

# 第17回 IEEE Engineer Spotlight

動く！光る！音が鳴る！アイデアをカタチにしよう！  
～今年の自由研究はプログラミングと電子工作だ！～

(株)東芝 研究開発センター / IEEE Tokyo Young Professionals Vice Chair

石垣 雄太郎

2021年7月24日

※本資料の無断転載を禁じます

※本講演のコンテンツ(プログラム例、工作例等)によって生じる損害の責任は負いかねます。ご同意の上、自己責任でご活用ください。

# 自己紹介: 石垣 雄太郎

所属: (株)東芝 研究開発センター / IEEE Tokyo Young Professionals Vice Chair

## 略歴

- 2007～2012 東京工業高等専門学校 (電子工学科)
- 2012～2016 東京農工大学・大学院 (電気電子工学科)
- 2016～ (株)東芝 ・ 東芝デバイス&ストレージ(株)

通称「高専」  
 高校と異なり5年制の学校

理系を目指したい人は、  
 選択肢の一つとしてご参考に！



## IEEE (アイトリプリー) とは？

- 電気・情報工学などの分野を中心とする世界最大の技術団体
- 活動内容
  - 学術会議の開催、論文誌の発行、技術標準化活動
  - セミナー開催、国際交流、教育活動

...など様々な活動

IEEE Engineer Spotlightではさまざまな講演を聴くことができます。

アーカイブ

第1回  
インターネットとIEEE [講演会](#)



日時 2020年5月31日(日) 午前 9:00～10:00

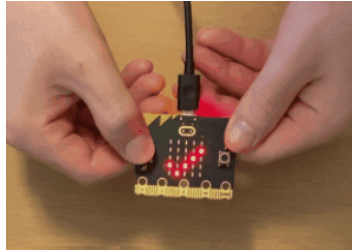
講演者 西原明法  
(東京工業大学名誉教授 / 東京工業大学超スマート社会学術教育院特任教授 /  
IEEE Region 10 Director / IEEE Life Fellow / 電子情報通信学会フェロー)

▼詳細▼  
オンライン視聴

# 今日のお話

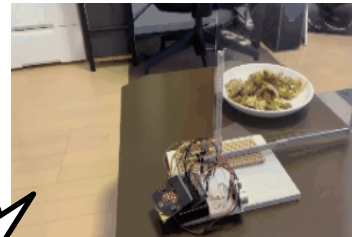
1

ミニゲームを作って  
プログラミングの<sup>きほん</sup>基本を学習!



2

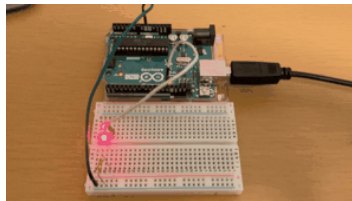
<sup>かいへい</sup>自動開閉パーティション  
を作って<sup>でんし</sup>電子工作を<sup>たいけん</sup>体験!



コロナ  
<sup>たいさく</sup>対策!

3

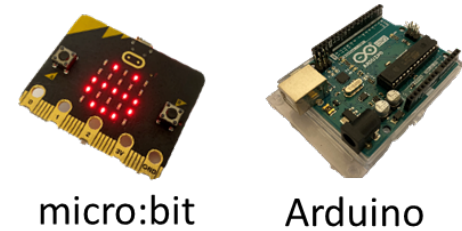
<sup>てんめつじっけん</sup>LED点滅実験で  
かいるせつけいたいけん  
回路設計体験!



自由研究にもぜひ!

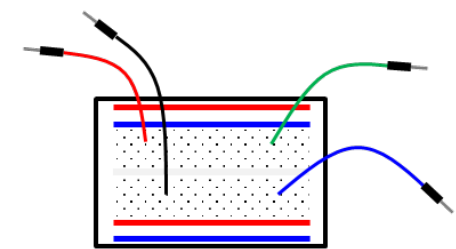
```
ずっと  
もし ボタンAが押されているなら  
音を鳴らす  
一時停止
```

プログラミングで  
自在にコントロール!

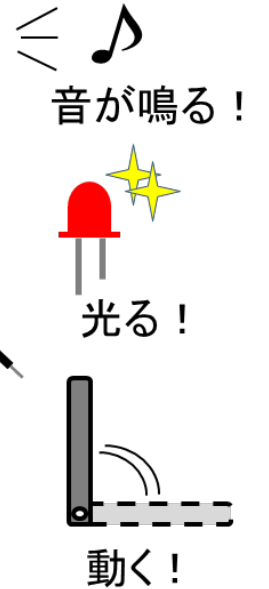


micro:bit

Arduino



ブレッドボードで  
回路を作る!  
(はんだ付け不要)



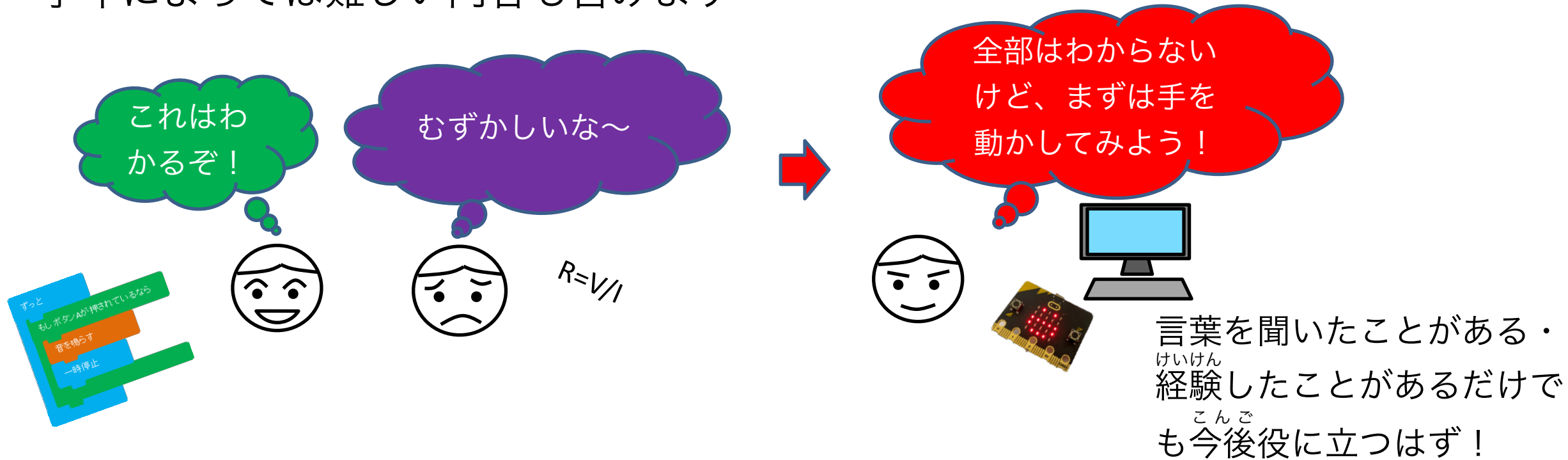
音が鳴る!

光る!

動く!

# 今日お話するにあたって

学年によっては<sup>むずか</sup>難しい内容も<sup>ふく</sup>含みます



今日のお話し中は、手を動かしてもらおう時間はないので、あとから動画や資料を<sup>みかえ</sup>見返して、ぜひチャレンジ!

動画と資料<sup>しりょう</sup>→「エンジニアスポットライト」で検索!<sup>けんさく</sup>

# アイデアをカタチにするには？

「できること」は多い方がいい

「できること」が少ないと...

アイデアがうかんでも<sup>じっげん</sup>実現できない



あんなこといいな..  
できたらいいな..



でも、どうやって作っ  
たらいいかわからない...

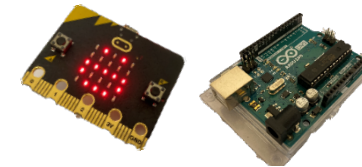
<sup>げんじつてき</sup>現実的なアイデアがうかばない



<sup>いちおう</sup>一応<sup>はんい</sup>できる範囲で考えたがこれ  
は一体何の役に立つんだ...

今日はみなさんの「できること」を増やします！  
アイデアのヒントになるような、<sup>れい</sup>工作例もいくつか<sup>しめ</sup>示します！

マイコンプログラミング



<sup>げんじつせかい</sup>“現実世界で”モノ  
を動かしたり、光  
らせたりできる！

# 工具と部品について

以下のページを参考に、必要なものをそろえてください

[https://www.ieee-jp.org/japancouncil/affinitygroup/R10\\_Spotlight/images/tools17.pdf](https://www.ieee-jp.org/japancouncil/affinitygroup/R10_Spotlight/images/tools17.pdf)

## 使うもの

必要に応じて部品や工具\*を購入してください。オンデマンド配信も予定しているため、講演後に購入頂いても問題ありません。

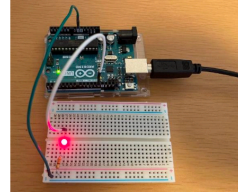
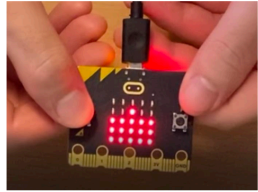
\*お近くのホームセンターやオンラインショップ等で購入することができます。実習される場合必要に応じてご準備ください。(micro:bit V2はお早めの購入をおすすめします。)

### 講演での実習内容

① ミニゲーム作成・プログラミングの基礎学習

② 自動開閉パーティション

③ Arduino LED点灯実験



### 工具表

### 実習内容

品名	①	②	③
ニッパー	-	★	-
ラジオペンチ	-	★	★
プラスドライバー (No. 1, No. 2)	-	★	-
錐 (キリ)	-	★	-

★: 必須 ■: 任意 -: 不要

次ページも同様



ニッパー



ラジオペンチ



プラスドライバー



錐 (キリ)

## 部品表

### 実習内容

※価格は2021年6月末時点のもの

★: 必須 ■: 任意 -: 不要

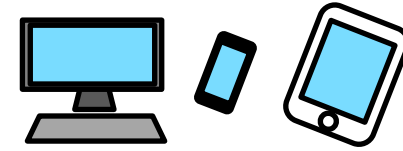
品名	【参考】価格(円)	個数	【参考】購入できる場所 (URL)	実習内容			備考
				①	②	③	
micro:bit go V2スターターキット	2400	1個	<a href="https://akizukidenshi.com/catalog/g/gk-16019/">https://akizukidenshi.com/catalog/g/gk-16019/</a>				【おすすめ】USBケーブル、電池ボックス付き
micro:bit V2	2200		<a href="https://akizukidenshi.com/catalog/g/gm-15882/">https://akizukidenshi.com/catalog/g/gm-15882/</a>	★	★	-	【おすすめ】
micro:bit V1	2160		<a href="https://akizukidenshi.com/catalog/g/gm-12513/">https://akizukidenshi.com/catalog/g/gm-12513/</a>				
USBケーブル A-microB	120	1	<a href="https://akizukidenshi.com/catalog/g/gc-07607/">https://akizukidenshi.com/catalog/g/gc-07607/</a>	★	★	-	micro:bit go V2スターターキットを購入した場合は不要
micro:bitブレイクアウトボード	680	1	<a href="https://akizukidenshi.com/catalog/g/gp-12836/">https://akizukidenshi.com/catalog/g/gp-12836/</a>	-	★	-	
Arduino Uno Rev3	2940	1	<a href="https://akizukidenshi.com/catalog/g/gm-07385/">https://akizukidenshi.com/catalog/g/gm-07385/</a>	-	-	★	
USBケーブル (Aオス-Bオス)	130	1	<a href="https://akizukidenshi.com/catalog/g/gc-07605/">https://akizukidenshi.com/catalog/g/gc-07605/</a>	-	-	★	
ACアダプタ(9V, 2.5A)	900	1	<a href="https://akizukidenshi.com/catalog/g/gm-10877/">https://akizukidenshi.com/catalog/g/gm-10877/</a>	-	-	■	パソコンから電源供給しない場合は必要
ブレッドボード	200	1	<a href="https://akizukidenshi.com/catalog/g/gp-05294/">https://akizukidenshi.com/catalog/g/gp-05294/</a>	-	★	★	
ブレッドボード用ジャンパー線 (オス-メス) 10本入	220	1	<a href="https://akizukidenshi.com/catalog/g/gc-08932/">https://akizukidenshi.com/catalog/g/gc-08932/</a>	-	★	-	黒色 (リンク先から他の色も選べます)
ブレッドボード用ジャンパー線 (オス-オス) 10本入	350	1	<a href="https://akizukidenshi.com/catalog/g/gp-02933/">https://akizukidenshi.com/catalog/g/gp-02933/</a>	-	★	★	黒色 (リンク先から他の色も選べます)
ブレッドボード ジャンパーワイヤ	400	1	<a href="https://akizukidenshi.com/catalog/g/gp-00288/">https://akizukidenshi.com/catalog/g/gp-00288/</a>	-	★	★	
カーボン抵抗 10kΩ 1/4W (100個入)	100	1	<a href="https://akizukidenshi.com/catalog/g/gR-25103/">https://akizukidenshi.com/catalog/g/gR-25103/</a>	-	★	-	
カーボン抵抗 20kΩ 1/4W (100個入)	100	1	<a href="https://akizukidenshi.com/catalog/g/gR-03940/">https://akizukidenshi.com/catalog/g/gR-03940/</a>	-	★	-	
カーボン抵抗 330Ω 1/4W (100個入)	100	1	<a href="https://akizukidenshi.com/catalog/g/gR-25331/">https://akizukidenshi.com/catalog/g/gR-25331/</a>	-	■	★	
赤色LED 10個入	100	1	<a href="https://akizukidenshi.com/catalog/g/gl-02320/">https://akizukidenshi.com/catalog/g/gl-02320/</a>	-	■	★	
三端子レギュレータ TA48033S (3.3V 1A)	100	1	<a href="https://akizukidenshi.com/catalog/g/gl-00534/">https://akizukidenshi.com/catalog/g/gl-00534/</a>	-	★	-	
超音波距離センサ HC-SR04	450	1	<a href="https://akizukidenshi.com/catalog/g/gM-11009/">https://akizukidenshi.com/catalog/g/gM-11009/</a>	-	★	-	
サーボモータ SG92R	500	1	<a href="https://akizukidenshi.com/catalog/g/gM-08914/">https://akizukidenshi.com/catalog/g/gM-08914/</a>	-	★	-	
スイッチ付き電池ボックス 単3×3本	90	1	<a href="https://akizukidenshi.com/catalog/g/gP-02666/">https://akizukidenshi.com/catalog/g/gP-02666/</a>	-	★	-	
電池ボックス 単3×2本 PHコネクタ付	70	1	<a href="https://akizukidenshi.com/catalog/g/gP-12665/">https://akizukidenshi.com/catalog/g/gP-12665/</a>	■	-	-	パソコンから電源供給しない場合は必要 (micro:bit go V2スターターキットを買った場合は単4のものが付属しているので不要)
アルカリ単3乾電池 4本入	80	1	<a href="https://akizukidenshi.com/catalog/g/gB-03256/">https://akizukidenshi.com/catalog/g/gB-03256/</a>	■	★	-	
ユニバーサルプレートセット	350	1	<a href="https://akizukidenshi.com/catalog/g/gP-09100/">https://akizukidenshi.com/catalog/g/gP-09100/</a>	-	★	-	
ユニバーサルアームセット	400	1	<a href="https://akizukidenshi.com/catalog/g/gK-10258/">https://akizukidenshi.com/catalog/g/gK-10258/</a>	-	★	-	
タッピングネジ (太さ3mm, 長さ16mm)		1		-	★	-	
木板 はがき倍判 (15cm×20cm, 厚さ1cm)		1	お近くのホームセンターなどで購入してください。	-	★	-	サイズは任意だが、縦横サイズ15cm×20cm, 厚さ1cm以上を推奨。
プラバン B4サイズ 厚み0.3mm		1		-	★	-	例: <a href="https://item.rakuten.co.jp/acrysunday/1017359/">https://item.rakuten.co.jp/acrysunday/1017359/</a>
プラスチック対応接着剤		1		-	★	-	例: <a href="https://amzn.to/3yktVt4">https://amzn.to/3yktVt4</a>

1

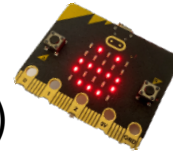
プログラミングの<sup>きほん</sup>基本  
～micro:bitを使って学ぼう～

# コンピュータの<sup>きほん</sup>基本

コンピュータと聞いて何を思いうかべる？



今日使うコンピュータは“マイコン”  
(マイクロコントローラ / マイクロコンピュータ)



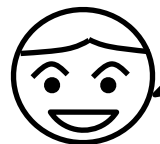
あらゆるものに組  
み込まれている

コンピュータの<sup>おも</sup>主な<sup>やくわり</sup>役割は“**計算**”と“**制御**(コントロール)”

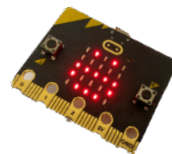
例: 一秒間に数千万回以上の計算ができる

<sup>たと</sup>例えば、スマホゲームの  
<sup>うら</sup>裏では、<sup>ふくざつ</sup>複雑な<sup>せいぎよ</sup>計算や制御  
が動いている

<sup>いっぱんてき</sup>一般的なコンピュータは**プログラム(命令)通り**に動く



LEDを光らせて！  
そのあと音を鳴らして！



りょうかい～

~~めんどいから  
やだ～~~

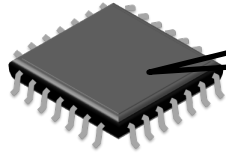


# プログラミングとは？

- コンピュータへの命令を<sup>きじゆつ</sup>記述すること

プログラム(命令)

0100010...  
0010101...  
0100100...  
0100100...



コンピュータは“1”と“0”  
しかあつかえない！

コンピュータが<sup>りかい</sup>理解できる形式でプログラミングするのはたいへん...



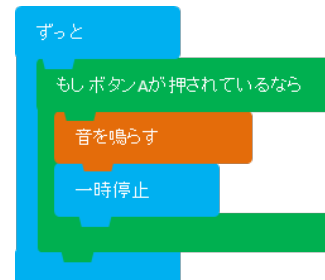
<sup>ふつう</sup>普通は人間が<sup>りかい</sup>理解しやすい<sup>げんご</sup>言語やブロックでプログラミングする

<sup>せんよう</sup>専用のソフトでコン  
ピュータが<sup>りかい</sup>理解でき  
る形式に<sup>へんかん</sup>変換する

<sup>げんご</sup>言語(文字)を使ったもの  
C言語, Python...など

```
int main()  
{  
    printf("HelloWorld");  
    return 0;  
}
```

ブロックを使ったもの  
Scratch, MakeCode...など



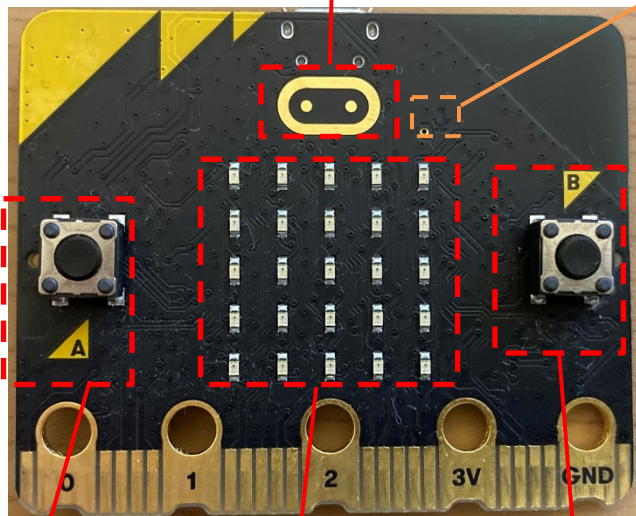
<sup>まちが</sup>間違った命令を与えると、<sup>ま</sup>間  
<sup>ちが</sup>違った通りに動く。(バグ)  
正しい命令を<sup>あた</sup>与えよう。

# micro:bit

ひょうめん きょうつう  
表面 (共通)

タッチセンサ

V2の場合は  
マイク用LED

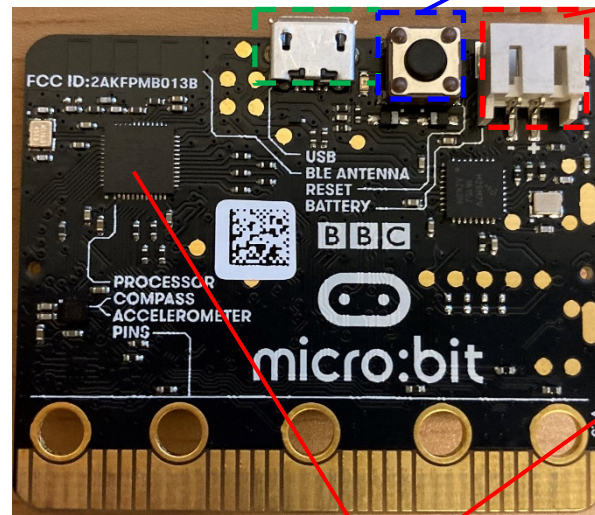


Aボタン LED (5×5個) Bボタン

りめん  
裏面 (micro:bit V1)

たんし  
USB端子(MicroB)

リセットボタン

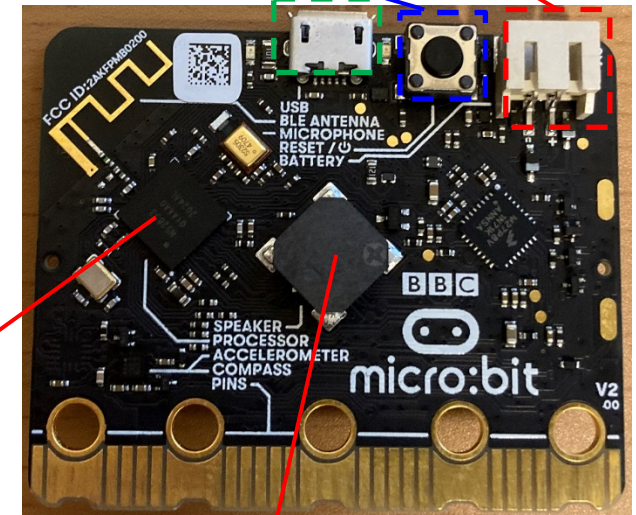


マイコン

りめん  
裏面 (micro:bit V2)

でんげん  
電源コネクタ

でんち  
(電池ボックスと接続)



スピーカー

せいでんき ひじょう  
静電気に非常に弱いので、できるだけ端子やマイコンなど(小さな黒い部品)を素手でさわらないように！

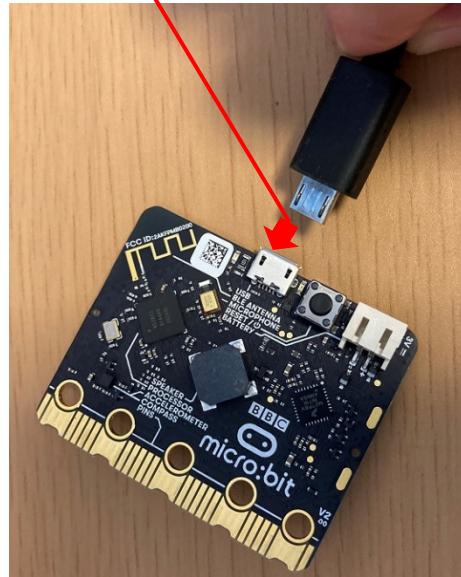
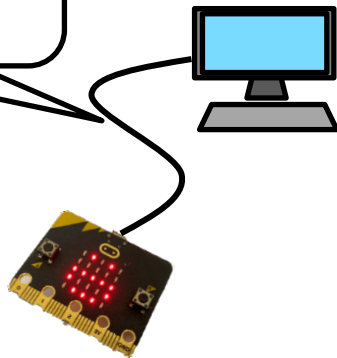
# micro:bitのプログラミング方法

- パソコンとmicro:bitをUSBケーブル(A-MicroB)でつないでプログラミング

① USBケーブルのMicroB側(端子が小さい方)をmicro:bitに接続

② USBケーブルのA側(端子が大きい方)をパソコンに接続

PCでプログラミングして、micro:bitにプログラムを送る



どちらでもOK

# MakeCodeでのプログラミングのはじめ方

Windows10では<sup>せんよう</sup>専用アプリもあるが、ここではブラウザ(**Google Chromeがおすすめ**)を使ってプログラミング

URL: <https://makecode.microbit.org/> または、<sup>けんさく</sup>検索エンジンで「makecode microbit」と<sup>けんさく</sup>検索

”プロジェクト”という単位で  
プログラミングを行う

プロジェクト名「test」

プロジェクト名「タイミングゲーム」

プロジェクト名「自動開閉パーティション」

好きな名前を入力 (日本語もOK)



# MakeCodeでのプログラミングのはじめ方

動画

The screenshot shows the Microsoft MakeCode for micro:bit website. A dialog box titled "プロジェクトを作成する" (Create Project) is open in the center. The dialog contains the text "プロジェクトに名前をつけてください。" (Please name your project.) and a text input field with "test" entered. Below the input field is a link for "コードのオプション" (Code options). At the bottom right of the dialog is a green button labeled "作成" (Create) with a checkmark icon. The background of the website shows a "マイプロジェクト" (My Projects) section with a "新しいプロジェクト" (New Project) button and a "チュートリアル" (Tutorials) section with various project cards like "点滅するハート" (Blinking Heart), "Name Tag", "Smiley Buttons", "Dice", "Love Meter", and "Micro Chat".

