

マイクロビット プログラミング実習 ～もぐらたたきゲーム～

塾のフリータイムの実習課題として、マイクロビットのみでもあそべる「もぐらたたきゲーム」を用意しました。

興味のある方はプログラミングしてみてください。

ゆめほたる環境科学技術クラブ

ゲームの内容

- Aボタンを押すと、ゲームがスタートします。
- 1～5秒たつと、画面に一瞬だけ「0」～「2」のいずれかの数字が表示されます。
- 数字が表示されている間に、同じ番号の端子にタッチすると、1点プラスされます。
- これを10回繰り返します。
- ゲームが終わったら、点数（タッチできた回数）が表示されます。

次のページからの「仕様」を見て、プログラミングしてみてください。

仕様 1 (変数の準備)

- 「点数」「モグラ」「ハンマー」という、3つの変数を準備します。
 - ① 「点数」はゲームの点数（タッチできた回数）
 - ② 「モグラ」は画面に表示される数字（0～2）
 - ③ 「ハンマー」はタッチされた端子の番号（0～2）

仕様 1 (変数の準備)

The image shows a software interface for creating a Scratch script. On the left is a sidebar with a search bar and a list of block categories: 基本 (Basic), 入力 (Input), 音楽 (Sound), LED, 無線 (Wireless), ループ (Loops), 論理 (Logic), 変数 (Variables), 計算 (Math), 拡張機能 (Extensions), and 高度なブロック (Advanced Blocks). The '変数' (Variables) category is selected and highlighted in red. The main workspace on the right is titled '変数' (Variables) and contains a '変数を追加する...' (Add Variable...) button. Below it are two red blocks: '変数 ハンマー を 0 にする' (Set variable 'ハンマー' to 0) and '変数 ハンマー を 1 だけ増やす' (Increase variable 'ハンマー' by 1). At the bottom, a section titled 'Your Variables' lists three variables: 'ハンマー', 'モグラ', and '点数', each with a dropdown arrow.

検索...

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

変数

変数を追加する...

変数 ハンマー を 0 にする

変数 ハンマー を 1 だけ増やす

Your Variables

- ハンマー
- モグラ
- 点数

仕様2（ボタンAが押されたとき）

- 「ボタンAが押されたとき」では、以下の処理を行います。
 - ① 変数「点数」の値を「0」にします。
 - ② 変数「モグラ」の値を「-1」（ありえない値）にします。
 - ③ 変数「ハンマー」の値を「-1」（ありえない値）にします。
 - ④ 画面に「3」→「2」→「1」→「格子アイコン」を0.5秒ずつ表示します。
 - ⑤ 画面表示を消してから、「ゲームをする」関数を呼び出します。

ゲームをはじめる処理です

仕様2 (ボタンAが押されたとき)



ボタン A が押されたとき

- 変数 点数 を 0 にする
- 変数 モグラ を -1 にする
- 変数 ハンマー を -1 にする
- 変数 カウンター を 0 から 2 に変えてくりかえす
- 数を表示 3 - カウンター
- 一時停止 (ミリ秒) 500
- アイコンを表示 [チェッカー]
- 一時停止 (ミリ秒) 500
- 表示を消す
- 呼び出し ゲームをする

仕様3（「ゲームをする」関数）

- 「ゲームをする」関数では、以下の処理を行います。
 - ① 1～5秒のランダムな時間だけ、一時停止します。
 - ② 変数「モグラ」の値を「0」～「2」のランダムな値にします。
 - ③ 画面に「モグラ」の値を0.2秒だけ表示します。
 - ④ 0.2秒たったら表示を消し、「モグラ」の値を「-1」（ありえない値）に戻します。
 - ⑤ 上記の 1～4 の処理を10回くり返します。
 - ⑥ 10回終わったら、0.5秒待ってから「格子アイコン」を3回点滅（0.5秒点灯→0.5秒消灯）させた後、画面に変数「点数」の値を表示します。

