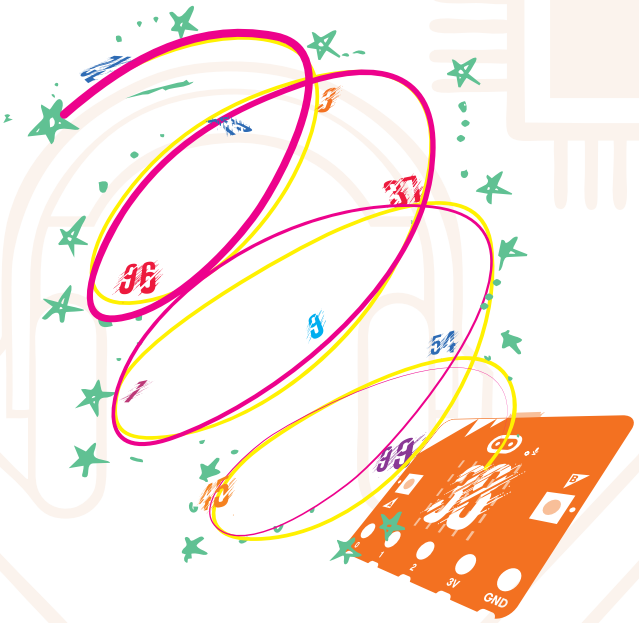
Parece complicado, mas é bem simples. 😎 😍
Por exemplo, queremos mapear os valores em um intervalo que vai de 0 a 9 para um novo intervalo entre 0 e 255. Quando ler um valor, por exemplo, 3, ele será remapeado em um novo valor na escala entre 0 e 255.

Teste o código abaixo! Pode ser em um novo projeto, apenas para você verificar o resultado.

Uau! Mostrou o valor 85.
Que legal, ele fez uma regra de três!!! 😳😳
Se 9 equivale a 255, que é o valor máximo, então 3 equivale a 85.

MAGIC:BIT 14

Vamos construir um pequeno jogo para que você tenha uma ideia de como algoritmos diferentes para o mesmo problema podem ter desempenhos completamente diferentes. Não é mágica, é matemática!



DIVIDIR PARA CONQUISTAR

Ed, aprendi uma coisa muito legal nas aulas de Pensamento Computacional, topa fazer um jogo de adivinhação?

Legal, Caio, gosto desses desafios! O que você propõe?

Você pensa em um número de 1 a 100, e garanto que a micro:bit irá adivinhar o número em que você pensou em no máximo 7 vezes!

Eu topo! Vai sonhando que vai conseguir...

Ed, "foi mal"... mas perdeu!!! 😎 😍

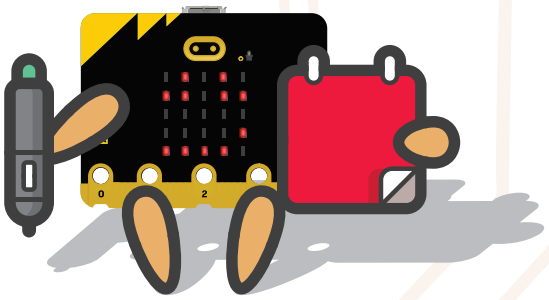
😳😳

Me pegou, Caio! Me mostra como que isso funciona...

Tranquilo, Ed, vamos falar de Busca Binária e, claro, vamos usá-la em um game...