もど  
ホームに戻る

# 画面の見方

Microsoft | micro:bit

ブロック JavaScript

検索...

- 基本
- 入力
- 音楽
- LED
- 無線
- ループ
- 論理
- 変数
- 計算
- 高度なブロック

最初だけ

ずっと

プログラミング画面

シミュレータ

プログラムのダウンロード

ブロックリスト

もど 戻るボタン

ダウンロード

test

# まずはLEDを点滅させてみよう！

てんめつ

動画

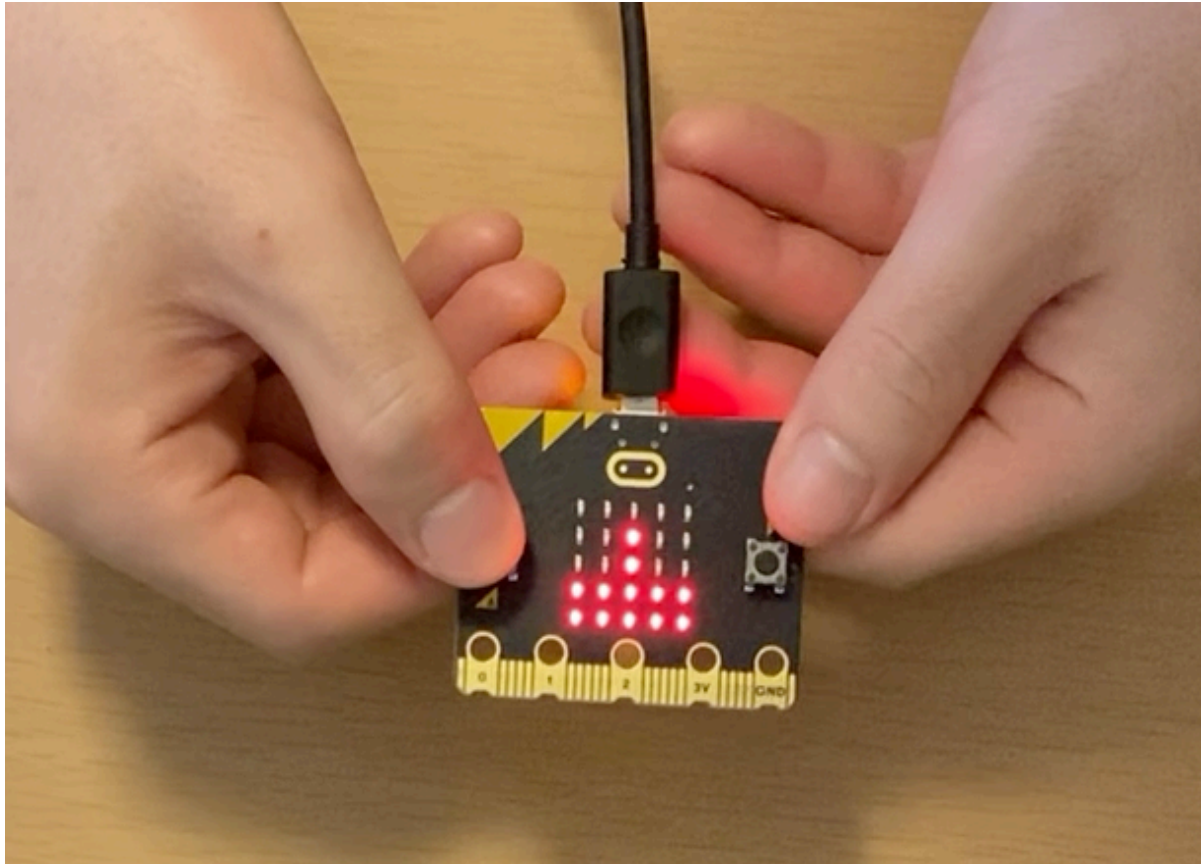
The screenshot shows the Microsoft MakeCode for micro:bit editor interface. On the left, there is a visual representation of the micro:bit board with several red LEDs lit. Below it are playback controls. The central pane shows a search bar and a list of block categories: 基本 (Basic), 入力 (Input), 音楽 (Music), LED, 無線 (Wireless), ループ (Loops), 論理 (Logic), 変数 (Variables), 計算 (Math), and 高度なブロック (Advanced Blocks). The right pane shows a 'ずっと' (Forever) loop block containing four steps: 'アイコンを表示' (Show icon), '一時停止 (ミリ秒) 500' (Pause 500 ms), '表示を消す' (Hide icon), and another '一時停止 (ミリ秒) 500' (Pause 500 ms). The bottom of the editor features a 'ダウンロード' (Download) button, a text input field containing 'test', and several utility icons.

\*動画はWindows10だが、Macでも基本的に同様の手順きほんてき どうよう ていじゆん

# かんたんなゲームを作りながらプログラミングの<sup>きほん</sup>基本を学ぼう

## □ タイミングゲーム！

- LEDがすべて光っているときにAボタンを押すとクリア<sup>お</sup>
- クリアするまでのボタンを押す回数<sup>お</sup>で競う<sup>きそ</sup> (少ない方が勝ち)



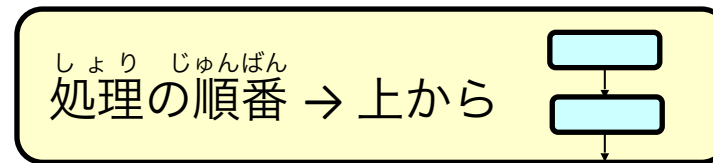
クリアすると「✓」マーク  
失敗すると「x」マーク

動画



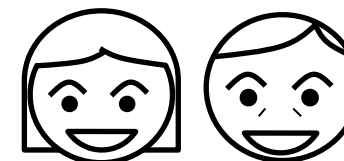
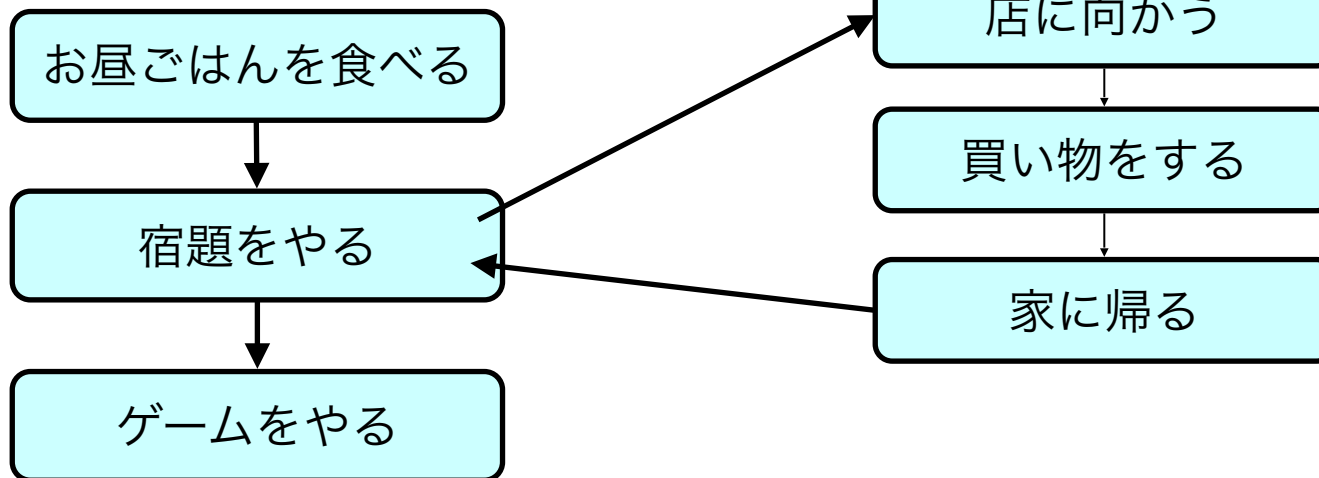
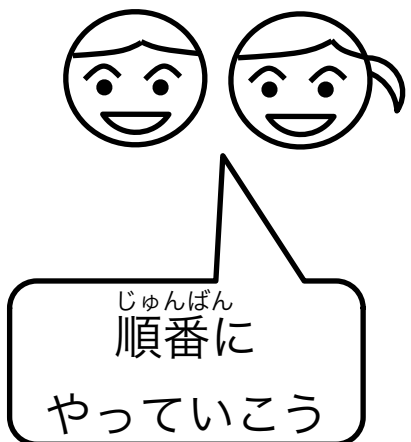
# プログラミングの<sup>きほん</sup>基本①: <sup>しより じゅんばん</sup>処理の順番

- <sup>じゅんばん</sup>上から順番に実行される
- イベントが発生すると<sup>しより</sup>処理の流れが<sup>うつ</sup>移ることも



もともとのプラン

急におつかいをたのまれると...



おつかい行っ  
てきて～

# しより micro:bitの処理の流れ



このブロックの中は、**でんげん**電源を  
入れたときに**一回だけ**実行



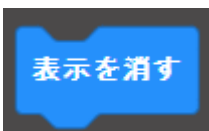
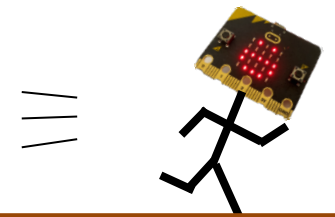
このブロックの中は  
**ずっと**くり返される



そのほかの四角いブロックの中は、  
**たいおう**対応する**イベント**が起こった**ときのみ**実行

しよりそくど  
コンピュータの**処理速度**は**とても速い**！

多くの処理は、**人間にとって一瞬**！  
適切に「一時停止ブロック」を入れよう！



- 0.1秒
- 0.2秒
- 0.5秒
- 1秒
- 2秒
- 5秒

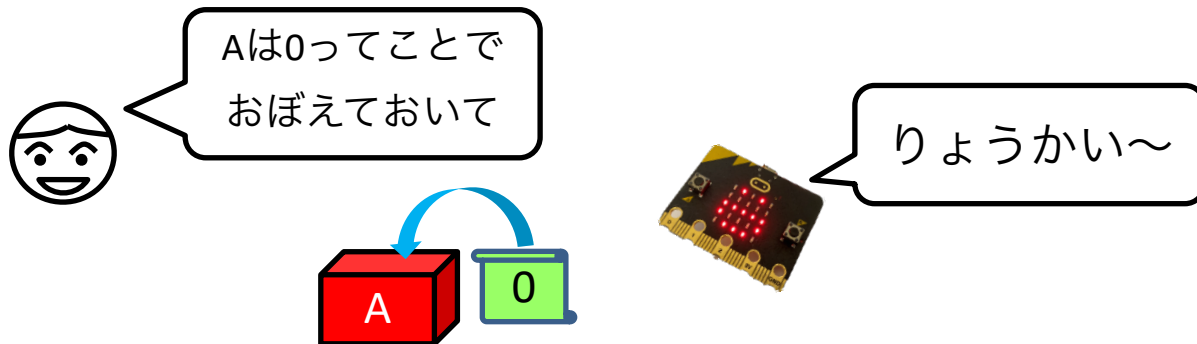
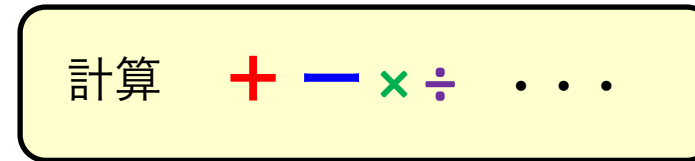
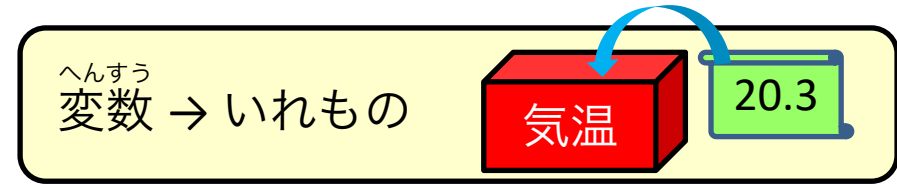
# プログラミングの<sup>きほん</sup>基本②: <sup>へんすう</sup>変数と計算

## □ <sup>へんすう</sup>変数とは？

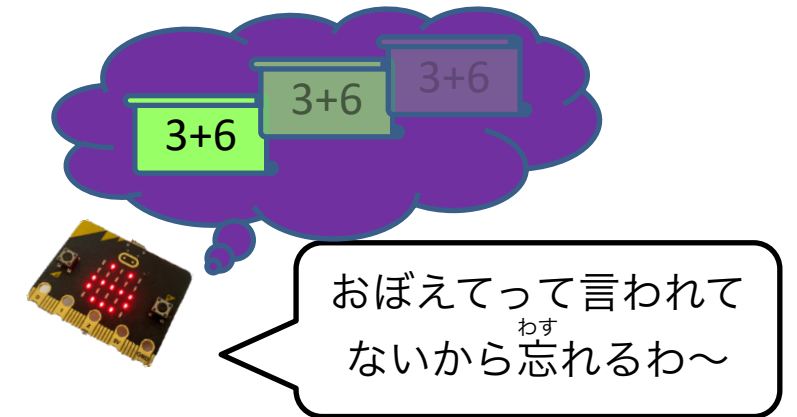
- データのいれもの (データ: <sup>すうち</sup>数値、文字など)
- 好きな名前をつけられる

## □ どんな計算ができる？

- 足し算、引き算、かけ算、わり算など

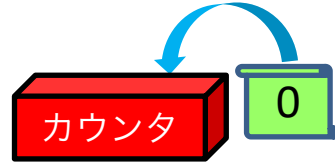


<sup>へんすう</sup>変数に入れないと...



# へんすう 変数と計算のブロック

へんすう  
変数にデータをセットするとき



へんすう  
変数を使うとき

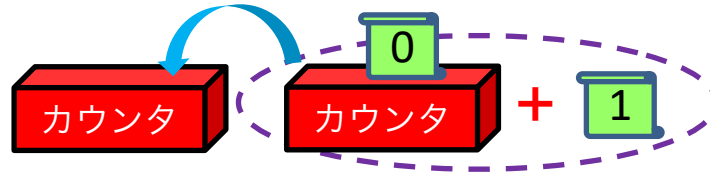
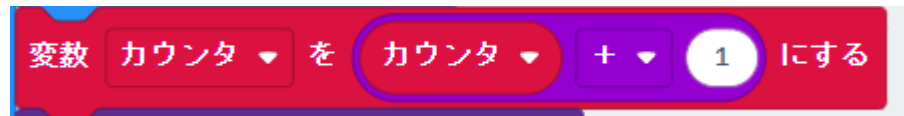


計算するとき



計算式にへんすうも入れることができる

組み合わせると...



“カウンタ”の中身が0から0+1(=1)に変化

けっか  
計算結果で“カウンタ”変数を書きかえる

# プログラミングの<sup>きほん</sup>基本③: ループと<sup>じょうけんぶんき</sup>条件分岐

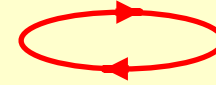
## □ ループとは？

- 同じ<sup>しより</sup>処理を何回もくり返すこと

## □ <sup>じょうけんぶんき</sup>条件分岐とは？

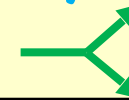
- 設定した<sup>じょうけん</sup>条件によって<sup>しより</sup>処理を変えること

ループ → くり返し



<sup>じょうけん</sup>条件 → もし...だったら ?

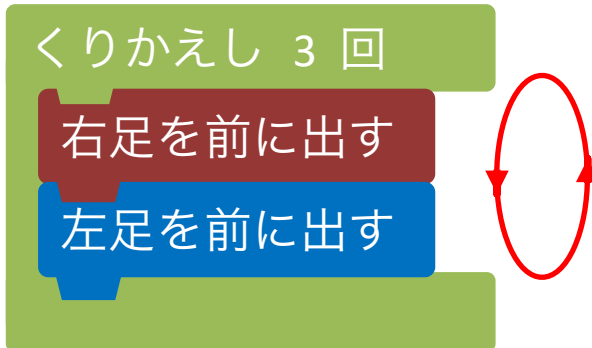
<sup>ぶんき</sup>分岐 → <sup>えだわ</sup>枝分かれ



## ループの<sup>れい</sup>例

6歩 歩く

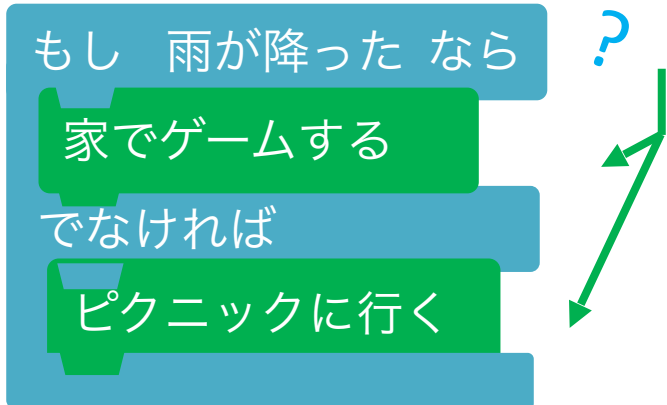
右足を前に出す  
左足を前に出す  
右足を前に出す  
左足を前に出す  
右足を前に出す  
左足を前に出す



## 条件分岐の<sup>れい</sup>例

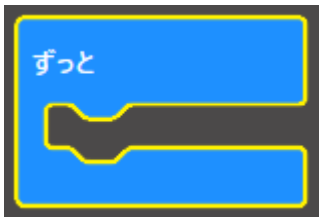
休日の予定

もし雨が降<sup>ふ</sup>ったら  
家でゲームする  
そうじゃなかったら  
ピクニックに行く



# ループのブロック

## ループのブロックの例<sup>れい</sup>



micro:bitの電源<sup>でんげん</sup>をONにしてからずっと中の処理<sup>しより</sup>をくり返す



ここで決めた回数、中の処理<sup>しより</sup>をくり返す



条件<sup>じょうけん</sup>が成り立っているときだけループが回る

# じょうけんぶんき 条件分岐のブロック

## じょうけん 条件を表すブロックの例



左と右のデータが同じか？



左のデータの方が右のデータより小さいか？



ボタンが押されているか？



じょうけん  
条件を組み合わせることもできる

## じょうけんぶんき 条件分岐のブロック

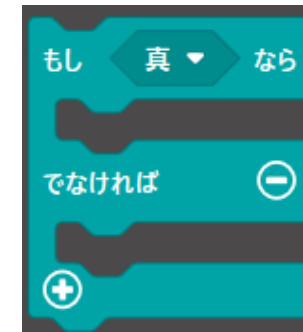
ほそく  
【補足】

← MakeCodeでは「条件判断」と呼んでいる

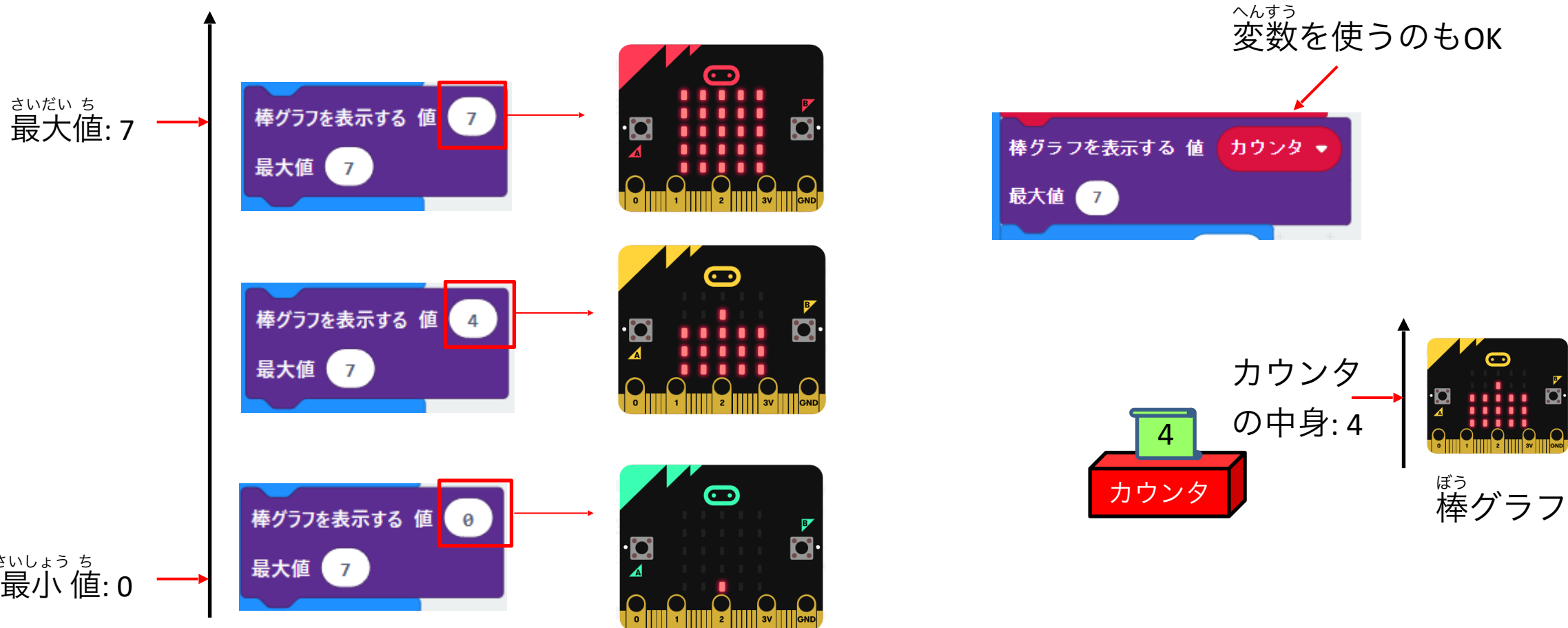


じょうけん  
条件が成り立っている場合、  
中のしよりに  
処理を行う

ここをおすとじょうけんふ  
条件が増やせる



# よびちしき (予備知識) ぼう LED棒グラフの使い方



さいだい値 0~最大値までの数字をLEDの光る量で表現  
りょうひょうげん



# ちしき ここまでの知識を使ってゲームを作ってみよう

動画

The screenshot shows the Microsoft MakeCode for micro:bit editor interface. On the left, there is a virtual micro:bit board with a grid of red LEDs. Below the board are buttons for 'コンソールを表示' and 'シミュレーター'. A search bar and a category menu are also visible. The main workspace contains a script with the following blocks:

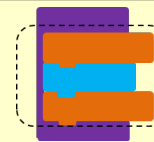
- 変数 カウンター を 0 にする
- くりかえし 8 回
- 棒グラフを表示する 値 カウンター
- 最大値 7
- 一時停止 (ミリ秒) 100
- もし ボタン A が押されている なら
- もし カウンター = 7 なら
- アイコンを表示 [Bar Graph]
- でなければ
- アイコンを表示 [Dotted Matrix]
- 変数 カウンター を カウンター + 1 にする

At the bottom, there is a 'ダウンロード' button and a 'タイピングゲーム' project name.