ゲームを出題する処理です

仕様3（「ゲームをする」関数）



仕様4 (ずっと)

- 「ずっと」では、以下の処理を行います。
 - ① もしも「P0」がタッチされていたら、変数「ハンマー」の値を「0」にして、「判定する」関数を呼び出します。
 - ② もしも「P1」がタッチされていたら、変数「ハンマー」の値を「1」にして、「判定する」関数を呼び出します。
 - ③ もしも「P2」がタッチされていたら、変数「ハンマー」の値を「2」にして、「判定する」関数を呼び出します。
 - ④ 上記のいずれでもなければ（どの端子にもタッチされていないければ）、変数「ハンマー」の値を「-1」（あり得ない値）に戻します。

ゲームに答える処理です

仕様4 (ずっと)

```
ずっと  
もし 端子 P0 がタッチされている なら  
変数 ハンマー を 0 にする  
呼び出し 判定する  
でなければもし 端子 P1 がタッチされている なら  
変数 ハンマー を 1 にする  
呼び出し 判定する  
でなければもし 端子 P2 がタッチされている なら  
変数 ハンマー を 2 にする  
呼び出し 判定する  
でなければ  
変数 ハンマー を -1 にする
```

The image shows a Scratch script for a 'ずっと' (Forever) loop. The script contains the following blocks:

- A blue 'ずっと' (Forever) loop block.
- A teal 'もし' (If) block: '端子 P0 がタッチされている' (Terminal P0 is touched) 'なら' (If).
- A red '変数' (Variable) block: 'ハンマー' (Hammer) 'を' (set) '0' 'にする' (to).
- A blue '呼び出し' (Call) block: '判定する' (Judge).
- A teal 'でなければもし' (If not) block: '端子 P1 がタッチされている' (Terminal P1 is touched) 'なら' (If).
- A red '変数' (Variable) block: 'ハンマー' (Hammer) 'を' (set) '1' 'にする' (to).
- A blue '呼び出し' (Call) block: '判定する' (Judge).
- A teal 'でなければもし' (If not) block: '端子 P2 がタッチされている' (Terminal P2 is touched) 'なら' (If).
- A red '変数' (Variable) block: 'ハンマー' (Hammer) 'を' (set) '2' 'にする' (to).
- A blue '呼び出し' (Call) block: '判定する' (Judge).
- A teal 'でなければ' (If not) block.
- A red '変数' (Variable) block: 'ハンマー' (Hammer) 'を' (set) '-1' 'にする' (to).

仕様5（「判定する」関数）

- 「判定する」関数では、以下の処理を行います。
 - もしも、変数「モグラ」の値と変数「ハンマー」の値が同じなら（正解なら）、変数「点数」の値を1だけ増やし、変数「モグラ」の値を「-1」（ありえない値）に戻します。さらに「チェックアイコン」を一瞬（0.1秒）だけ表示してから、表示を消します。

ゲームの答え合わせの処理です

仕様5（「判定する」関数）



完成 (プログラム全体)

```
ボタン A が押されたとき
  変数 点数 を 0 にする
  変数 モグラ を -1 にする
  変数 ハンマー を -1 にする
  変数 カウンター を 0 から 2 に変えてくりかえす
  数を表示 3 - カウンター
  一時停止 (ミリ秒) 500
  アイコンを表示
  一時停止 (ミリ秒) 500
  表示を消す
  呼び出し ゲームをする
```