PENSAMENTO COMPUTACIONAL 16

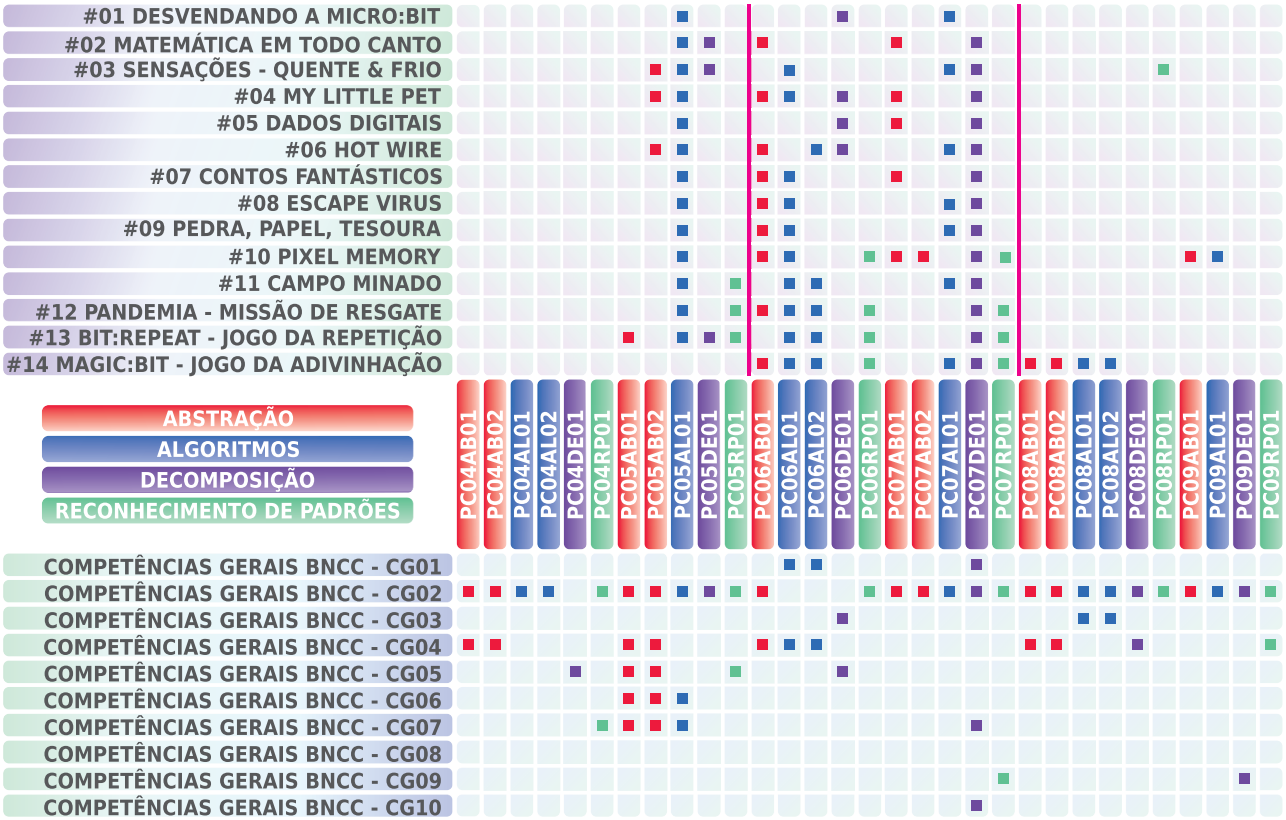
Neste apêndice nosso objetivo é apresentar o Currículo de Tecnologia e Computação (CIEB), sua correlação com a Base Nacional Comum Curricular e como o conteúdo deste livro está conectado a esta matriz.



MATRIZ

Alguns dos projetos deste livro possuem uma lista de critérios de sucesso. Nosso objetivo é que, ao adotar esta técnica de Gestão de Projetos, o leitor terá um ponto focal e irá manter uma lista do que precisa ser alcançado. Tais elementos são importantes como habilidade STEM, pois projetos e produtos sempre têm critérios que precisam ser cumpridos para que sejam bem sucedidos.

O objetivo da matriz abaixo é estabelecer um mapeamento do conteúdo deste livro com o Currículo de Tecnologia e Computação desenvolvido pelo Centro de Inovação para a Educação Brasileira (CIEB).



ABSTRAÇÃO

Este conceito envolve a filtragem dos dados e sua classificação, ignorando elementos que não são necessários, visando aos que são relevantes. Envolve também formas de organizar informações em estruturas que possam auxiliar na resolução de problemas.

ALGORITMOS

É um conceito que agrega todos os demais. O algoritmo é um plano, uma estratégia ou um conjunto de instruções claras e necessárias para a solução de um problema. Em um algoritmo, as instruções são descritas e ordenadas para que o objetivo seja atingido e podem ser escritas em formato de diagramas, pseudocódigo (linguagem humana) ou escritos em códigos, por meio de uma linguagem de programação.

RECONHECIMENTO DE PADRÕES

Trabalha a identificação de características comuns entre os problemas e suas soluções. Resulta do fato de realizar a decomposição de um problema complexo para encontrar padrões entre os subproblemas gerados. Estes padrões são similaridades ou têm características que alguns dos problemas compartilham e que podem ser explorados para que sejam solucionados de forma mais eficiente.

DECOMPOSIÇÃO

A decomposição trabalha o processo pelo qual os problemas são divididos em partes menores e mais fáceis de resolver. Compreende também a prática de analisar problemas a fim de identificar que partes podem separadas, e também de que forma podem ser reconstituídas para a solução de um problema global. Essa prática também possibilita aumentar a atenção aos detalhes.

fonte:https://curriculo.cieb.net.br/

CURRÍCULO 6º ANO EF

PENSAMENTO COMPUTACIONAL

CONCEITO: ABSTRAÇÃO

PC06AB01 - Interpretar um algoritmo em pseudo-linguagem e transpor para uma linguagem de programação visual e vice-versa.

Nível de Adoção de Tecnologia

Competências Gerais BNCC: 01 02 03 04 05 06 07 08 09 10

EF06MA04 - Construir algoritmo em linguagem natural e representá-lo por fluxograma que indique a resolução de um problema simples (por exemplo, se um número natural qualquer é par).

CONCEITO: ALGORITMOS

PC06AL01 - Experienciar e construir algoritmos com desvios condicionais utilizando uma linguagem de programação visual (blocos).

Nível de Adoção de Tecnologia

Competências Gerais BNCC: 01 02 03 04 05 06 07 08 09 10

EF06MA04 - Construir algoritmo em linguagem natural e representá-lo por fluxograma que indique a resolução de um problema simples (por exemplo, se um número natural qualquer é par).