# プログラミングの<sup>きほん</sup>基本④: <sup>かんすう</sup>関数

## □ <sup>かんすう</sup>関数とは？

<sup>かんすう</sup>関数 → まとめる

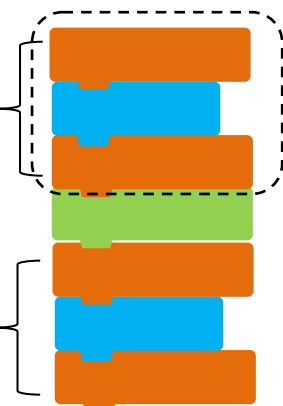


➤ <sup>きのう</sup>機能ごとにプログラムをまとめ、使いやすく・わかりやすくするもの

同じようなプログラムを何度も作るのはめんどくさい！



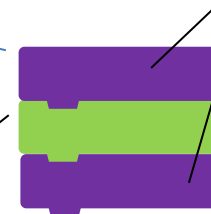
同じ<sup>しより</sup>処理



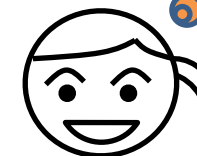
<sup>かんすう</sup>関数



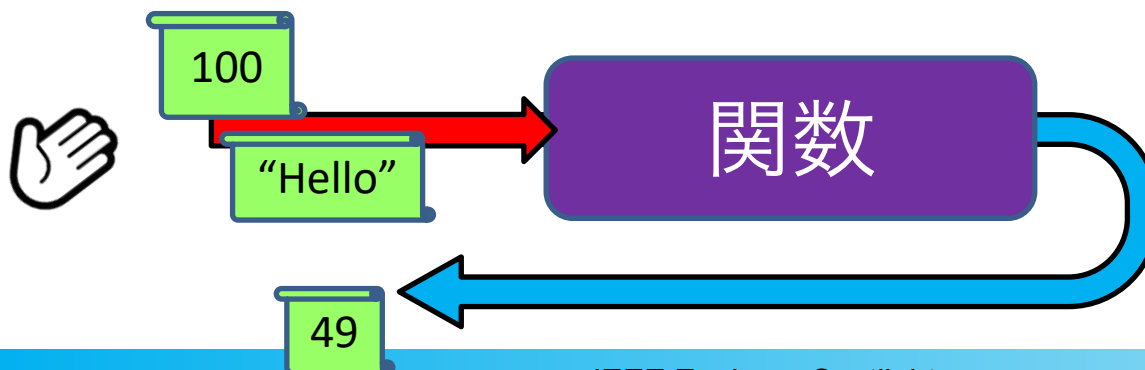
<sup>かんすう</sup>関数を<sup>さいりよう</sup>再利用



関数でまとめて、プログラムがすっきり！



データを<sup>かんすう</sup>関数に<sup>わた</sup>渡したり、<sup>かんすう</sup>関数からデータを<sup>かえ</sup>返したりできる (渡す / 返すデータがなくてもOK)



<sup>しより</sup>処理内容は同じだが、<sup>しより</sup>処理するデータ自体は変えたい  
というときに<sup>べんり</sup>便利

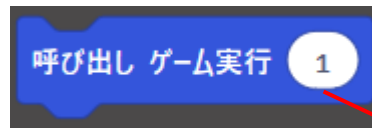
# かんすう 関数を使ってゲームの機能<sup>きのう</sup>をまとめよう

## かんすう 関数のブロック



かんすう  
関数の名前

かんすう わた  
関数に渡すデータ  
(関数の中では<sup>へんすう</sup>変数として扱う)



かんすう じっさい  
関数を実際に使うときのブロック

かんすう わた  
関数に渡すデータ

## 予備知識<sup>よびちしき</sup> (ちょっと難しい)<sup>むずか</sup>

しんぎち  
真偽値



じょうけん  
「条件が成り立っている」  
ことを表すデータ



じょうけん  
「条件が成り立っていない」  
ことを表すデータ

へんすう  
変数に入れることもできる



# かんすう 関数を使ってゲームの機能をまとめよう

動画

The screenshot shows the Microsoft MakeCode for micro:bit editor interface. The main workspace contains a script for a timing game. The script is organized into three sections: '最初だけ' (Initially), 'ずっと' (Forever), and '関数 ゲーム実行' (Function Game Execution). The '最初だけ' section contains a '文字列を表示' (Show text) block with 'Level1'. The 'ずっと' section contains a loop with '文字列を表示' blocks for 'Level2' and 'Level3'. The '関数 ゲーム実行' section contains a function block with the following logic: clear the '変数 クリア' (Variable Clear) block, set '変数 カウンター' (Variable Counter) to 0, loop 8 times to show a bar graph and stop for a time delay, then check if 'ボタン A' (Button A) is pressed. If pressed, check if the counter is 7, show an icon, clear the variable, and increment the counter. If not pressed, show an icon and increment the counter.

# プログラミングの<sup>きほん ふくしゅう</sup>基本の復習

- <sup>へんすう</sup>変数とは？  のいれもの
- ループとは？  のくり返し
- <sup>じょうけんぶんき</sup>条件分岐とは？  によって<sup>しより</sup>処理を変えること

A

- (1): プログラム
- (2): 同じ<sup>しより</sup>処理
- (3): 計算結果

B

- (1): データ(数値や文字など)
- (2): 異なる<sup>こと</sup> <sup>しより</sup>処理
- (3): 設定<sup>せってい</sup>した<sup>じょうけん</sup>条件

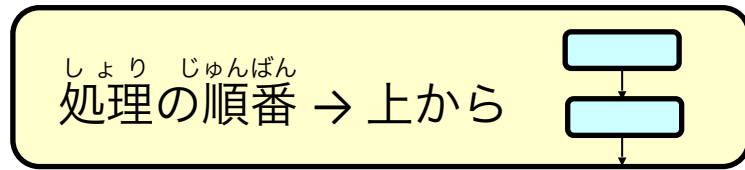
C

- (1): データ(数値や文字など)
- (2): 同じ<sup>しより</sup>処理
- (3): 設定<sup>せってい</sup>した<sup>じょうけん</sup>条件

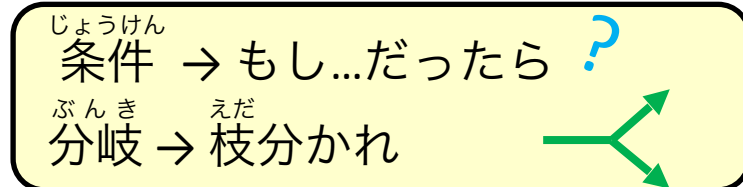
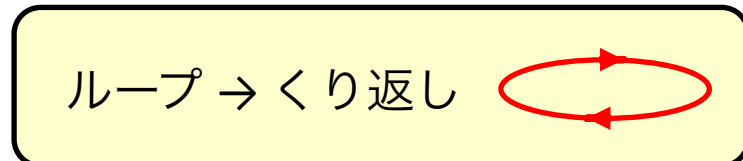
<sup>てきせつ</sup>適切な組み合わせを<sup>えら</sup>選んでください！

# プログラミングの<sup>きほん</sup>基本のまとめ

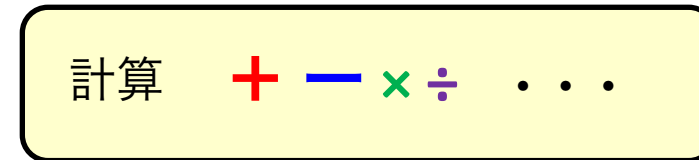
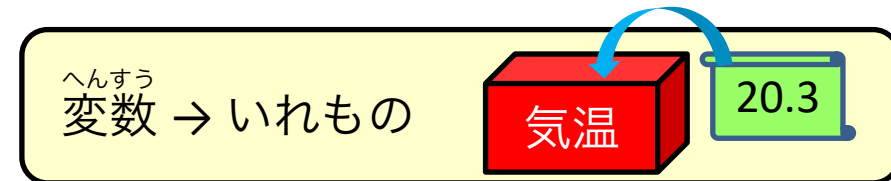
## 1. <sup>しより じゅんばん</sup>処理の順番



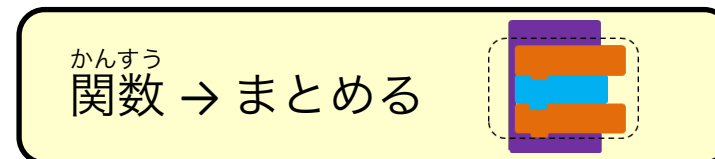
## 3. <sup>じょうけん ぶんき</sup>ループと条件分岐



## 2. <sup>へんすう</sup>変数と計算



## 4. <sup>かんすう</sup>関数



# おうようへん 【応用編】 ゲームをさらにアレンジしてみよう

例えば...

- いまのプログラムでは、何回スイッチを<sup>お</sup>押したかを人間が<sup>かくにん</sup>確認している  
ので、これもコンピュータにやらせる
- 各レベルでクリアまでにスイッチを<sup>お</sup>押した回数から<sup>さいしゅうてき</sup>最終的なスコアを計算する
- クリアしたときに音を出す

など

プログラミング例は<sup>れい</sup>参考資料<sup>さんこうしりょう</sup>を見てください。  
電池をつなぐ場合<sup>じょうほう</sup>の情報<sup>の</sup>も載せています。

2

でんしこうさく たいけん  
電子工作を体験！

かいへい  
～自動開閉パーティションを作って学ぼう～



# でんあつ でんりゅう ていこう 電圧・電流・抵抗

## □ でんあつ 電圧

➤ 電気を流そうとする力の量<sup>りょう</sup>

## □ でんりゅう 電流

➤ 流れる電気の量<sup>りょう</sup>

## □ ていこう 抵抗

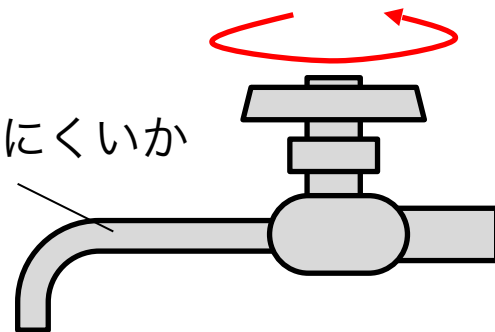
➤ どれくらい電気が流れにくいという量<sup>りょう</sup>

ほそく  
【補足】 単位について  
でんあつ  
電圧: V (ボルト)  
でんりゅう  
電流: A (アンペア)  
ていこう  
抵抗: Ω (オーム)

## イメージ

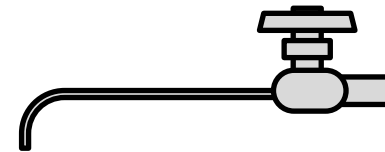
でんあつ  
電圧: どれくらい蛇口をひねるか  
じゃぐち

ていこう  
抵抗: どれくらい水が流れにくい  
か

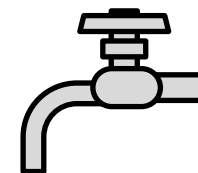


でんりゅう  
電流: どれくらい水が流れる  
か

ていこう  
抵抗が大きい

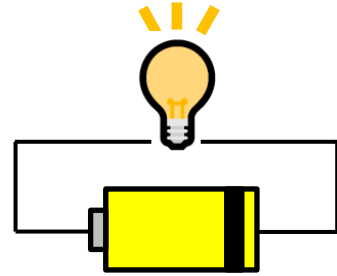


ていこう  
抵抗が小さい



# でんしかいる 電子回路ってなんだ？

かいる  
回路: 電気の通り道



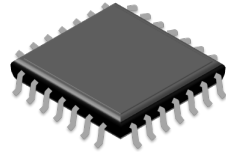
かいる  
回路を作ると電気が流れる

でんしかいる いっぱんてき  
電子回路: 一般的には半導体という部品を使った回路のこと

はんどうたい

かいる

ダイオード、トランジスタ、LEDなど



はんどうたい  
マイコンも半導体！

今日はこれだけわかればOK

でんしかいる  
電子回路を作る → いろいろな部品をつなげて、電気の通り道を作る

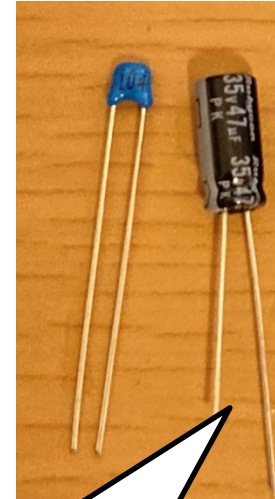
# 部品の例<sup>れい</sup>

ていこうき  
抵抗器



カラーコード: 色で<sup>ていこうち</sup>抵抗値がわかる

コンデンサ



LED

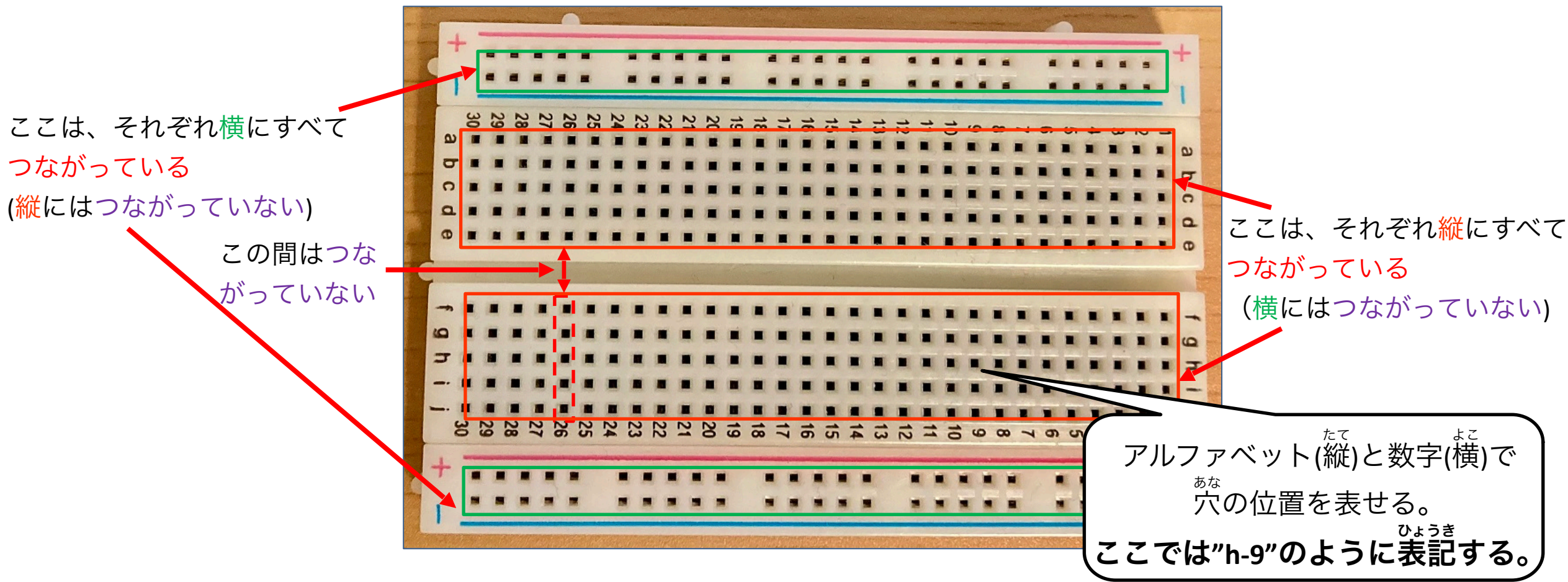


取り付ける向きがある部品  
もあるので気をつけよう！

# ブレッドボードでかんたんにかいる回路を作成！

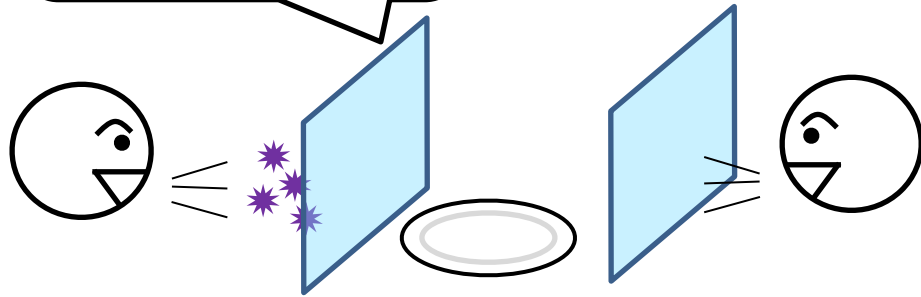
## □ ブレッドボードとは？

- かんたんに部品の抜き差しができ、回路の試作によく使われる



# 作るもの: 自動開閉パーティション かいへい

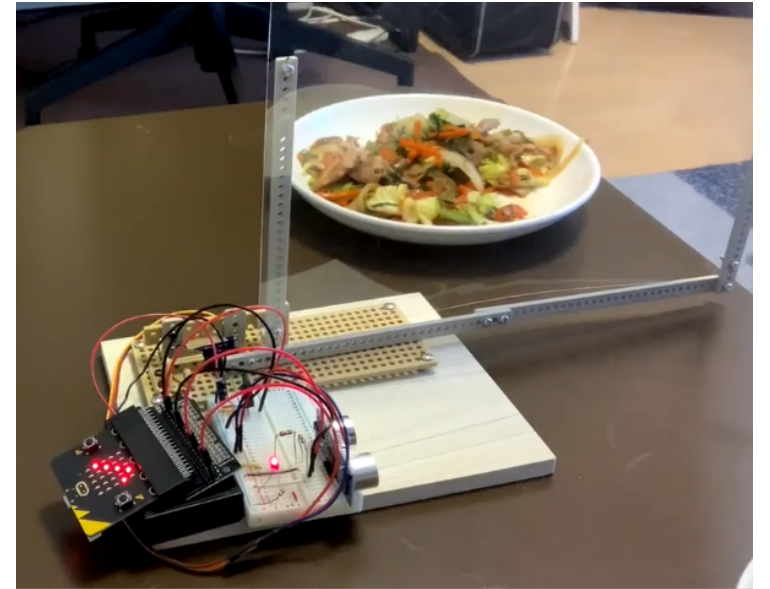
パーティション  
ひまつかんせん  
で飛沫感染を  
ぼうし  
防止!



いちいちパーティションをどけるの  
もめんどうだ・・・  
ひんぱんに触るのもよくない・・・



ひせつしよく かいへい  
非接触で開閉できる  
パーティションを作  
ろう!



# よびちしき 予備知識

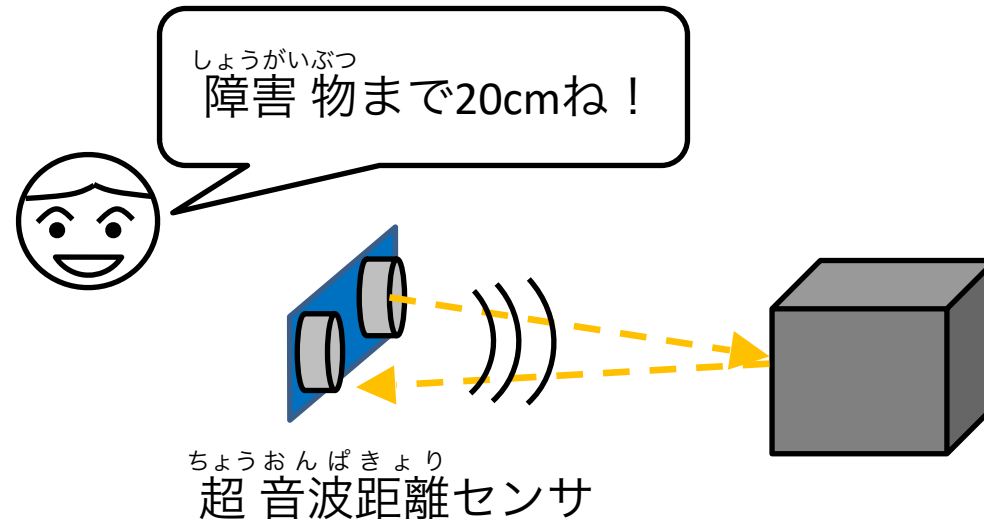
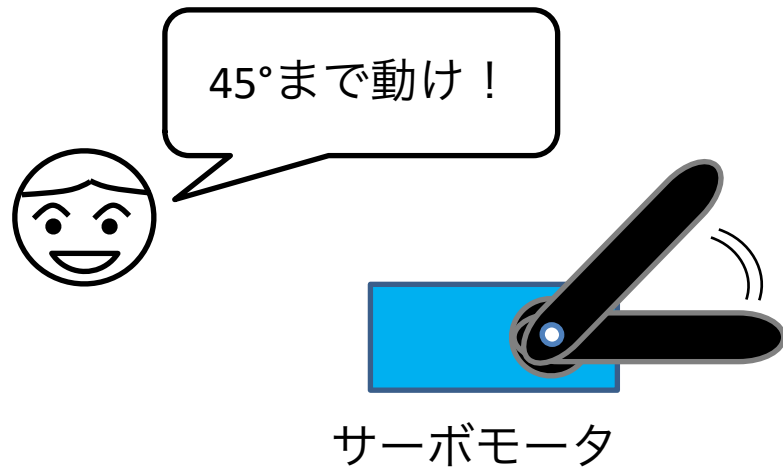
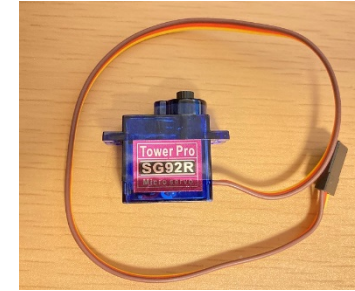
## □ この工作で使う重要な部品<sup>じゅうよう</sup>

### ➤ サーボモータ

- 位置・角度制御が得意なモータ<sup>いち かくどせいぎょ とくい</sup>

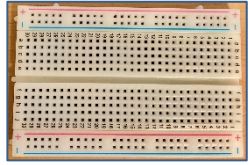
### ➤ 超音波距離センサ<sup>ちょうおんぱきより</sup>

- 超音波の反射時間によって距離を測定できるセンサ<sup>ちょうおんぱ はんしゃ きより そくてい</sup>



# サーボモータと超音波距離センサを動かす回路を作ってみよう

## ここで使うもの



ブレッドボード



ていこうき  
10kΩ抵抗器  
(カラーコード: 茶黒橙金)



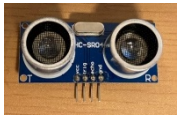
ていこうき  
20kΩ抵抗器  
(カラーコード: 赤黒橙金)



ていこうき  
330Ω抵抗器  
(カラーコード: 橙橙茶)



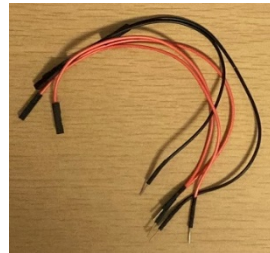
赤色LED 電池ボックス  
(単三3本)



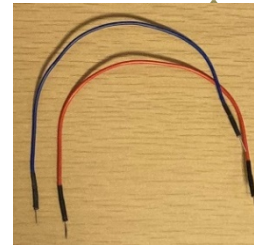
ちょうおんぱきより  
超音波距離センサ



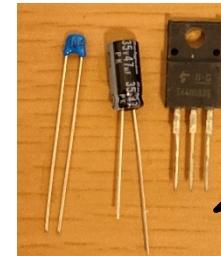
サーボモータ



オス-メスジャンパー線  
(5本)