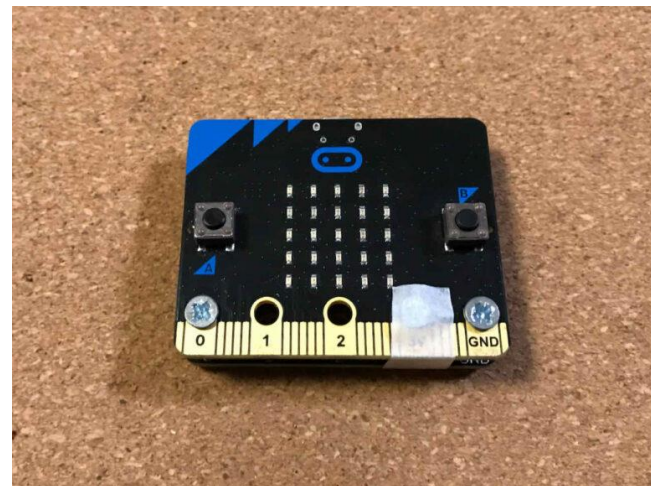
```
関数 ゲームをする
  くりかえし 10 回
    一時停止 (ミリ秒) 1000 から 5000 までの乱数
    変数 モグラ を 0 から 2 までの乱数 にする
    数を表示 モグラ
    一時停止 (ミリ秒) 200
    表示を消す
    変数 モグラ を -1 にする
  一時停止 (ミリ秒) 500
  くりかえし 3 回
    アイコンを表示
    一時停止 (ミリ秒) 500
    表示を消す
    一時停止 (ミリ秒) 500
  数を表示 点数
```

```
ずっと
  もし 端子 P0 がタッチされている なら
    変数 ハンマー を 0 にする
    呼び出し 判定する
  でなければもし 端子 P1 がタッチされている なら
    変数 ハンマー を 1 にする
    呼び出し 判定する
  でなければもし 端子 P2 がタッチされている なら
    変数 ハンマー を 2 にする
    呼び出し 判定する
  でなければ
    変数 ハンマー を -1 にする
```

```
関数 判定する
  もし モグラ = ハンマー なら
    変数 点数 を 1 だけ増やす
    変数 モグラ を -1 にする
    アイコンを表示
    一時停止 (ミリ秒) 100
    表示を消す
```