CONCEITO: ALGORITMOS

PC06AL02 - Encontrar e solucionar problemas em programas (depurar) utilizando uma linguagem de programação visual (blocos).

Nível de Adoção de Tecnologia

Competências Gerais BNCC: 01 02 03 04 05 06 07 08 09 10

EF06MA04 - Construir algoritmo em linguagem natural e representá-lo por fluxograma que indique a resolução de um problema simples (por exemplo, se um número natural qualquer é par).

CONCEITO: DECOMPOSIÇÃO

PC06DE01 - Identificar e categorizar elementos que compõem a interface de um ambiente de programação visual (menus, botões, painéis etc.).

Nível de Adoção de Tecnologia

Competências Gerais BNCC: 01 02 03 04 05 06 07 08 09 10

EF69AR02 - Pesquisar e analisar diferentes estilos visuais, contextualizando-os no tempo e no espaço.
EF69AR06 - Desenvolver processos de criação em artes visuais, com base em temas ou interesses artísticos, de modo individual, coletivo e colaborativo, fazendo uso de materiais, instrumentos e recursos convencionais, alternativos e digitais.

CONCEITO: RECONHECIMENTO DE PADRÕES

PC06RP01 - Identificar padrões de instruções que se repetem em um algoritmo e utilizar um módulo ou função para representar estas instruções.

Nível de Adoção de Tecnologia

Competências Gerais BNCC: 01 02 03 04 05 06 07 08 09 10

EF06MA04 - Construir algoritmo em linguagem natural e representá-lo por fluxograma que indique a resolução de um problema simples (por exemplo, se um número natural qualquer é par).

fonte:https://curriculo.cieb.net.br/

BNCC
COMPETÊNCIAS GERAIS

CONHECIMENTO [CG01]

Valorizar e utilizar os conhecimentos historicamente construídos sobre o mundo físico, social, cultural e digital para entender e explicar a realidade, continuar aprendendo e colaborar para a construção de uma sociedade justa, democrática e inclusiva.

PENSAMENTO CIENTÍFICO, CRÍTICO E CRIATIVO [CG02]

Exercitar a curiosidade intelectual e recorrer à abordagem própria das ciências, incluindo a investigação, a reflexão, a análise crítica, a imaginação e a criatividade, para investigar causas, elaborar e testar hipóteses, formular e resolver problemas e criar soluções (inclusive tecnológicas) com base nos conhecimentos das diferentes áreas.

REPERTÓRIO CULTURAL [CG03]

Valorizar e fruir as diversas manifestações artísticas e culturais, das locais às mundiais, e também participar de práticas diversificadas da produção artístico-cultural.

COMUNICAÇÃO [CG04]

Utilizar diferentes linguagens – verbal (oral ou visual-motora, como Libras, e escrita), corporal, visual, sonora e digital –, bem como conhecimentos das linguagens artística, matemática e científica, para se expressar e partilhar informações, experiências, ideias e sentimentos em diferentes contextos, além de produzir sentidos que levem ao entendimento mútuo.

CULTURA DIGITAL [CG05]

Compreender, utilizar e criar tecnologias digitais de informação e comunicação de forma crítica, significativa, reflexiva e ética nas diversas práticas sociais (incluindo as escolares) para se comunicar, acessar e disseminar informações, produzir conhecimentos, resolver problemas e exercer protagonismo e autoria na vida pessoal e coletiva.

TRABALHO E PROJETO DE VIDA [CG06]

Valorizar a diversidade de saberes e vivências culturais, apropriar-se de conhecimentos e experiências que lhe possibilitem entender as relações próprias do mundo do trabalho e fazer escolhas alinhadas ao exercício da cidadania e ao seu projeto de vida, com liberdade, autonomia, consciência crítica e responsabilidade.

ARGUMENTAÇÃO [CG07]

Argumentar com base em fatos, dados e informações confiáveis, para formular, negociar e defender ideias, pontos de vista e decisões comuns que respeitem e promovam os direitos humanos, a consciência socio-ambiental e o consumo responsável em âmbito local, regional e global, com posicionamento ético em relação ao cuidado de si mesmo, dos outros e do planeta.

AUTOCONHECIMENTO E AUTOCUIDADO [CG08]

Conhecer-se, apreciar-se e cuidar de sua saúde física e emocional, compreendendo-se na diversidade humana e reconhecendo suas emoções e as dos outros, com autocrítica e capacidade para lidar com elas.

EMPATIA E COOPERAÇÃO [CG09]

Exercitar a empatia, o diálogo, a resolução de conflitos e a cooperação, fazendo-se respeitar e promovendo o respeito ao outro e aos direitos humanos, com acolhimento e valorização da diversidade de indivíduos e de grupos sociais, seus saberes, suas identidades, suas culturas e suas potencialidades, sem preconceitos de qualquer natureza.

RESPONSABILIDADE E CIDADANIA [CG10]

Agir pessoal e coletivamente com autonomia, responsabilidade, flexibilidade, resiliência e determinação, tomando decisões com base em princípios éticos, democráticos, inclusivos, sustentáveis e solidários.

fonte:https://curriculo.cieb.net.br/