オス-オスジャンパー線  
(2本)

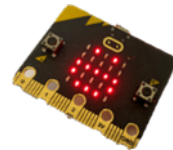


さんたんし  
三端子レギュレータ  
ふぞく  
と付属のコンデンサ

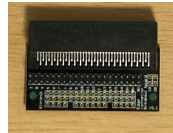
micro:bitに必要な  
でんあつ  
な電圧を作る



ジャンパー線セット



micro:bitと  
ブレークアウトボード



ちゅうい かいるさくせい  
【注意】回路作成は十分に気をつけよう！

かいる まちが  
回路を間違えると、部品はかんたんに壊れる。

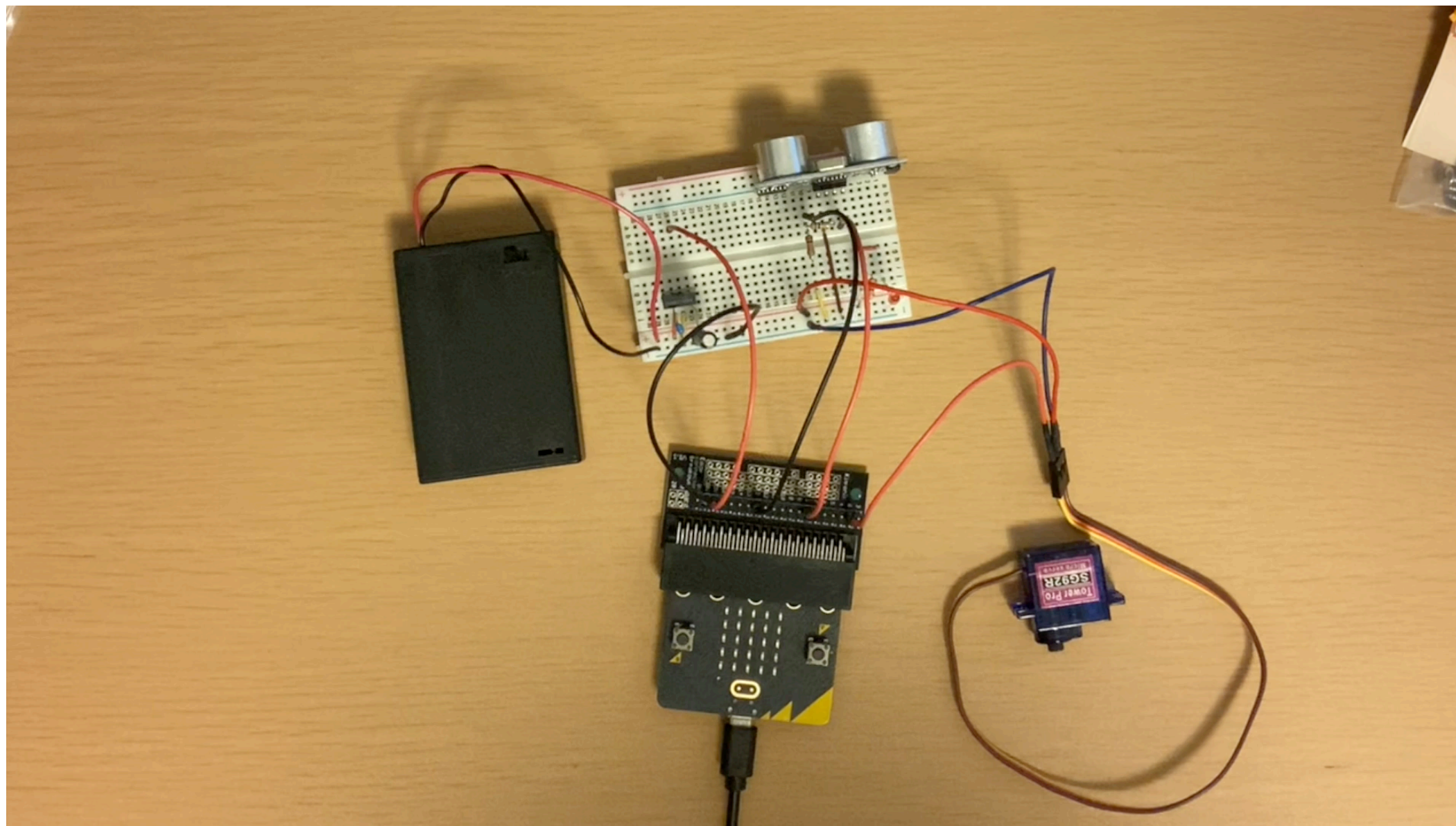
少しでもおかしいと思ったらすぐに電源を切って、回路をチェック！

サーボモータと超音波距離センサーを動かす回路を作ってみよう

ちょうおんぱきより

かいる

動画





# ちょうおんぱきより サーボモータと超音波距離センサを動かすプログラム

動画

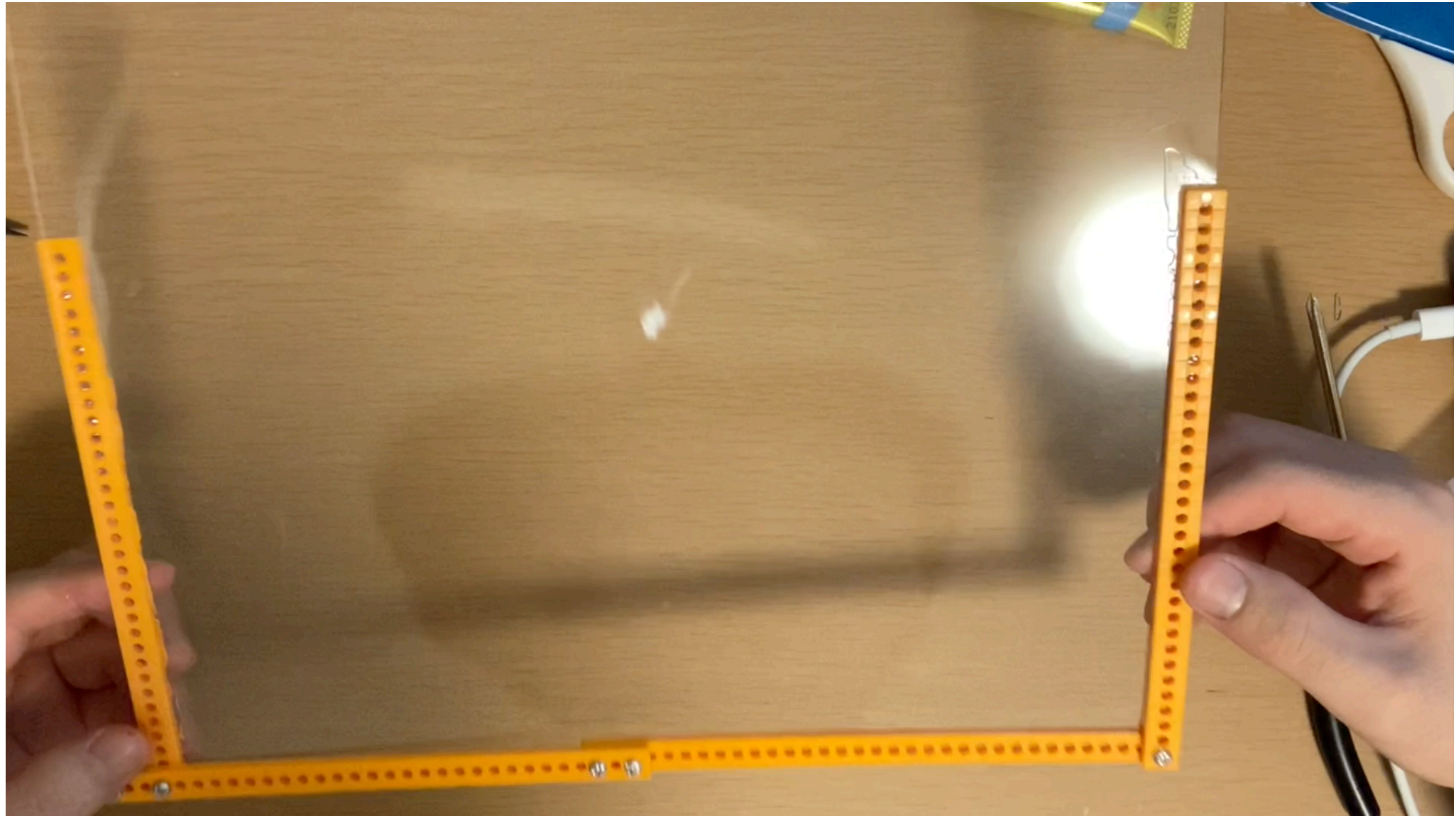
The screenshot shows the Microsoft MakeCode for micro:bit editor interface. On the left, a virtual micro:bit board is shown with a blue ultrasonic sensor connected to pins 0, 1, and 2, and a blue servo motor connected to pins 3V and GND. The central panel displays a block-based program:

- ずっと** (Forever) loop:
- ping trig P1** block with **変数 距離** (variable distance) and **unit cm**.
- echo P2** block with **にする** (set to).
- もし 距離 < 10 なら** (if distance < 10 then) conditional block:
- サーボ 出力する 端子 P0 角度 90** (servo output pin P0 angle 90) block.
- 一時停止 (ミリ秒) 500** (wait 500 ms) block.
- サーボ 出力する 端子 P0 角度 180** (servo output pin P0 angle 180) block.
- 一時停止 (ミリ秒) 500** (wait 500 ms) block.

The bottom of the editor shows a **ダウンロード** (Download) button and a **自動パーティション** (Auto-partition) button. The status bar at the bottom left indicates the file name: **microbit-自動パーテ...hex**.

# パーティション用のアームを作る

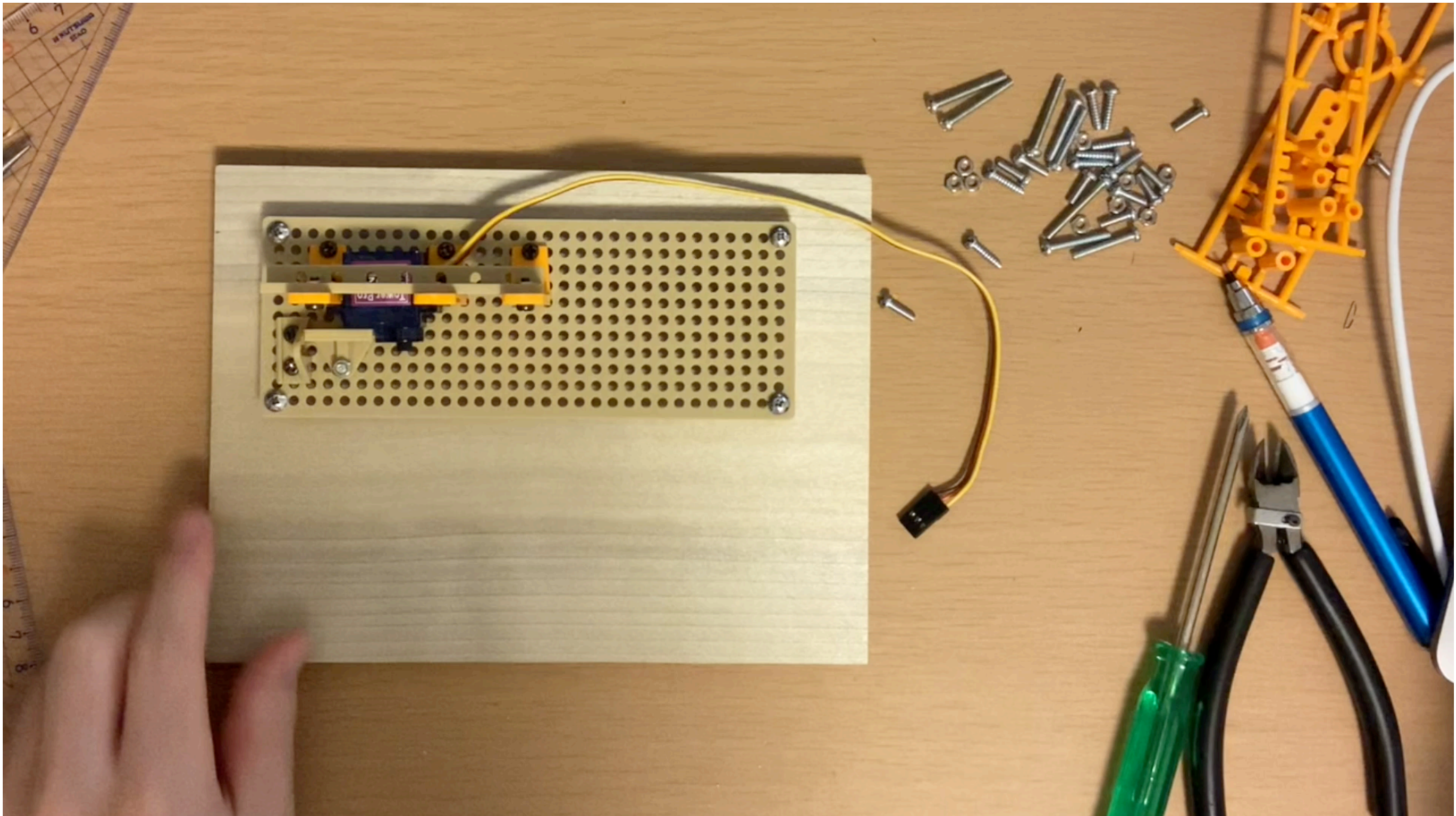
動画



パーティションの大きさは必要<sup>ひつよう</sup>に応じて調整<sup>おう</sup>してください。ちょうせい(このサイズより大きくはしないように)

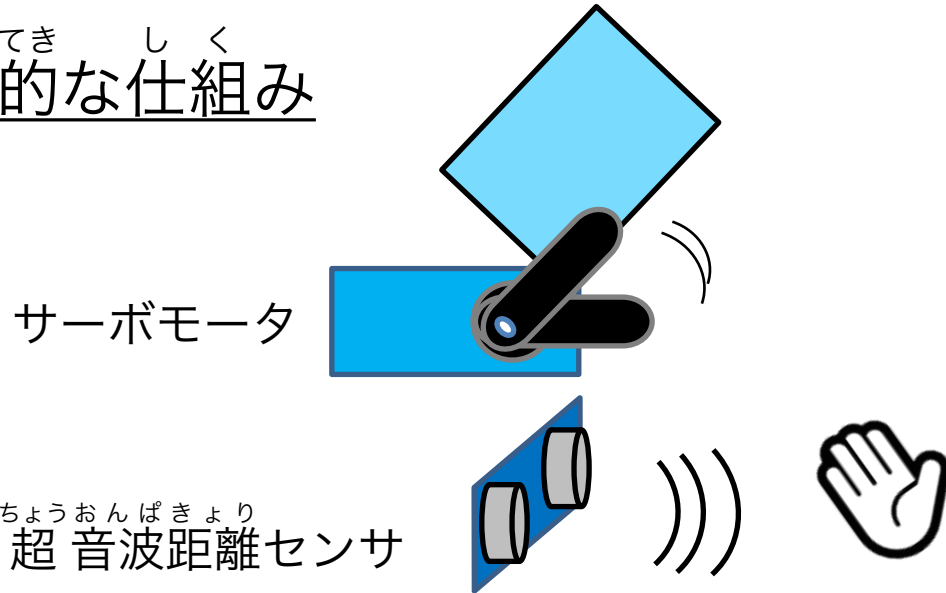
# 土台を作る

動画



# かいへい 自動開閉パーティションのプログラムを作る

## きほんてき し く 基本的な仕組み



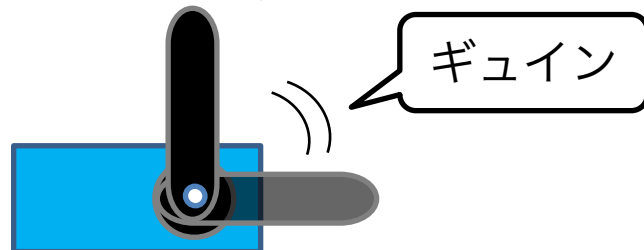
ちょうおんぱきより けんしゅつ  
超音波距離センサで近いものを検出

サーボモータを動かしてパー  
ティションを開ける

かんかく  
ある一定間隔で近くに何かあるかを  
確認し、なければサーボモータを動  
かし、パーティションを閉じる

ずっとくり返す

サーボモータはふつうに角度を  
与えると、動きが速すぎる



かんかく  
短い時間間隔で少しずつ動  
かし、速度を調整する



ここでは、90°~180°の間を  
5°ずつ約0.1秒ごとに動かす

# かいへい 自動開閉パーティションのプログラムを作る

動画

Microsoft MakeCode for micro:bit

検索...

基本

入力

音楽

LED

無線

ループ

論理

変数

計算

Sonar

高度なブロック

関数

配列

文字列

ゲーム

画像

入出力端子

シリアル通信

制御

最初だけ

アイコンを表示

サーボ 出力する 端子 P0 角度 90

アナログで出力する 端子 P0 値 0

ずっと

ping trig P1

変数 距離 を echo P2 にする

unit cm

もし 距離 < 10 なら

呼び出し 開く

もし 距離 < 10 ならくりかえし

一時停止 (ミリ秒) 5000

ping trig P1

変数 距離 を echo P2 にする

unit cm

呼び出し 閉じる

一時停止 (ミリ秒) 100

関数 開く

アイコンを表示

変数 角度 を 90 にする

くりかえし 16 回

変数 角度 を 角度 + 5 にする

サーボ 出力する 端子 P0 角度 角度

一時停止 (ミリ秒) 100

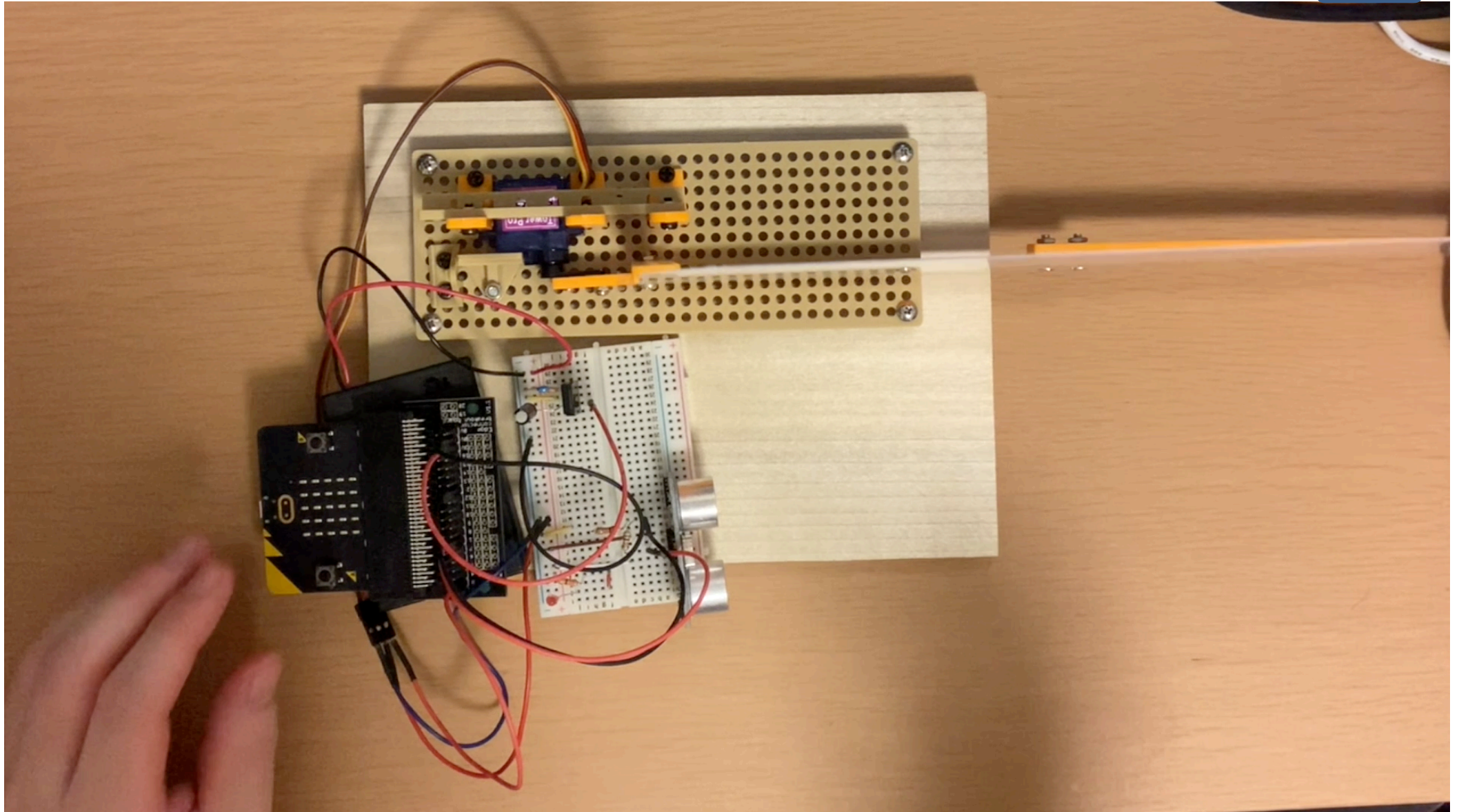
ダウンロード

自動パーティション

パーティションを開く距離は、必要に応じて変えてください。(例: 20cm)

# 仕上げ

動画



# ほそく 補足

## □ 動きがおかしいなと思ったら

- 回路の接続<sup>せつぞく</sup>をチェック (ジャンパー線が抜けていたりなど)
- 電池<sup>ちゆうかん</sup>を交換してみる

## □ サーボモータがガタつくときは

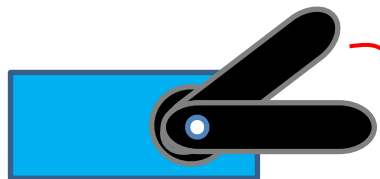
- 接着剤<sup>せつちやくざい</sup>を使ったり、間に厚紙<sup>あつがみ</sup>などを挟<sup>はさ</sup>んだりする

## □ 急に動くパーティションに注意

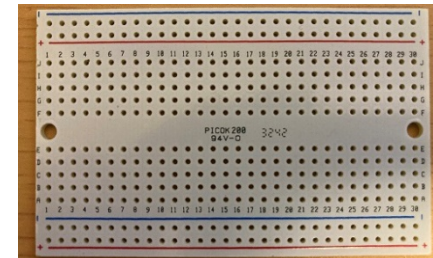
- パーティションが初期位置<sup>しよきいち</sup>でないところで電源<sup>でんげん</sup>を入れると、最初に初期位置<sup>しよきいち</sup>(90°)に戻ろうとするので、勢いよくパーティションが動いてしまう
- 必要に応じて一度パーティションをサーボモータから取り外すなどの対応<sup>たいおう</sup>をする
- 電源<sup>でんげん</sup>を切るときは、初期位置<sup>しよきいち</sup>に戻った状態<sup>もど じゆうたい</sup>で

## □ ブレッドボードだと配線<sup>はいせん</sup>が抜けやすいので、はんだ付けしてしまうのも手

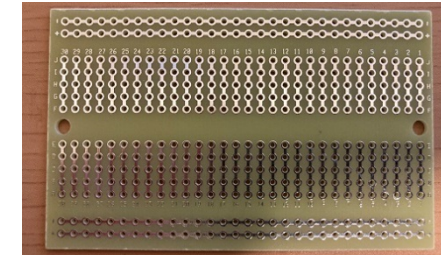
- ブレッドボードと同じ配線<sup>はいせん</sup>の基板<sup>きばん</sup>もある