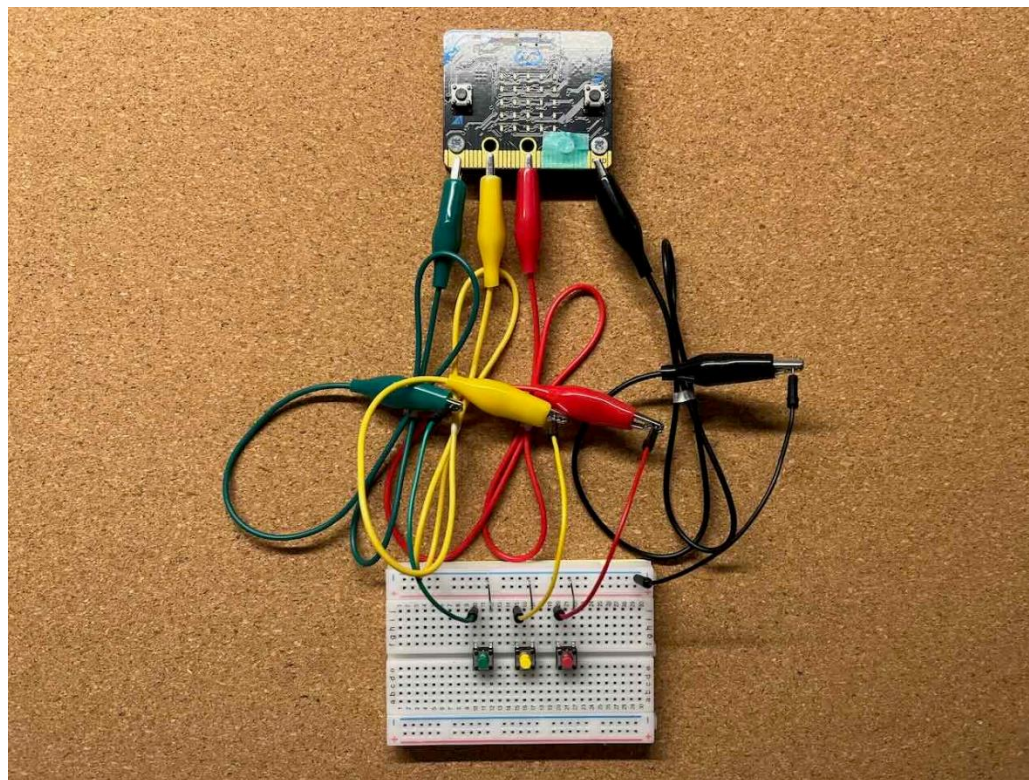
あそびかた

- 右手の親指で「GND」にさわった状態で、左手で「0」「1」「2」にタッチします。
- タッチセンサは感度が悪いので、タッチしても認識されないことがあります。そのような場合は、かわりに「ワニ口クリップ」を使うこともできます。
 - ワニ口クリップの一方を「GND」につなぎ、もう一方を「0」「1」「2」にタッチさせます。
- ✓ マイクロビットの「3V」端子と「GND」端子がショートするとこわれる可能性がありますので、ワニ口クリップであそぶ場合は、あらかじめ「3V」端子をテープなどでふさいでおいてください。