電源<sup>でんげん</sup>を入れた瞬間<sup>しゆんかん</sup>、勢いよくここに<sup>もど</sup>戻ろうとする



表面

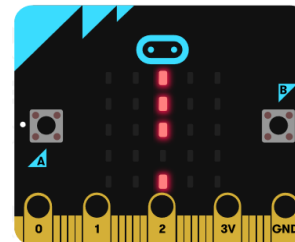
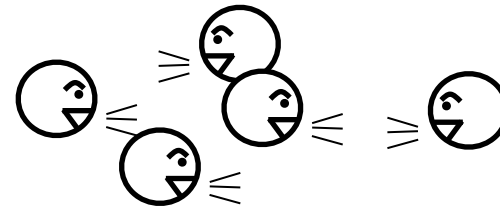
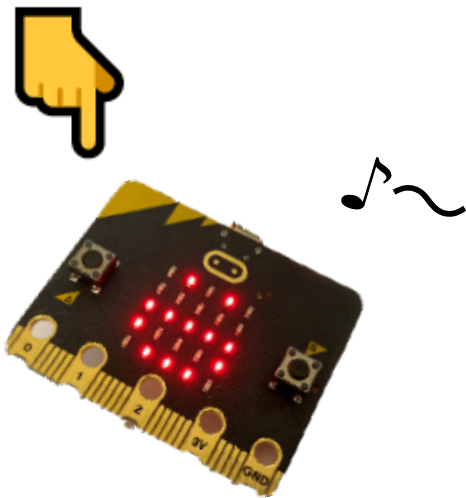


裏面

ブレッドボードと同じ配線<sup>はいせん</sup>のユニバーサル基板<sup>きばん</sup>

# オリジナルの工作をするときの発想法の例

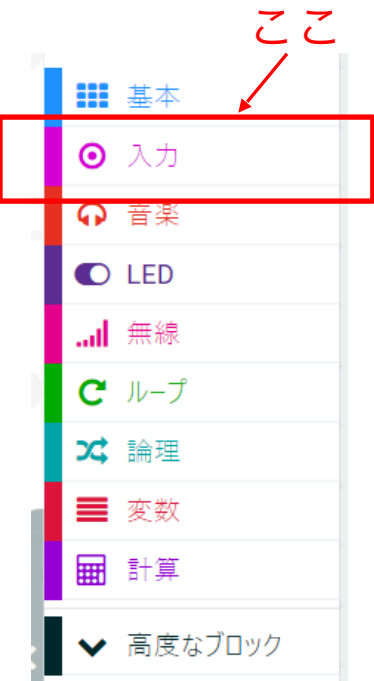
- イベントとその結果(処理)を考えよう！
  - 例1: イベント「ボタンAが押されたとき」 → 結果「音を鳴らす」
  - 例2: イベント「まわりの音がうるさくなったとき」 → 結果「LEDでアイコンを表示する」



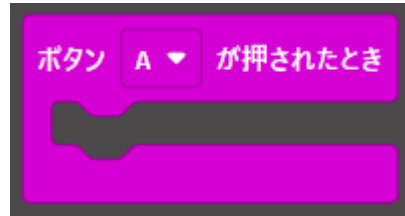


# micro:bitでどんなことができるか

## イベントの例

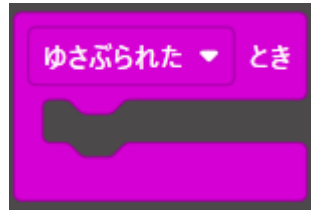


### ボタン入力



Bボタンや、AとBの同時押し  
(A+B)も選択できる

### micro:bitの動きや傾き

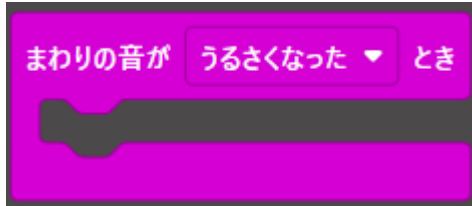


傾きなども条件に設定できる

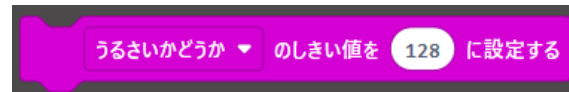


など

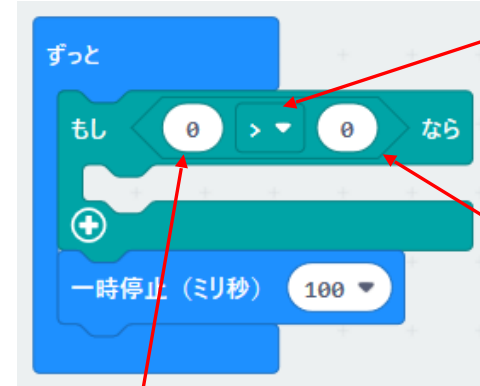
### 音量 (micro:bit V2のみ対応)



「静かになったとき」も選択できる  
どれくらいでうるさいと判断するか  
はこのブロックで設定できる



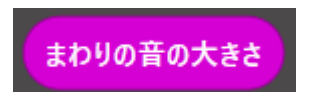
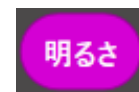
### 条件分岐と組み合わせる



条件を選択

適切な数値  
を設定

ここに以下のような  
データを入れる



micro:bit V2のみ対応

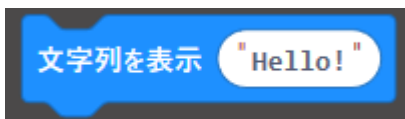
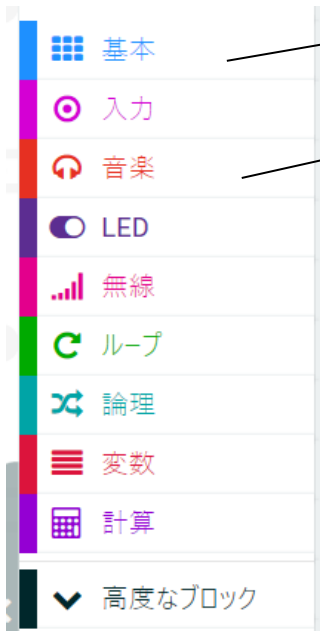
# micro:bitでどんなことができるか

## しより れい 処理の例

### ひょうじ LED表示

### な 音を鳴らす

自分だけのメロディーを作ってみよう！



クリックした部分のLEDが光る



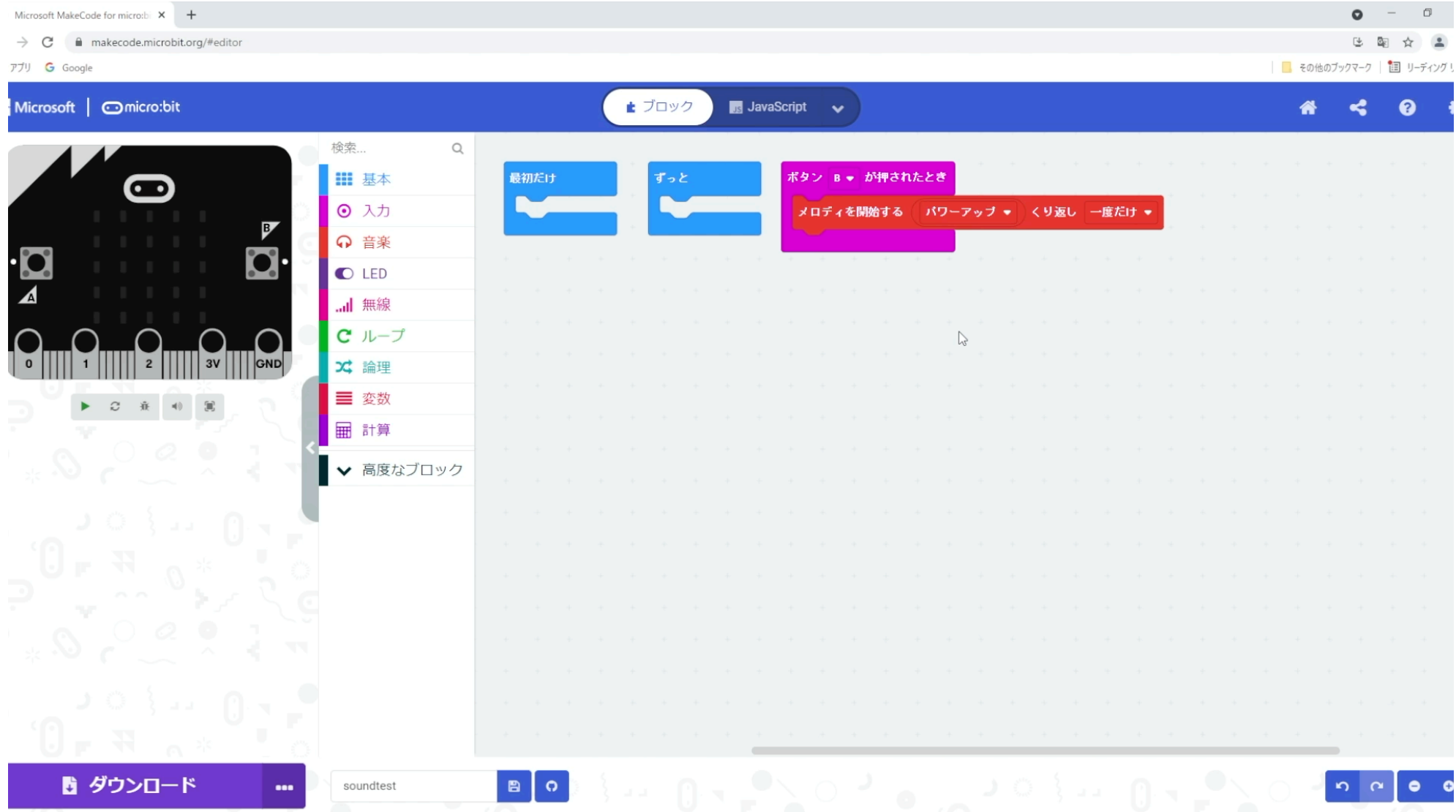
- ✓ ダダダム
- ジ・エンターテイナー
- プレリュード
- 歓喜の歌
- ニャン・キャット
- 着信メロディー
- ファンク
- ブルース
- ハッピーバースデー
- ウェディング・マーチ

さまざま  
様々なメロ  
ディーがある



micro:bit V2では、そのまま音を鳴らせる  
V1の場合は参考資料を参照

# な れい 音を鳴らす例



micro:bitでできることは、ここで紹介しきれないぐらいたくさんある  
しょうかい  
IEEE Engineer Spotlight

3

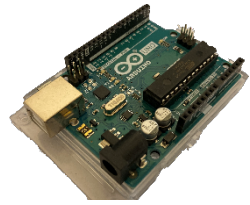
じょうきゅうへん

【上級編】

Arduinoを使ったLED

てんめつじっけん

点滅実験



# Arduinoとは

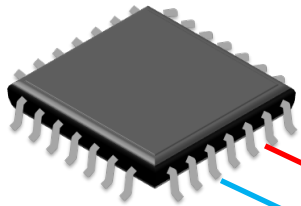
## マイコンボードの一つ

- 様々な電子部品を制御できる

micro:bitとは搭載している  
コンピュータが違う

## プログラミングはC言語を使う

```
void setup() {
  pinMode(13, OUTPUT); // 13番ピンを出力に設定
}
void loop() {
  digitalWrite(13, HIGH); // 13番ピンをHIGH(5V)に
  delay(100); // 0.1秒一時停止
  digitalWrite(13, LOW); // 13番ピンをLOW(0V)に
  delay(100); // 0.1秒一時停止
}
```



マイコンは端子の  
電圧を制御できる

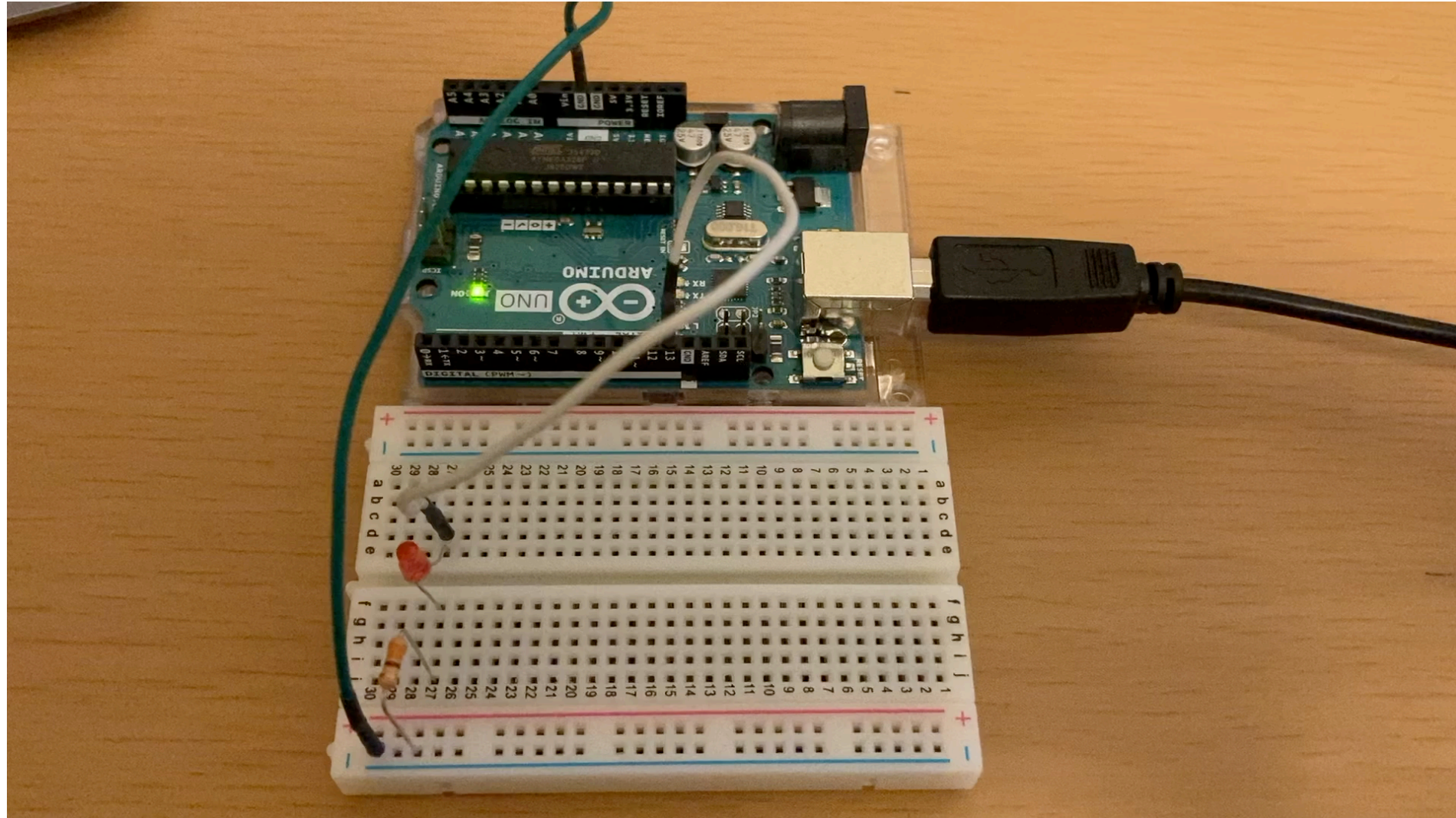
5V (HIGH)  
0V (LOW)

### IEEE TryEngineering

- IEEEが提供するオンライン学習コンテンツ
- Arduinoを使ったLED点灯実験も掲載
  - <https://tryengineering.org/teacher/arduino-blink-challenge/>

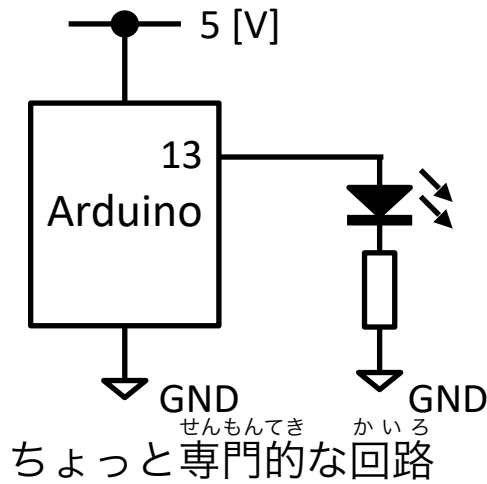
# てんめつじっけん LEDの点滅実験

プログラミング方法などは参考資料を参照  
(Webにも様々な情報が載っています。)

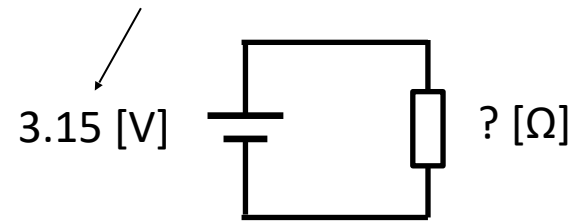


# かいるせっけい たいけん 回路設計を体験！

- 中学校の理科(物理)で習う「**オームの法則**」を使ってLEDの点灯回路を設計する



なぜこの電圧かは参考資料を参照



LEDの影響を考慮したシンプルな回路 (端子から5V出力時)



LEDに流れる電流は10 [mA]ほどにしておきたい。  
抵抗値はどのようにすればよいか？

$$\text{オームの法則: 抵抗}(R) = \text{電圧}(V) / \text{電流}(I)$$

$$R = 3.15 \text{ [V]} / 0.010 \text{ [A]} = 315 \text{ [}\Omega\text{]}$$

⇒ 近い値の330 [Ω]を選択

なぜ10 [mA]？

→ LEDやArduinoに流せる最大の電流を  
超えない&明るすぎない程度に

4

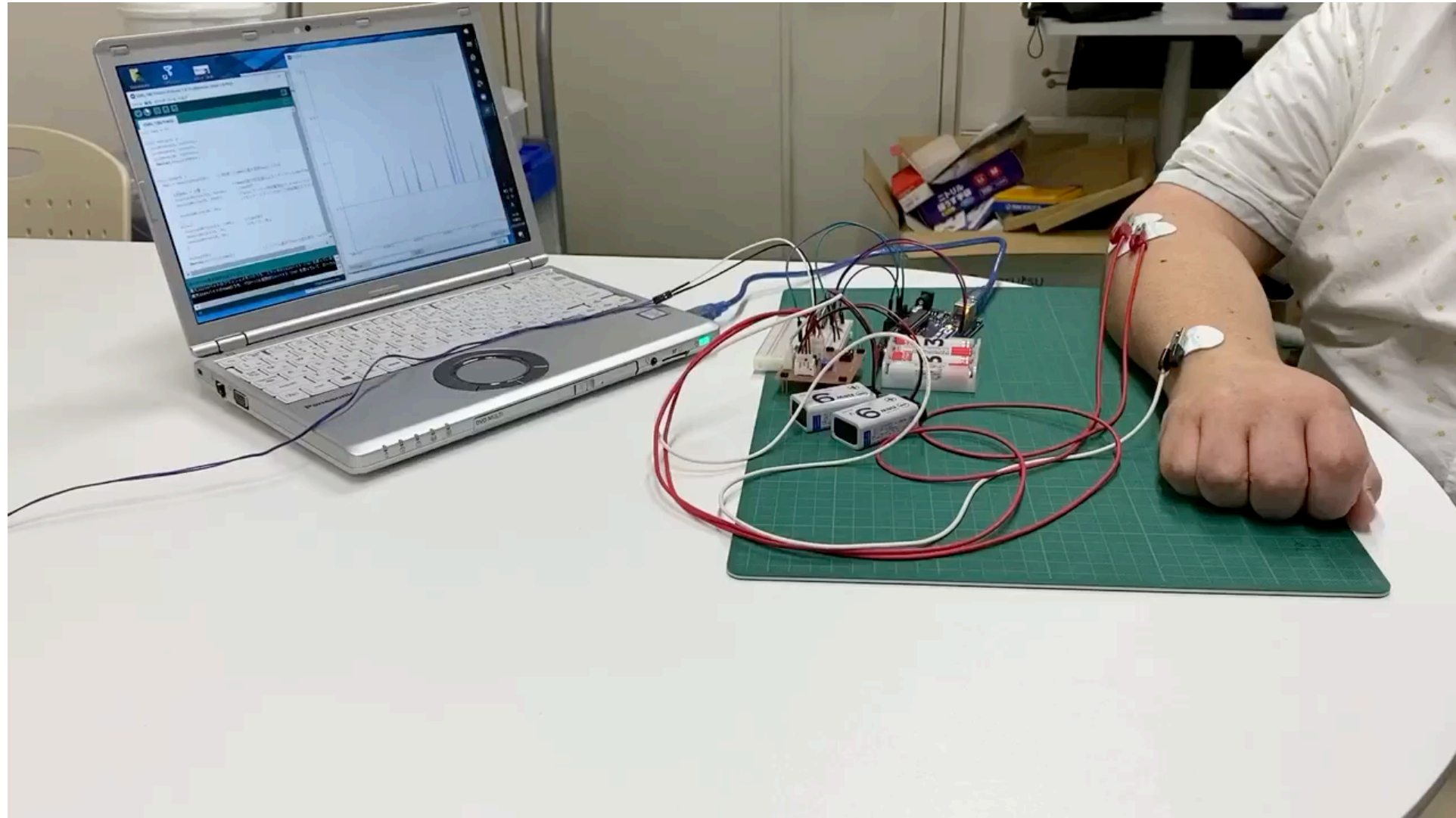
# Arduinoの応用例 紹介

おうようれいしょうかい



# 体の電気信号しんごうを使ってムカデロボットを操作そうさ！

とうきょう とりつさんぎょうぎじゅつこうとうせんもんがっこう いりょうふくしこうがく よしだたかし ていきょう  
東京都立産業技術高等専門学校 医療福祉工学コース 吉田 嵩 先生からのご提供



動画

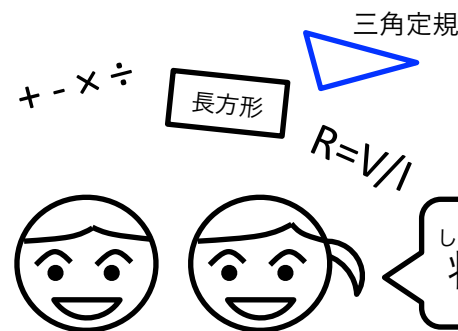
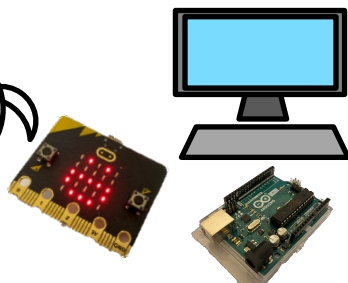
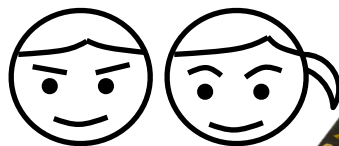
5

おわりに

# おわりに

- みなさんのアイデア<sup>じっげん</sup>実現のお手伝いするため、<sup>じっさい</sup>実際のプログラミングや<sup>れい</sup>工作例を見せながらみなさんの「できること」を増<sup>ふ</sup>やしました
- ぜひ、この<sup>こうえんしりょう</sup>講演資料や<sup>さんこうしりょう</sup>参考資料、<sup>はいしん</sup>オンデマンド配信の<sup>みかえ</sup>動画を見返して、<sup>りかい</sup>理解を<sup>ふか</sup>深めてください
- みなさんが<sup>なら</sup>学校で<sup>べんきょう</sup>習っている<sup>じゅけん</sup>勉強は、決して<sup>じゅけん</sup>テストや<sup>べんきょう</sup>受験のためだけのものではないと<sup>じっかん</sup>実感できたのでは？ <sup>しょうらい</sup>将来<sup>ちしき</sup>使える<sup>きょうみ</sup>知識として<sup>べんきょう</sup>興味を持って<sup>べんきょう</sup>勉強してくれるとうれしいです

おもしろそうだから  
作ってみよう！いろいろ  
調べてみよう！



<sup>しょうらい</sup>将来 使えるかもね！

# 第17回 IEEE Engineer Spotlight

## 【参考資料】

動く！光る！音が鳴る！アイデアをカタチにしよう！  
～今年の自由研究はプログラミングと電子工作だ！～

(株)東芝 研究開発センター / IEEE Tokyo Young Professionals Vice Chair

石垣 雄太郎

2021年7月24日

※本資料の無断転載を禁じます

※本講演のコンテンツ(プログラム例、工作例等)によって生じる損害の責任は負いかねます。ご同意の上、自己責任でご活用ください。

1

# micro:bitプログラミングやタイミングゲームに関する ほそくじょうほう する補足情報

# 画面の見方

ホームに戻る  
他のプロジェクトを開くときなど



シミュレータ  
micro:bit実物を動かさなくても、動作確認できる

ダウンロード  
プログラムができたからここからmicro:bitにダウンロード

ブロックリスト  
ここにあるブロックを組み合わせるプログラミングを行う

プログラミング画面  
ここにブロックを置いてプログラミングを行う

ブロックはドラック&ドロップで置く  
(マウスを左ボタンを押したまま、マウスを動かして、左ボタンから指をはなす)

【補足】  
つながっているブロックから一つだけブロックを動かしたいときは、キーボードのCtrlキーを押しながらドラック&ドロップする

戻るボタン  
クリックすると一つ前の状態に戻る (操作を間違えてしまったときなどに利用)

# タイミングゲームおうようへん応用編

## □ 以下の機能きのう ついかを追加

- クリアしたときや失敗しっばいしたときなどに音を鳴ならす
- クリアまでボタンを押おした回数をmicro:bitで数える
- クリアまでの回数でスコアを計算ひょうじして表示
  - スコアの計算式れいの例: Level1の回数×10 + Level2の回数×4 + Level3の回数
  - スコアが少ない方が勝ち

かんたんなレベルで失敗しっばいするとペナルティが大きくなるようにする

## □ 以下のURLからプログラムをコピーできる (やり方は次のページ)

- [https://makecode.microbit.org/\\_1uviKLM042vV](https://makecode.microbit.org/_1uviKLM042vV)

# プログラムのコピー方法

Google ChromeでURLを入力し、このページで「<sup>へんしゅう</sup>編集」をクリックすればコピーできる



各プログラムのURLは以下の通り

タイミングゲーム(講演で解説): [https://makecode.microbit.org/\\_hvuKFrPoDKL2](https://makecode.microbit.org/_hvuKFrPoDKL2)

タイミングゲーム応用編(前ページ): [https://makecode.microbit.org/\\_1uviKLM042vV](https://makecode.microbit.org/_1uviKLM042vV)

自動開閉パーティション(講演で解説): [https://makecode.microbit.org/\\_9TFAYsiC3Ehi](https://makecode.microbit.org/_9TFAYsiC3Ehi)

プログラムをコピーして、プログラムがうまくダウンロードできない場合は、一度ブラウザ(Google Chrome)を閉じて、再びMakeCodeでコピー後のプロジェクトを開き直すとうまくいく可能性がある。(MakeCodeのバグ?)



# タイミングゲーム応用編

クリアまでの回数をカウントするための変数「回数」を追加し、初期値0をセット

このループを使ってプログラムをすっきりさせる



クリアしたとき、失敗したときに音を鳴らす

ボタンを押すたびに「回数」を1増やす

ボタンが押されたら「カウンター」を変えるループから抜ける

クリアしたら「回数」を表示する

関数から「回数」を返すようにする

# タイミングゲームおうようへん応用編

ずっと

文字列を表示 "Level1"

変数 Level1の回数 を 呼び出し ゲーム実行 100 にする

文字列を表示 "Level2"

変数 Level2の回数 を 呼び出し ゲーム実行 50 にする

文字列を表示 "Level3"

変数 Level3の回数 を 呼び出し ゲーム実行 20 にする

変数 スコア を Level1の回数 × 10 + Level2の回数 × 4 + Level3の回数 にする

文字列を表示 "Score"

数を表示 スコア

メロディを開始する ジ・エンターテイナー <繰り返し 一度だけ

一時停止(ミリ秒) 2000

関数から「回数」を返すようにしたので、返したデータを変数に入れる

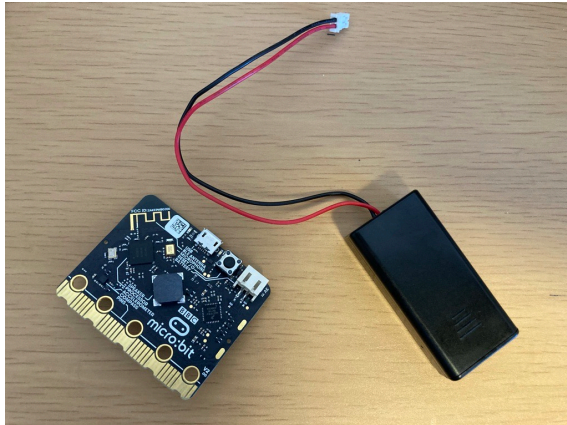
スコアを計算する

スコアを表示する

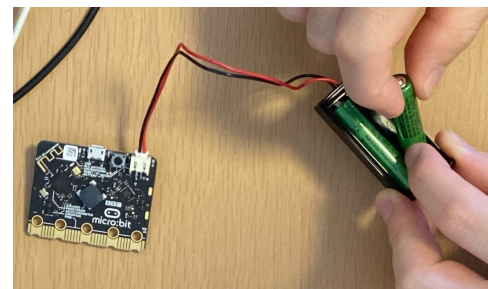
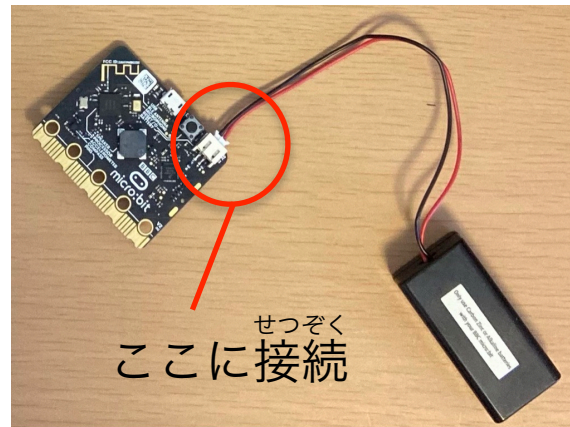
スコアを表示したら音楽を鳴らす

# micro:bitに電池ボックスを直接つなぐ場合

- タイミングゲームのようにブレッドボードを使わない場合、USBケーブルを接続していない間は電池ボックスをつなぐ必要がある
- 取り外しは大変なので、スイッチが電池の取り外しで電源のON/OFFを行う

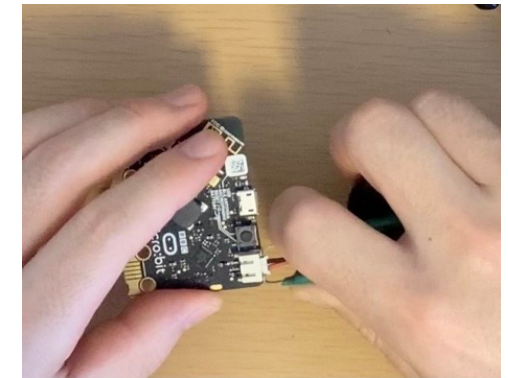
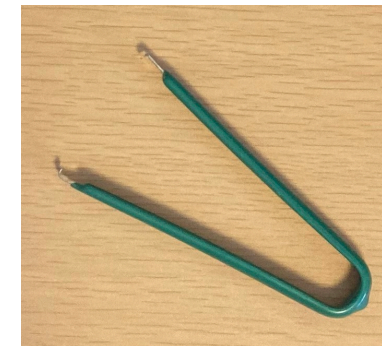


micro:bit go V2 スターターキットの場合には単4電池ボックス



電池を入れて電源ON

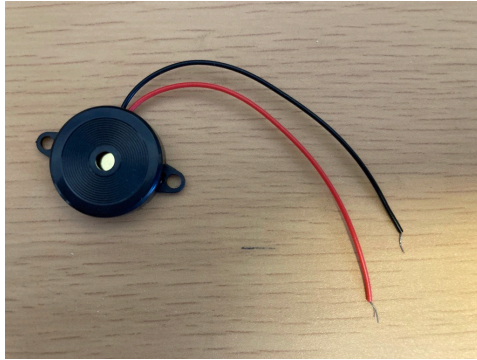
※取り外すときは、ケーブルを切らないように、しっかりプラスチック部分を持つ



IC引き抜き工具を使うと抜きやすい  
(【参考】購入できる場所の例: <https://akizukidenshi.com/catalog/g/gT-13303/>)

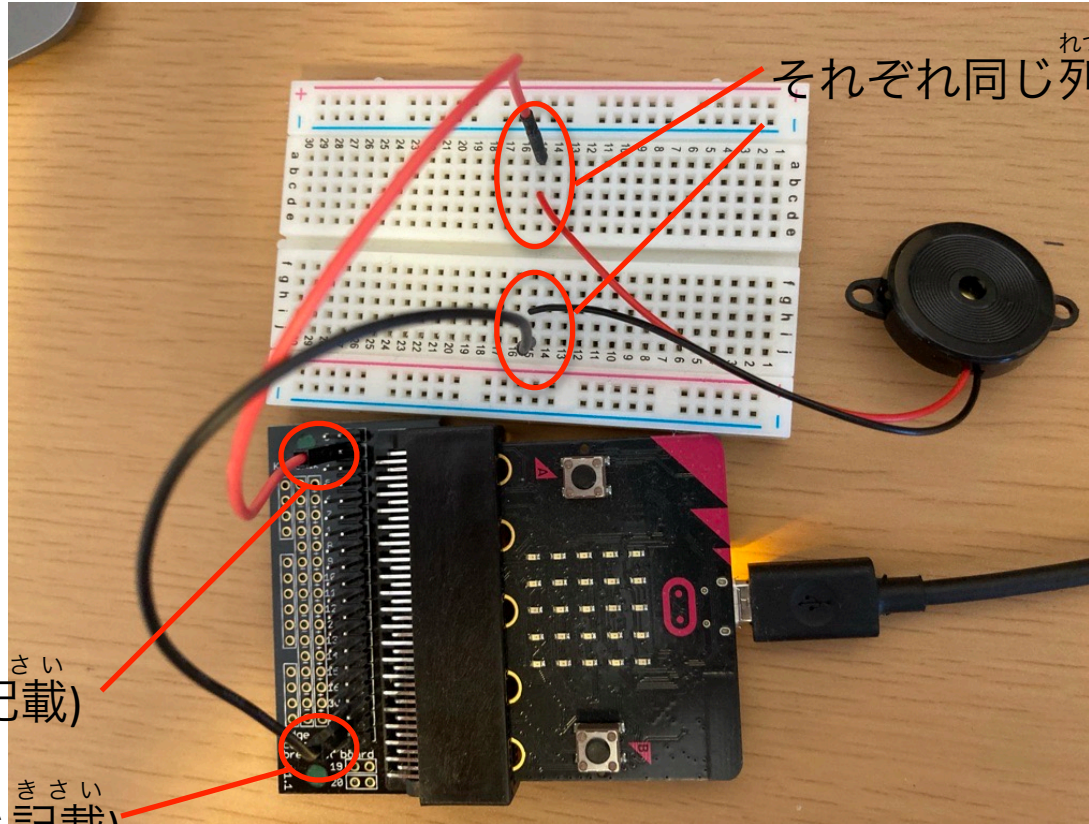
# micro:bit V1で音を鳴らす方法

- ブレッドボードを使って、<sup>あつでん</sup>圧電スピーカーと<sup>せつぞく</sup>micro:bitを接続する



<sup>あつでん</sup>圧電スピーカー

(【参考】<sup>こうにゆう</sup>購入できる場所の例:<sup>れい</sup><https://akizukidenshi.com/catalog/g/gP-01251/>)



<sup>れつ</sup>それぞれ同じ列の穴に挿す

上から2番目のピン("0"と<sup>きさい</sup>記載)

下から2番目のピン("0V"と<sup>きさい</sup>記載)

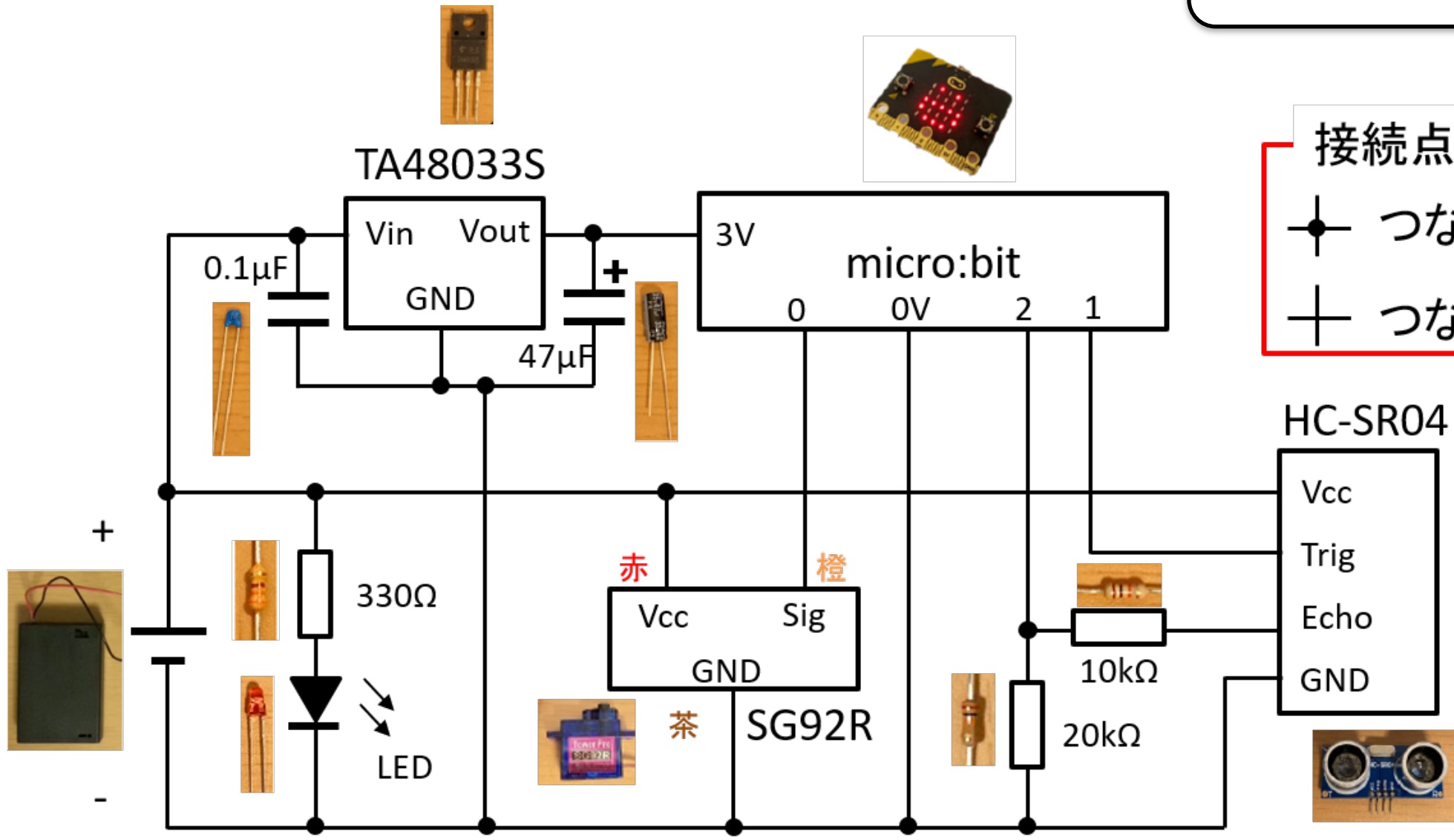
※micro:bit V2はスピーカーが付いているので上の<sup>たいおう</sup>対応は<sup>ふよう</sup>不要

2

# 自動開閉パーティションの補足情報

# かいるず 回路図

部品がどのようにつながっているかを表す図



**接続点**

⊕ つながっている

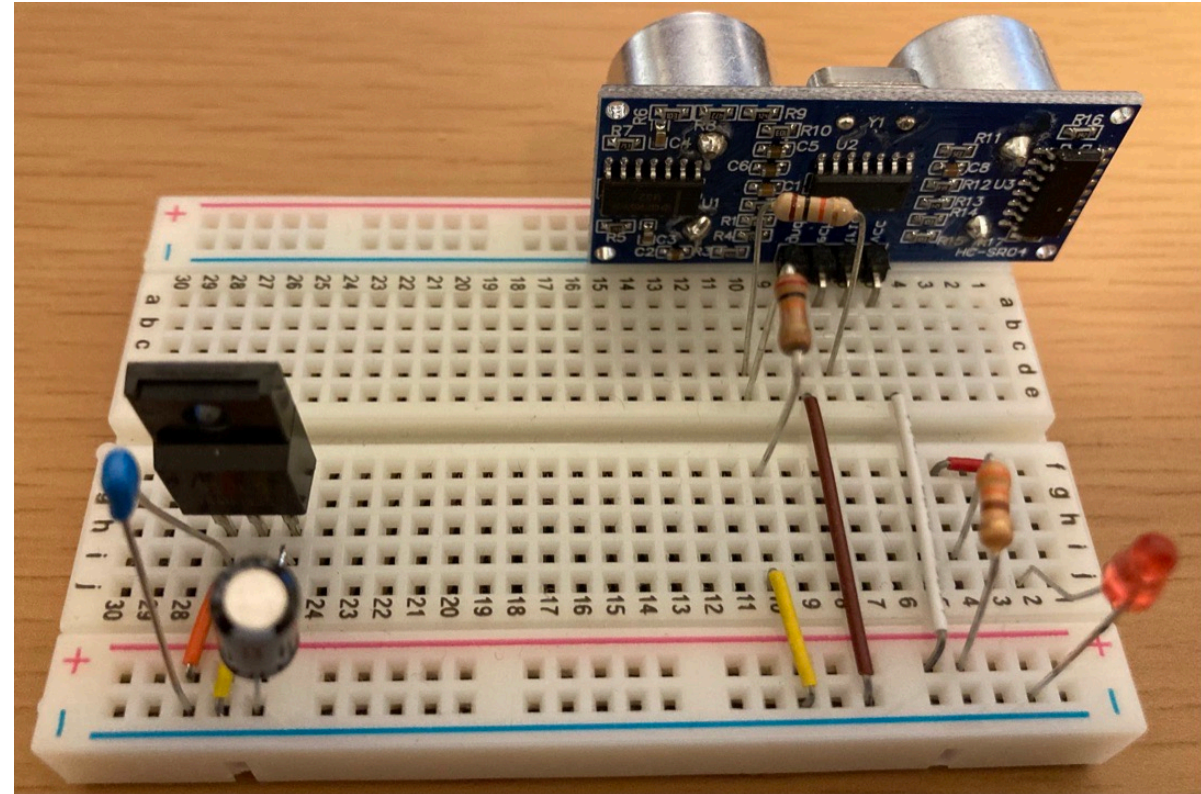
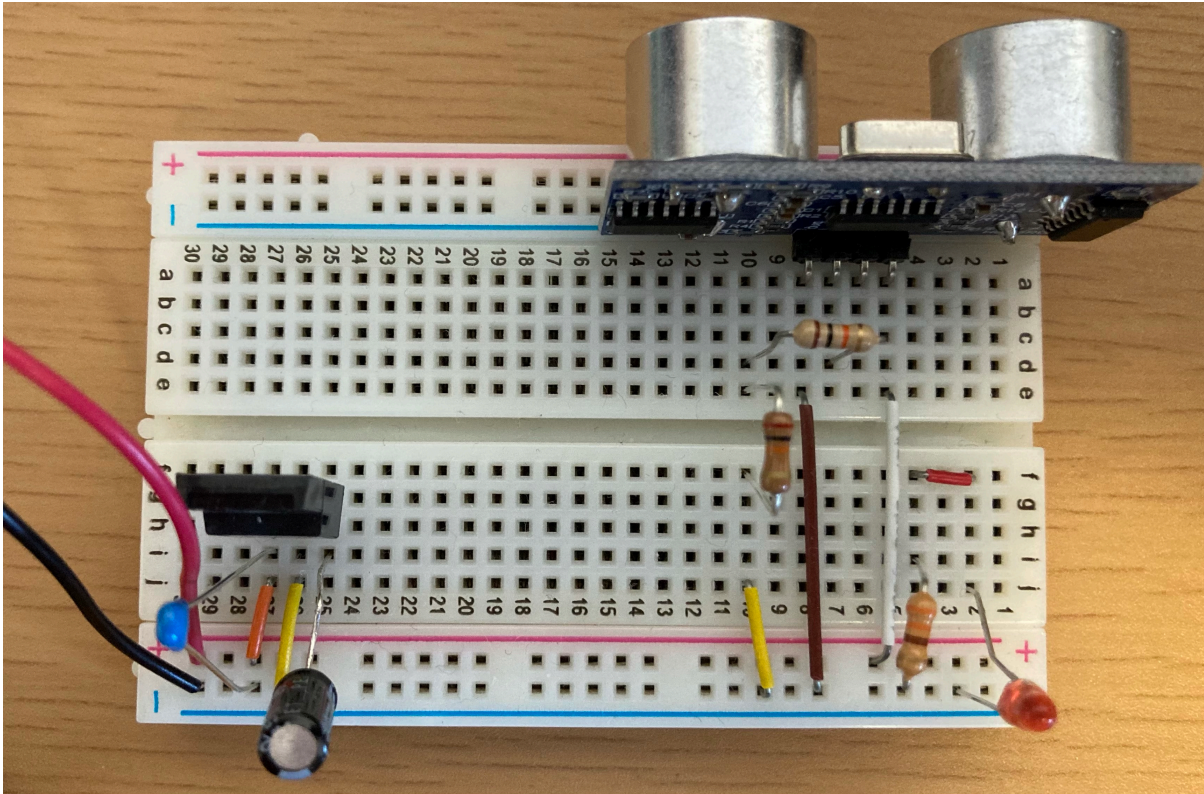
⊕ つながっていない

ブレッドボード上の配線を見て、回路図と対応づいているか確認してみましょう

micro:bitとサーボモータ・超音波距離センサは動作する電圧が異なる。「電圧レベル変換」をするのが正しいやり方だが、ここでは回路をかたんにするため抵抗器などで代替。