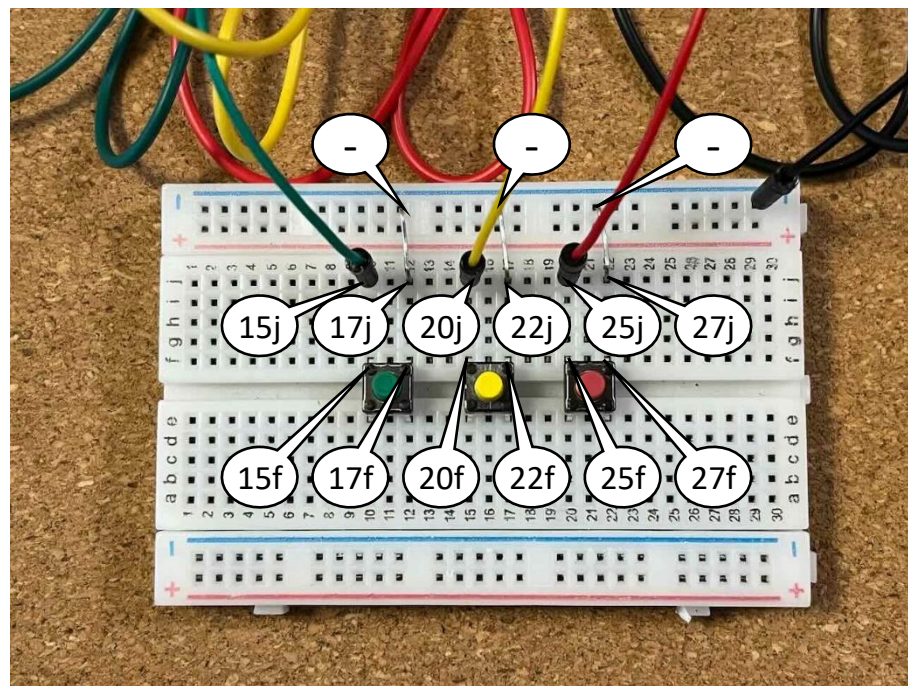
おまけ 1

- ブレッドボードやタクトスイッチを組み合わせると、タッチのかわりにスイッチをおして操作できるようになります。



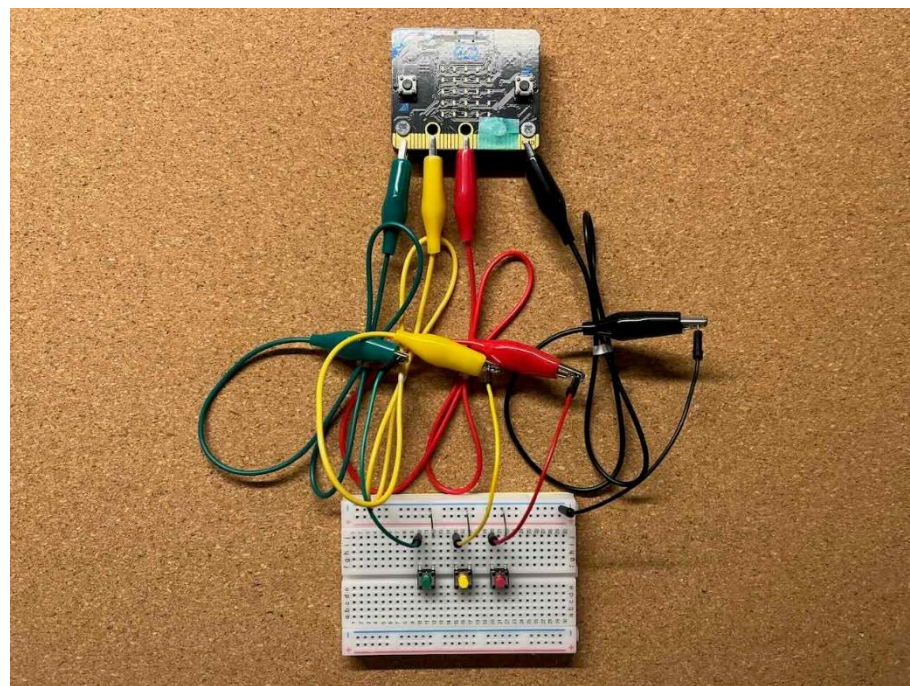
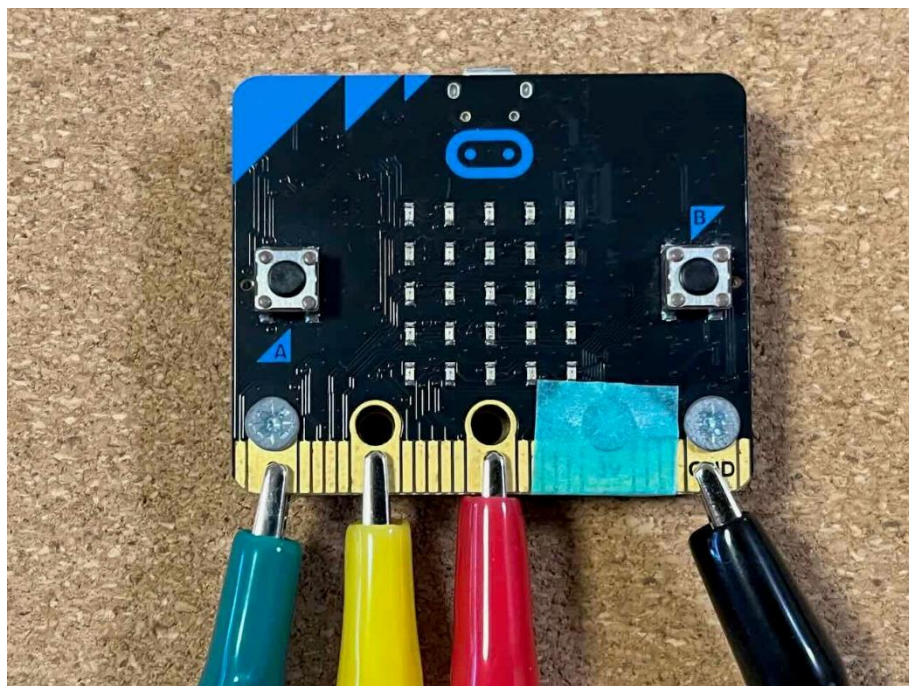
おまけ1 (つづき)

- すずメッキ線で、左の写真のような「コの字」型のパーツを3個つくります (1辺8mmぐらい)。
- 右の写真のようにつなぎます。



おまけ1 (つづき)

- ワニ口クリップ付コードをつかって、「緑」「黄」「赤」「黒」のジャンパーワイヤを、マイクロビットの「0」「1」「2」「GND」端子につなぎます。



おまけ 2

- 段ボールやベニヤ板などで、このようなものをつくれれば、ピコピコハンマーなどでたたけるようになり、本物の「もぐらたたきゲーム」のようになります。