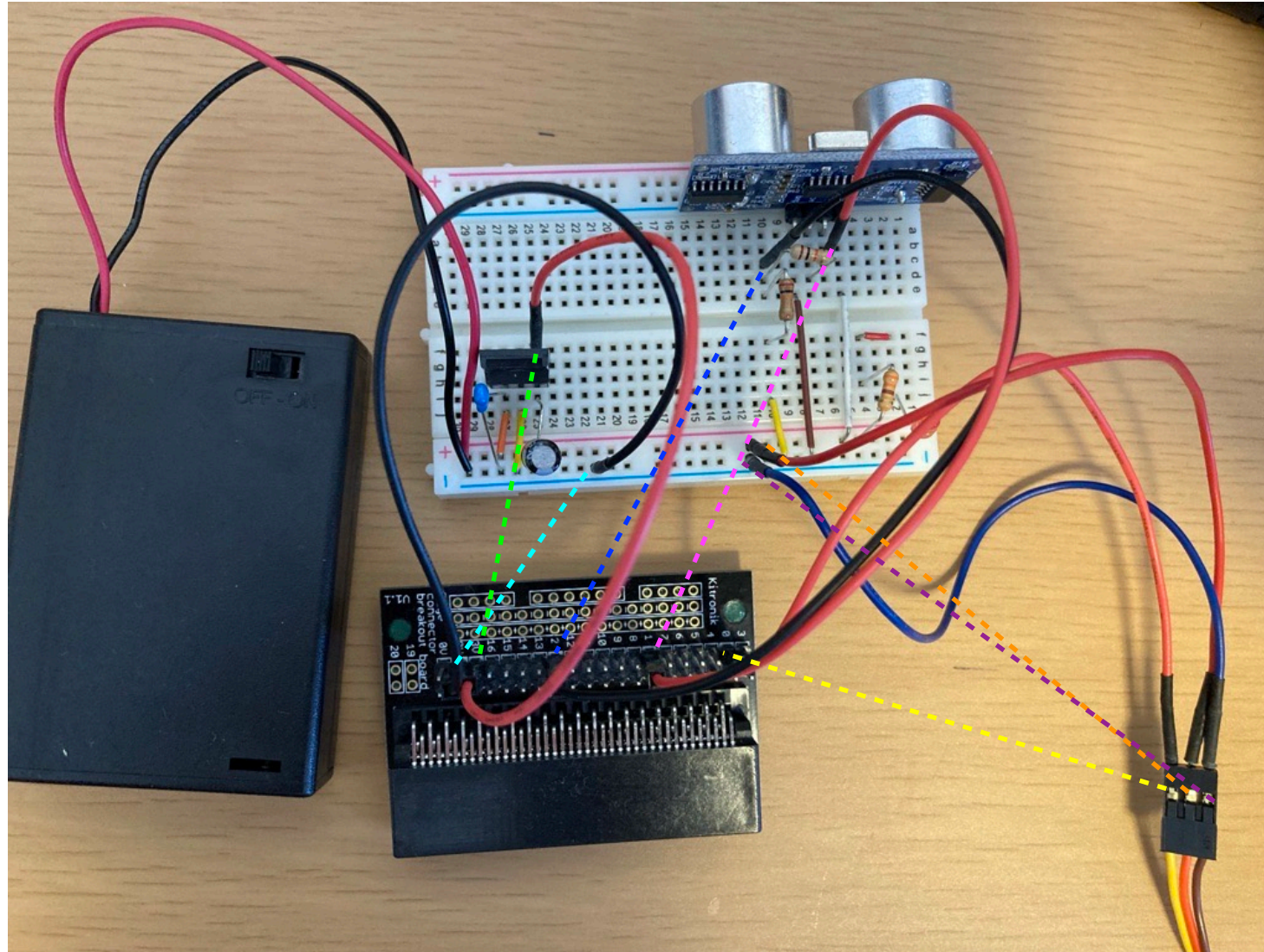
# ブレッドボード上の回路 かいる

micro:bitやサーボモータと接続する前の状態 せつぞく じょうたい



# ブレッドボード上の回路 かいる

Micro:bitやサーボモータを せつぞく接続したあと

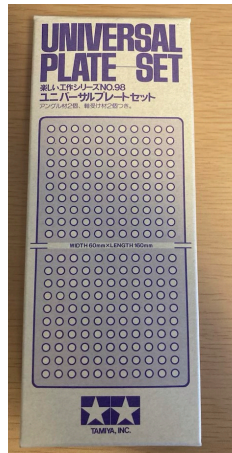


※点線でジャンパー線  
せつぞく ひょうげん  
の接続を表現

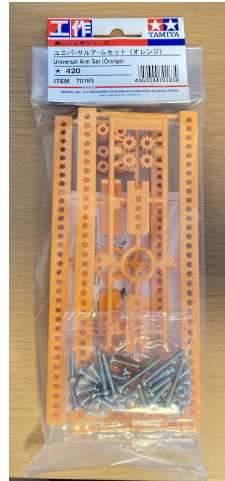


# かいへい 自動開閉パーティションの作り方

## ここで使うもの



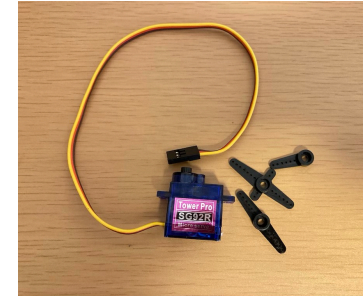
ユニバーサル  
プレートセット



ユニバーサル  
アームセット



プラバン(B4 厚み0.3mm)



サーボモータ

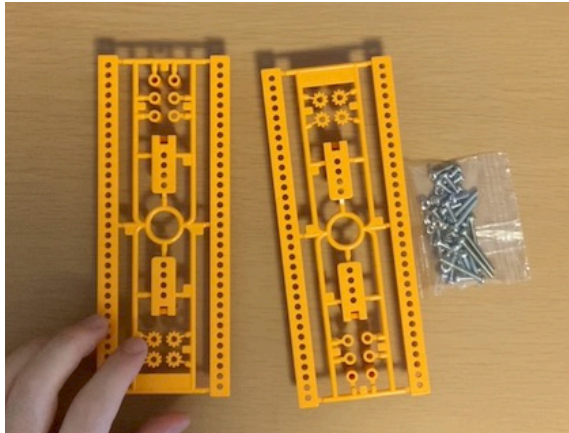


せっちやくざい  
プラスチック対応 接着剤

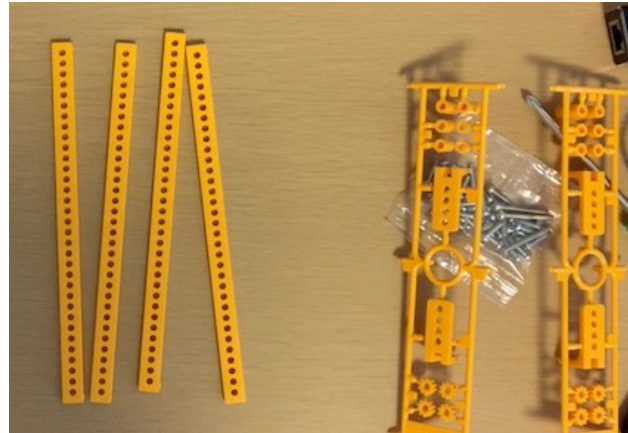


木板とタッピングネジ

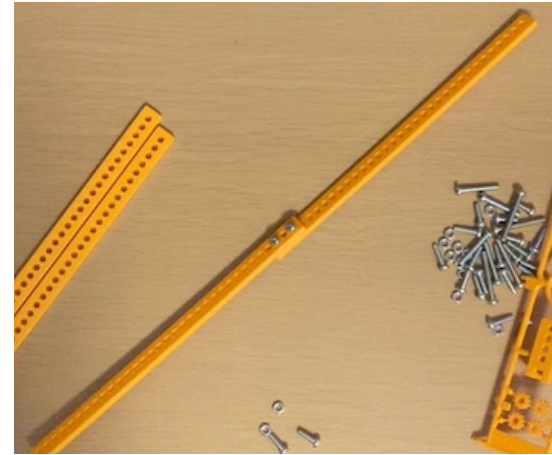
# かいへい 自動開閉パーティションの作り方



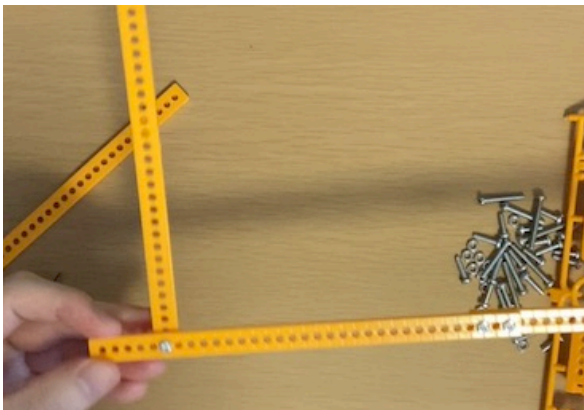
(1) ユニバーサルアームセットを<sup>かいふう</sup>開封



(2) 4つのアーム部品を切り取る



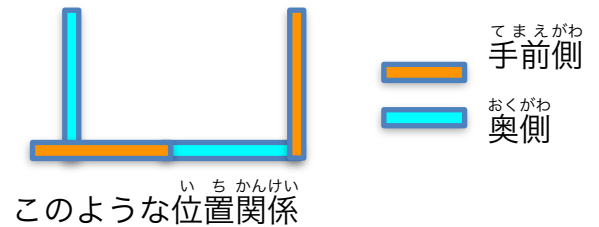
(3) 長さ1cmのネジと、ナットをそれぞれ2つずつ使  
い、2つのアーム部品を<sup>れんけつ</sup>連結する  
長さは必要に応じて変えてよい(ここでは<sup>あな</sup>穴が3つ重なるように)



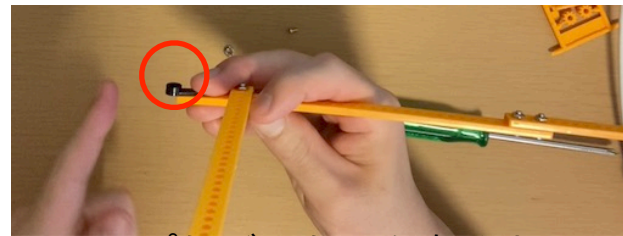
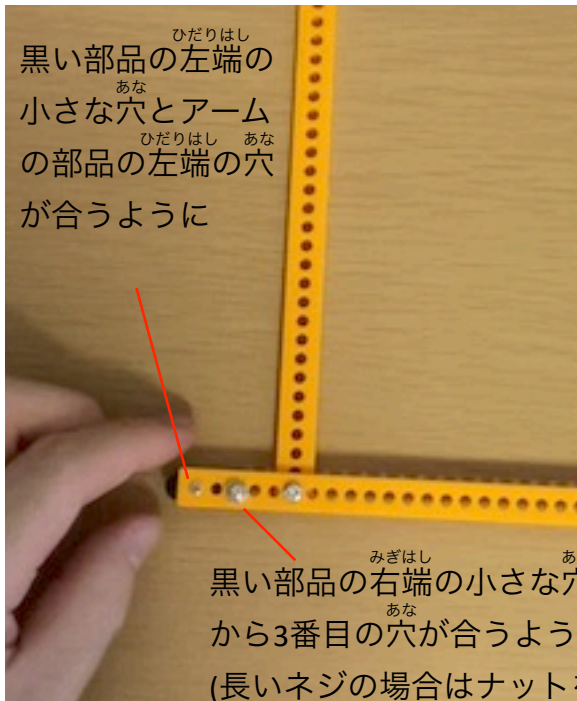
(4) <sup>れんけつ</sup>連結したアームの左から6つ目  
<sup>あな</sup>の穴にアーム部品を取り付ける  
(長さ1cmのネジと、ナットを使う)



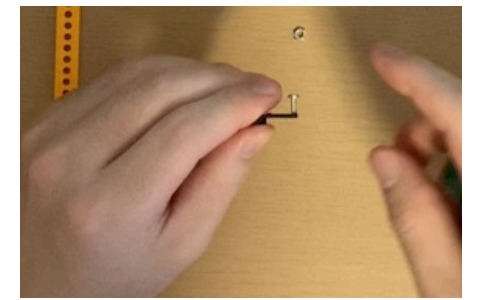
(5) <sup>れんけつ</sup>連結したアームの右端の<sup>あな</sup>穴に  
アーム部品を取り付ける  
(長さ1cmのネジと、ナットを使う)



# かいへい 自動開閉パーティションの作り方

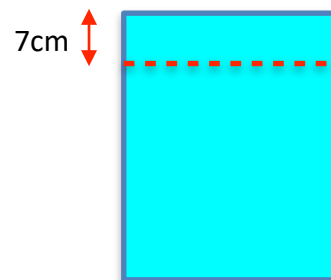


でっぱりがこちらを向くように



じぜん  
事前にサーボモータ付属のネジで少し刺さるぐらいにネジを回しておいて  
から、ネジを取り外し、それからアーム部品に取り付けるとやりやすい

れんけつ  
(6) 連結したアームの左端にサーボモータ  
ひだりはし  
付属の黒い部品を取り付ける  
ふぞく  
(サーボモータ付属のネジ2つを使う)

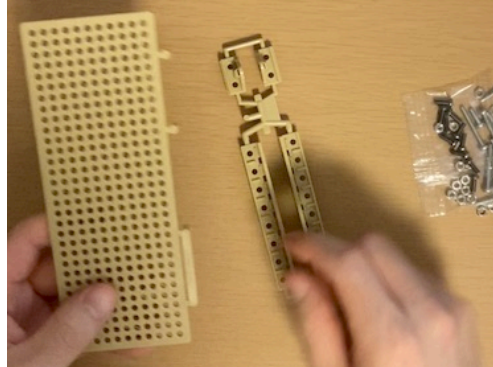


(7) パーティションの大きさに合わせてプ  
ラバンを切り取る  
い ち  
(ここでは上から7cmの位置で切り取る)



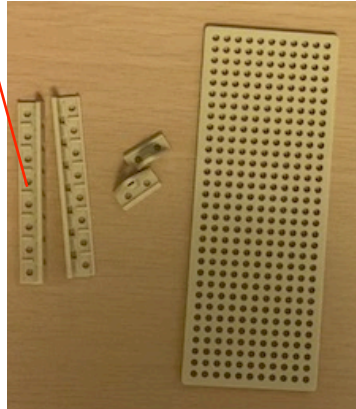
せつちやくざい  
(8) 切り取ったプラバンを接着剤でアーム  
部品につける

# かいへい 自動開閉パーティションの作り方



(9) ユニバーサルプレートセットを  
かいふう  
開封する

一つは使用しない

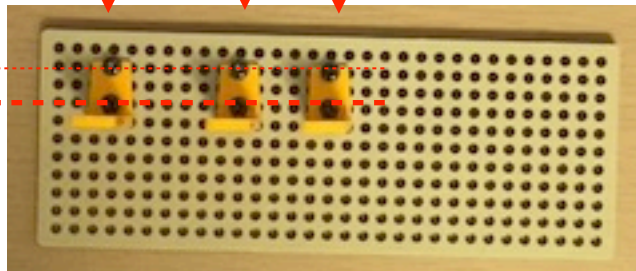


(10) 部品をニッパーで切りはなす



(11) ユニバーサルアームセットの  
L字パーツを3つ切りはなす

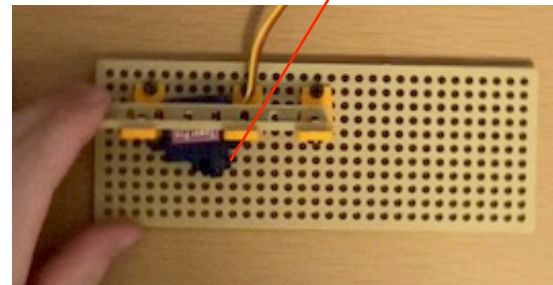
左から4番目の穴<sup>あな</sup>  
左から11番目の穴<sup>あな</sup>  
左から16番目の穴<sup>あな</sup>



上から2番目の穴<sup>あな</sup>  
上から4番目の穴<sup>あな</sup>

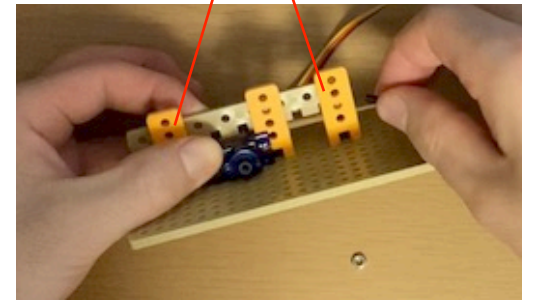
(12) L字パーツをプレートに取り付ける

じく みぎがわ  
軸が右側に来るように



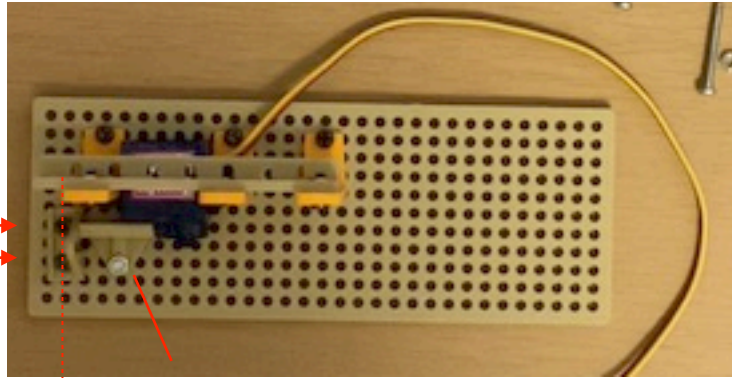
(13) サーボモータを置き、上から細長いL字パーツを取り付ける

上から2番目の穴<sup>あな</sup>をネジでとめる



# かいへい 自動開閉パーティションの作り方

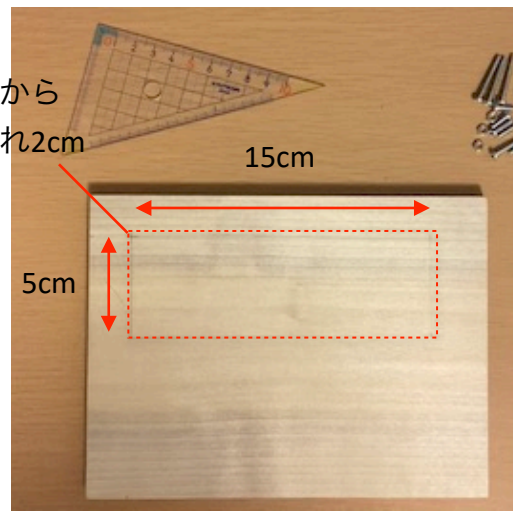
あな  
下から5番目の穴  
あな  
下から3番目の穴  
(ネジは上から)



あな  
左から2番目の穴  
あな  
左から5番目、下から3番目の穴  
(ネジはプレートの下から通す)

(14) 三角パーツをプレートに取り付ける

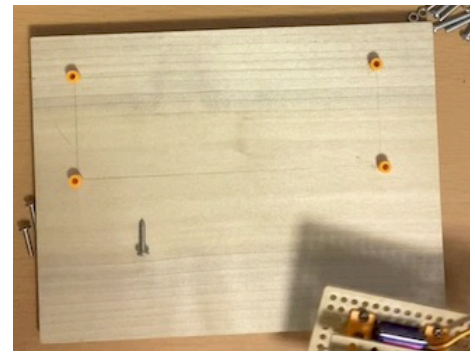
左・上から  
それぞれ2cm



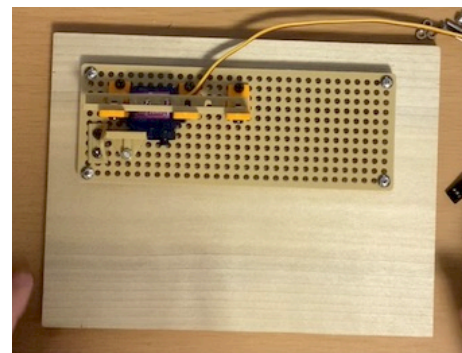
(15) 5cm×15cmの長方形を木板上に描く



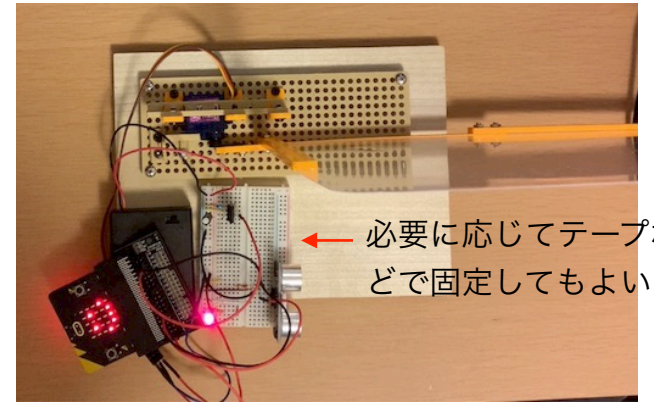
(16) 長方形の四隅をキリで軽く穴あけ



(17) 一番短いスペーサーを4つ穴の上に置く



(18) プレートを上に置いて、  
四隅を木ネジでとめる



(19) アーム部品をサーボモータにつなぎ、回路をつなぐ

必要に応じてテープなどで固定してもよい

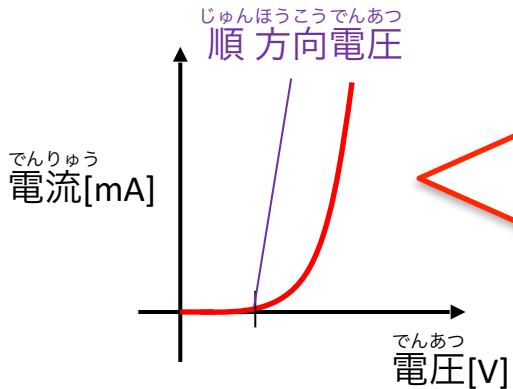
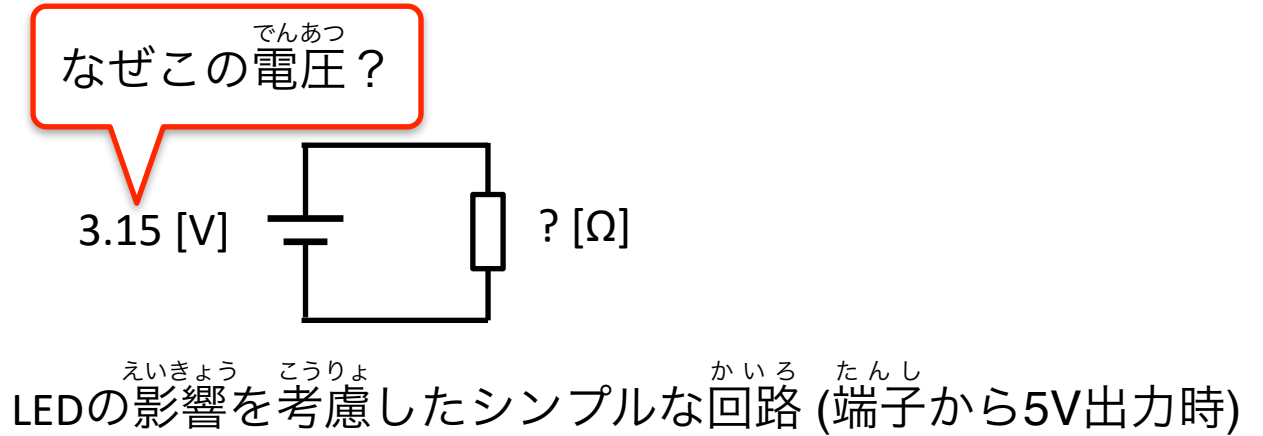
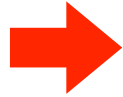
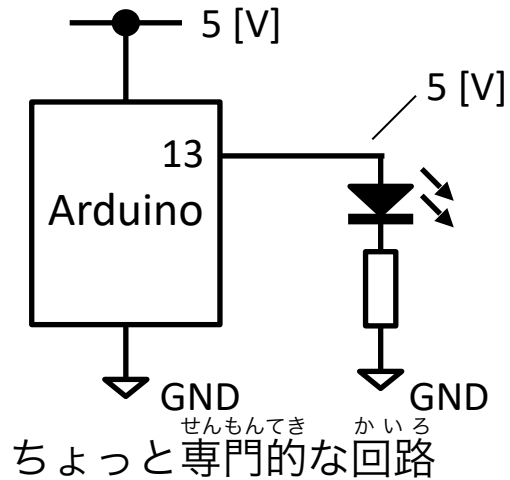
3

# Arduinoを使ったLED点滅実験の補足情報

てんめつじっけん

ほそくじょうほう

# シンプルな回路にしたときの電圧について



LEDはある程度電圧を加えないと電流が流れない!  
この電圧が「順方向電圧」

LEDの電圧-電流特性グラフ

今回使用するLEDの順方向電圧: 1.85~2.5V



抵抗器に加わる電圧は、順方向電圧を除いて考える

$$\text{Arduinoの端子の電圧}(5 \text{ [V]}) - \text{LEDの順方向電圧}(1.85 \text{ [V]}) = 3.15 \text{ [V]}$$

# かいはつかんきょう Arduino開発環境のインストール方法

以下のページからインストーラをダウンロード

<https://www.arduino.cc/en/software>

PROFESSIONAL EDUCATION STORE

Search on Arduino.cc

SIGN IN

HARDWARE SOFTWARE CLOUD DOCUMENTATION COMMUNITY BLOG ABOUT

TUTORIALS  
REFERENCE  
PLAYGROUND

Arduino Web Editor  
Start coding online and save your sketches in the cloud. The up-to-date version of the IDE includes all libraries and also supports new Arduino boards.  
CODE ONLINE GETTING STARTED

The Arduino® Student Kit: bring the buzza home

### Downloads

Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.  
Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE  
Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

- Windows Win 7 and newer
- Windows ZIP file
- Windows app Win 8.1 or 10 [Get](#)
- Linux 32 bits
- Linux 64 bits
- Linux ARM 32 bits
- Linux ARM 64 bits
- Mac OS X 10.10 or newer

Release Notes Checksums (sha512)

Hourly Builds  
Download a **preview of the incoming release**

Previous Releases  
Download the previous version of the current

Help

Windowsの場合は  
ここをクリック

Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **51,232,022** times — impressive! Help its development with a donation.

\$3 \$5 \$10 \$25 \$50 Other

JUST DOWNLOAD [CONTRIBUTE & DOWNLOAD](#)

きふ  
寄付する必要がなければこちらをクリック

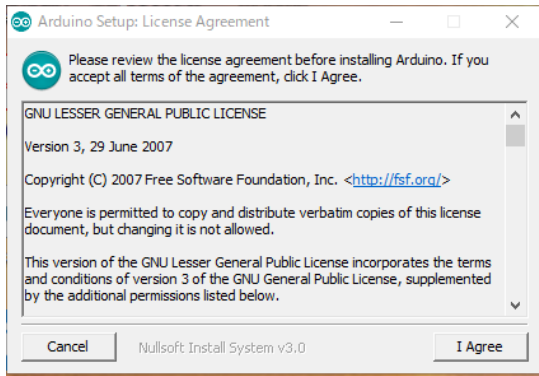
Learn more about [donating to Arduino](#).

いこう ばん  
以降はWindows版で説明

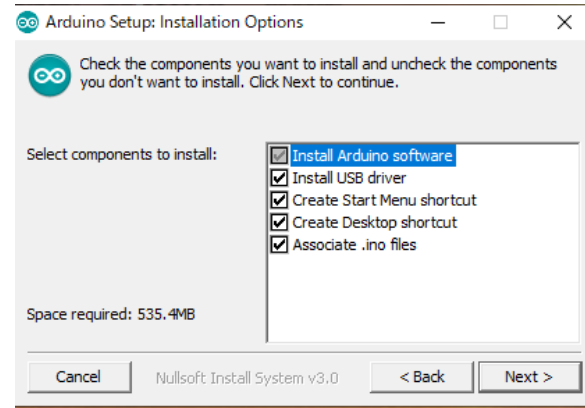


# Arduino開発環境のインストール方法

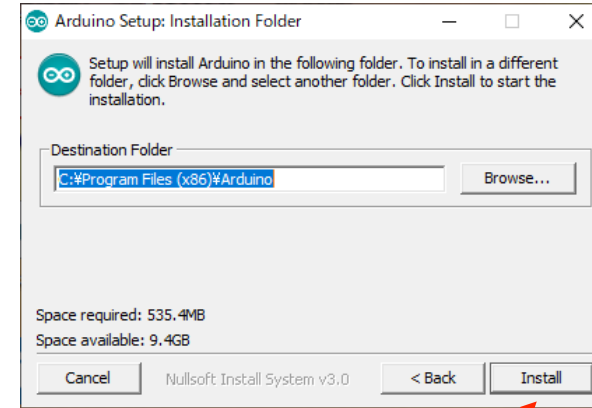
ダウンロードしたインストーラーをダブルクリックで実行し、インストールをすすめる



ここをクリック



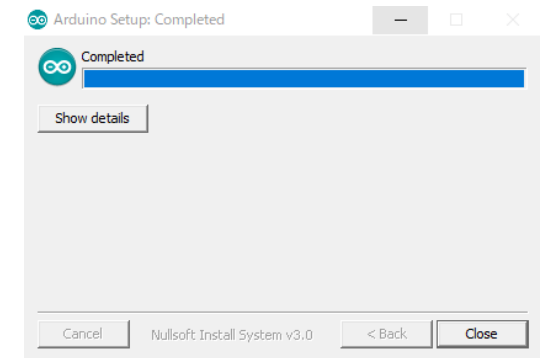
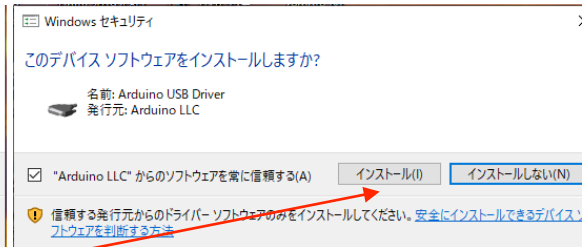
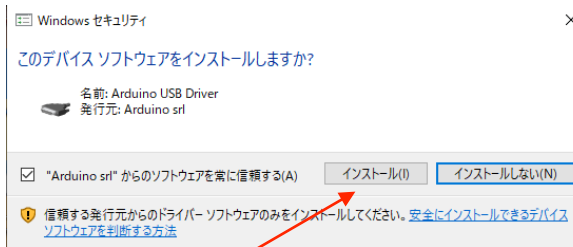
そのままここをクリック



そのままここをクリック



この表示が出たら「インストール」をクリック



ここをクリックして完了

かんりょう

# かいる 回路作成

せつぞく  
いい  
接続例

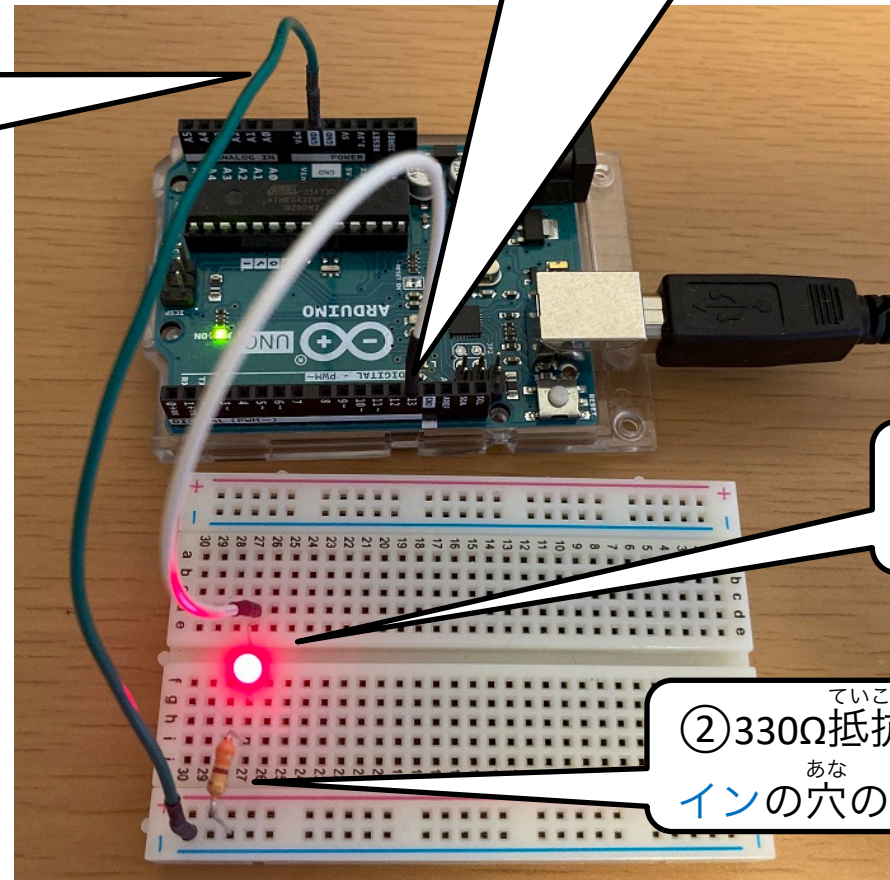
③ ジャンパー線をArduinoのGND  
端子とブレッドボードの青ライン  
の穴いずれかに挿す。

④ ジャンパー線をArduinoの13番  
ピンとブレッドボードのd-27に  
挿す。

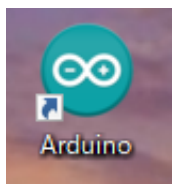
⑤ USBケーブルを  
Arduinoとパソコン  
に接続する。

① LEDの足の長い端子をe-27、短  
い端子をf-27に挿す。

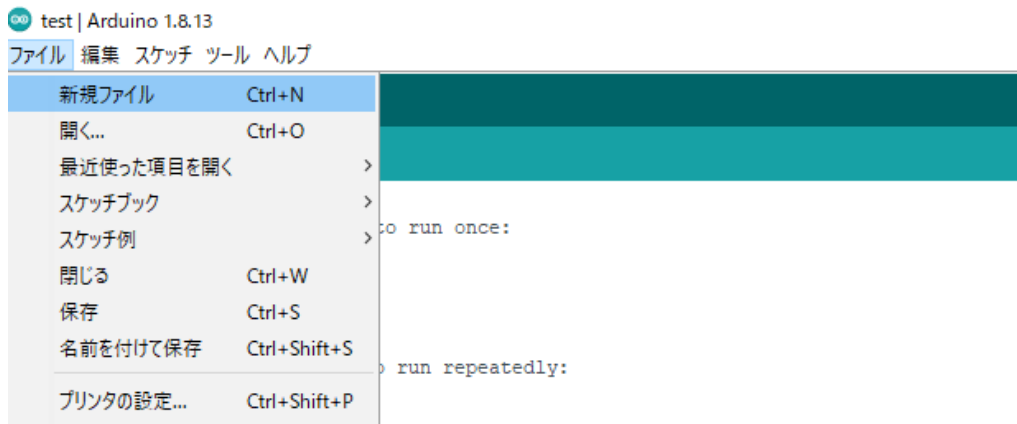
② 330Ω抵抗器(橙橙茶金)をi-27, 青ラ  
インの穴のいずれかに挿す。



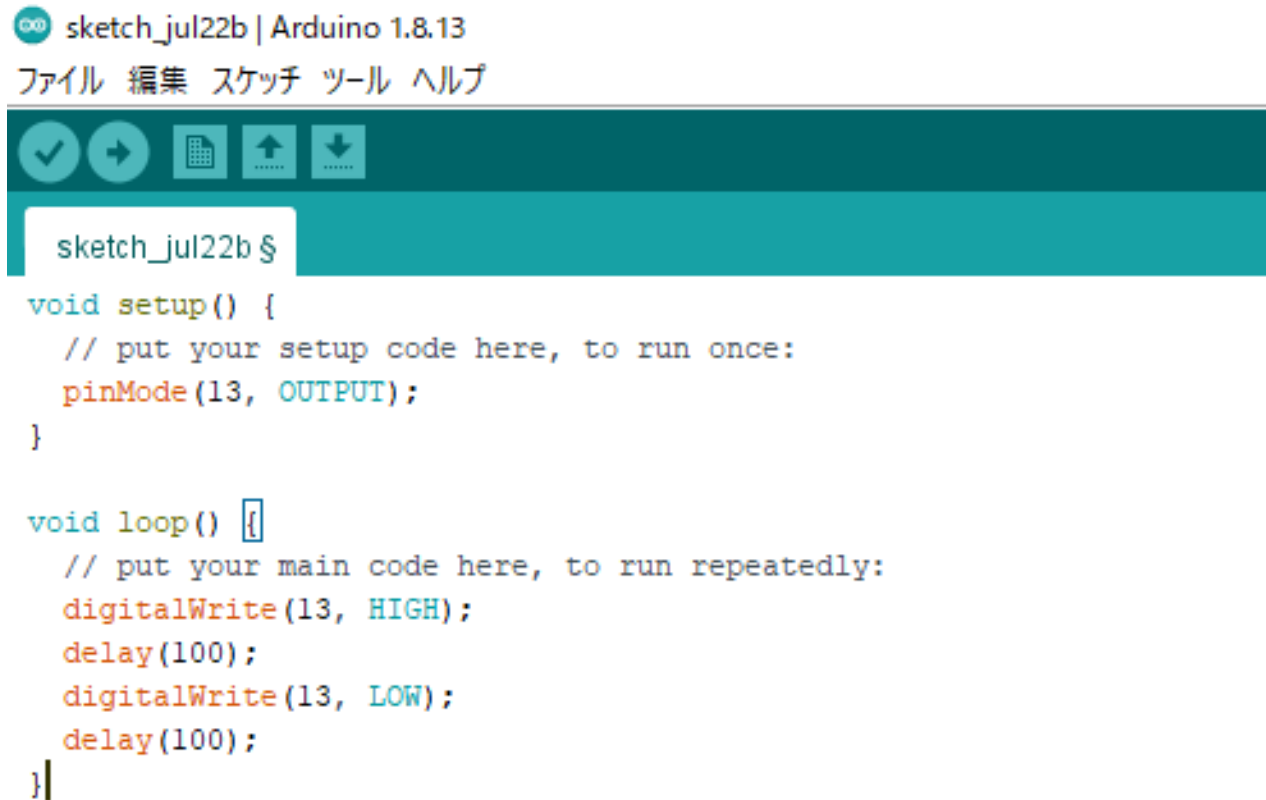
# てんめつ LED点滅プログラムの作成



(1) ダブルクリックして  
Arduino IDEを起動

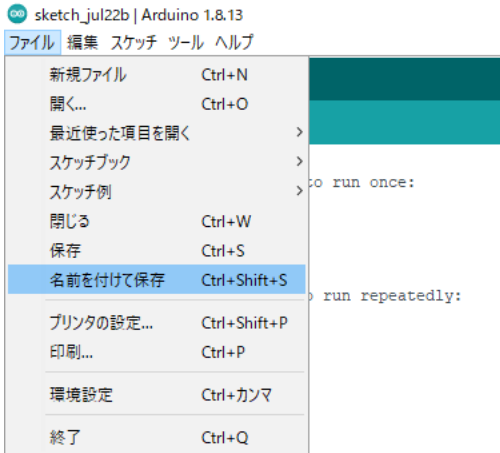


(2) 「ファイル」 → 「新規ファイル」 をクリック

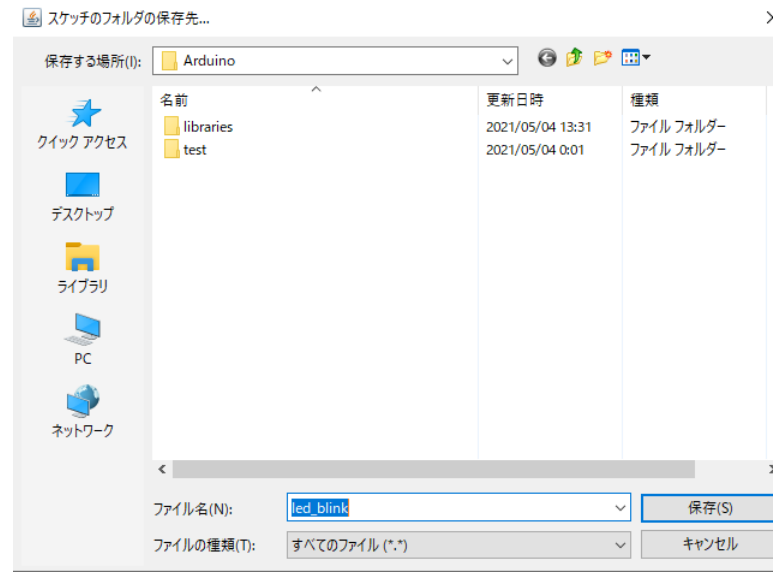


(3) プログラムを入力

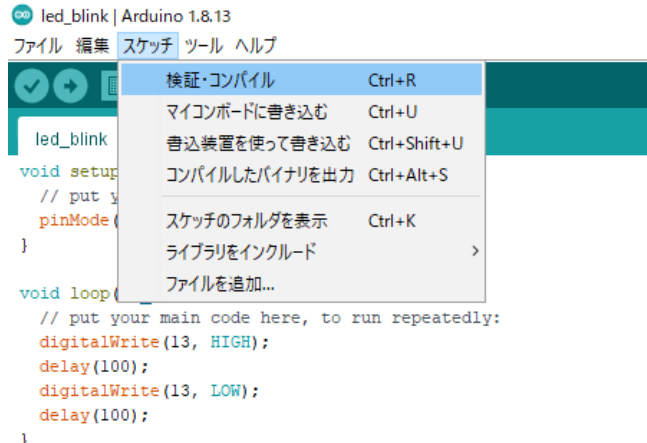
# てんめつ LED点滅プログラムの作成



(4) 「ファイル」 → 「名前を付けて保存」をクリック  
いこう  
(以降は、同じファイルであれば「保存」をクリックして変更を保存する)  
へんこう ほぞん

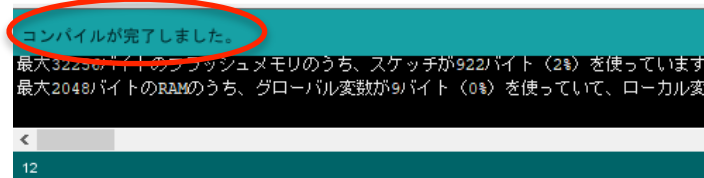


てきとう  
(5) 適当な名前をつけて「保存」をクリック  
(ここでは「led\_blink」とした)



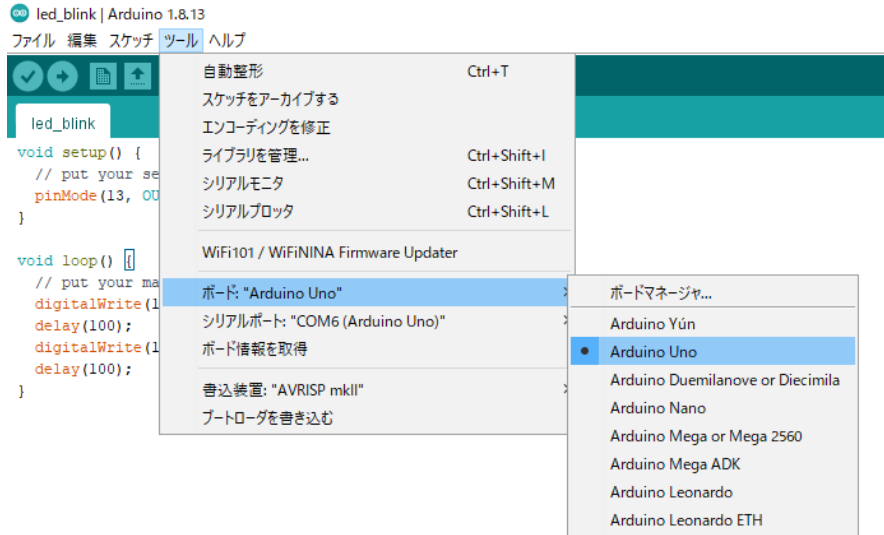
(6) 「スケッチ」 → 「検証・コンパイル」をクリック

```
// put your main code here, to run repeatedly.
digitalWrite(13, HIGH);
delay(100);
digitalWrite(13, LOW);
delay(100);
}
```

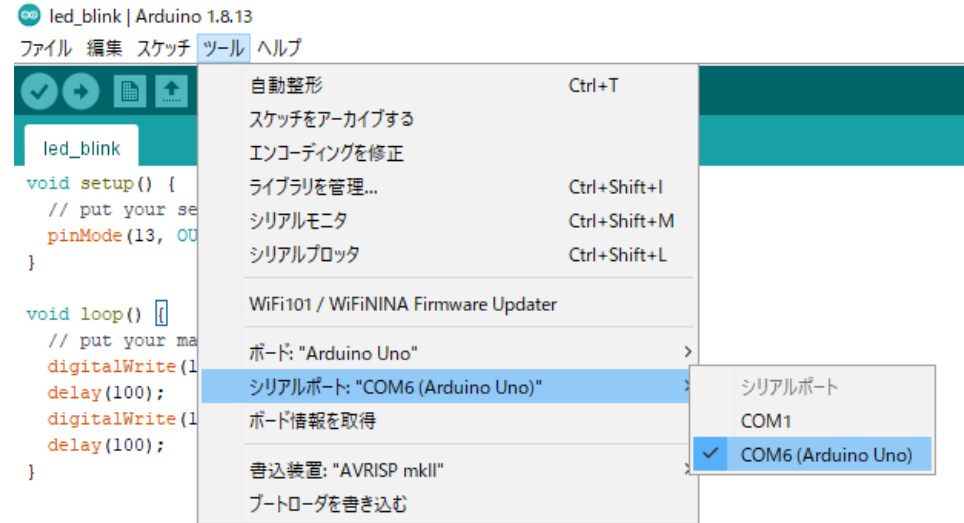


かんりょう かくにん  
コンパイルが完了することを確認

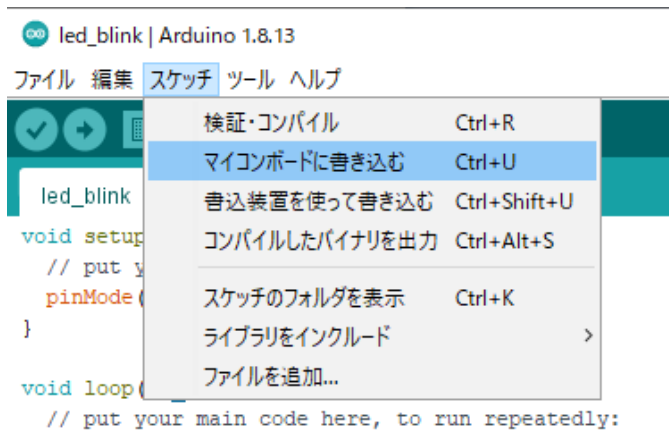
# てんめつ LED点滅プログラムの作成



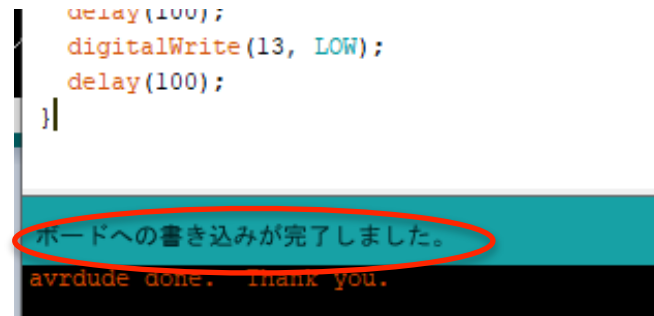
(7) 「ツール」 → 「ボード」 → 「Arduino Uno」 をクリック



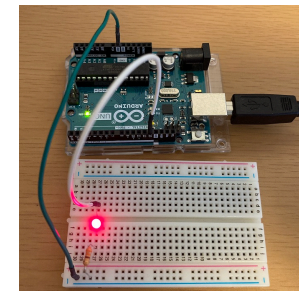
(8) 「ツール」 → 「シリアルポート」 → 「(Arduino Uno) がついているCOMをクリック  
(Arduinoがパソコンに接続されていないと出てこない)



(9) 「スケッチ」 → 「マイコンボードに書き込む」 をクリック



ボードへの書き込みが完了することを  
確認し、LEDが点滅することを確認する



# プログラムの解説

```
void setup() {  
  pinMode(13, OUTPUT); // 13番ピンを出力に設定  
}
```

マイコンの端子は、「出力」や「入力」を設定できる  
(入力の場合はスイッチが押されているかなどを判断できる)

```
void loop() {  
  digitalWrite(13, HIGH); // 13番ピンをHIGH(5V)に  
  delay(100); // 0.1秒一時停止  
  digitalWrite(13, LOW); // 13番ピンをLOW(0V)に  
  delay(100); // 0.1秒一時停止  
}
```

電流が流れるのでLEDが光る

電流が流れないのでLEDが消える

## setupやloopは関数

setup関数の中の処理は、電源がON  
になったとき一回だけ実行される

MakeCodeのこの  
ブロックと同じ



loop関数の中の処理は、ずっとくり返される

MakeCodeのこの  
ブロックと同じ

