小中学生のための micro:bit を利用した

プログラミング教室（第２回）資料

氏　名＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

主催　NPO 法人 学習開発研究所

後援　京田辺市教育委員会

講習(1)：エレベータのシミュレーションをしてみよう（テーマ７）

レインボーパターンを点灯してみよう（テーマ６）

日　時：2025 年８月 10 日(日)　10:00～12:30

場　所：京田辺市中央公民館　２F　第３・４研修室

問い合わせ先：ild-kemsyu@u-manabi.org
担当者：NPO 法人 学習開発研究所　三輪、高橋

テーマ７　エレベータのシミュレーションをしてみよう
テーマ６　レインボーパターンを点灯してみよう

目　次

注）例題番号は、「小中学生のためのプログラミング～学習ガイド」の例題番号に対応しています。
本文中の例題の右端の、「☆」は応用、「★」は発展、「無印」で基礎の問題を示しています。

＜小中学生のためのプログラミング教室＞

https://u-manabi.net/ild-pkouza/

＜参考文献＞

高橋参吉、喜家村奨、稲川孝司：micro：bit で学ぶプログラミング、ブロック型から JavaScript そして Python へ、コロナ社(2019)

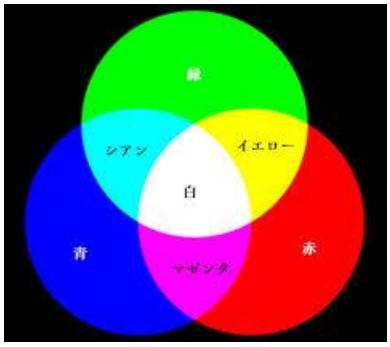
https://u-manabi.net/microbit/

１．フルカラーLED の制御

【Neopixel】

Neopixel は、１つのセルごとに赤（R）、緑（G）、青（B）の 3 つの LED と制御回路が入っており、フルカラーで光らせることができる LED の集合体である。

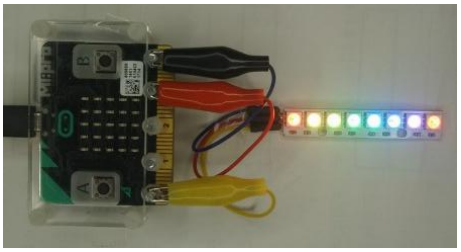
フルカラーは、色の明るさを、赤(R)、緑(G)、青(B)が、それぞれ 0～255 の 256 段階で選べるので、256*256*256=16,777,216 色の表現が可能となる。なお、R:255、G:255、B:255 では白、R:0、G:0、B:0 では黒になる。

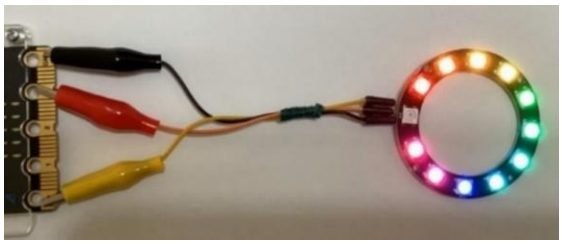


光の 3 原色　R(赤) G(緑) B(青)

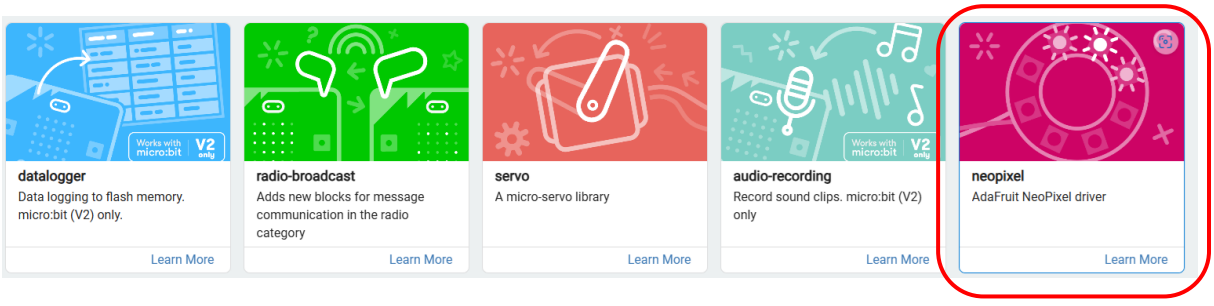
【micro:bit と Neopixel の接続】

micro:bit と Neopixel を下図のように接続する（黄色は P0 端子、赤は 3V 端子、黒は GND 端子）。Neopixel は、8 個の LED が棒状（Stick 型）（左）のに接続された製品で、その他にも、Neopixel には、リング状（右）の製品がある。



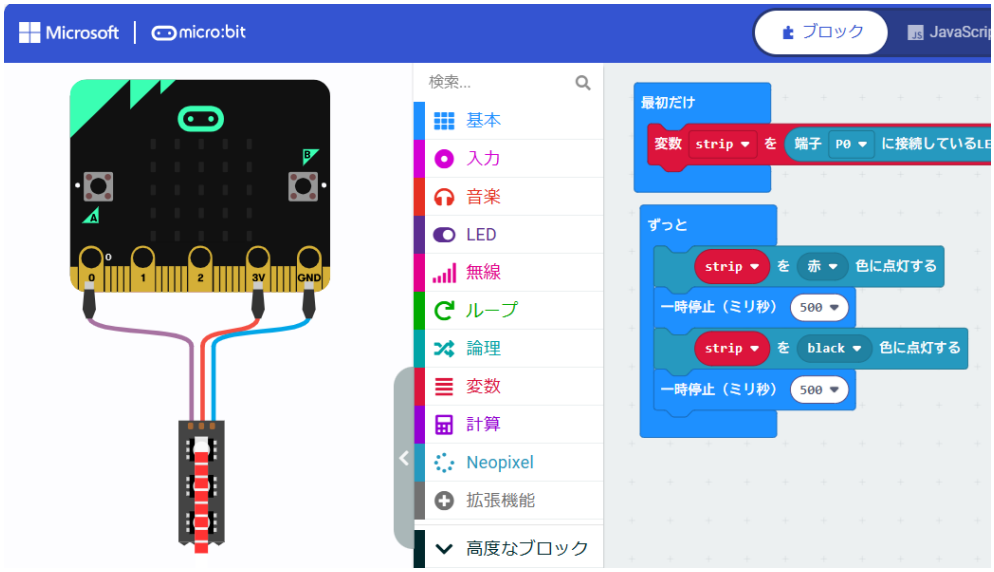


Neopixel を利用するには、ライブラリが必要である。ツールボックスの下にある「拡張機能」をクリックすると、拡張機能の一覧が表示されるので、「Neopixel」を選択する。ツールボックスの「計算」の下に、「Neopixel」のブロックが追加される。



【例題 1-1】Neopixel の点滅（プログラム prei6-4）

Neopixel(Stick 型)を赤色で点滅してみよう。



「最初だけ」ブロック

＊「Neopixel」－「変数 strip を 端子 P0 に接続している LED 個… にする」

ここで、LED 24 個→ 8 個」に変更しておく。

「ずっと」ブロック

＊「Neopixel」－「strip 赤色に点灯する」 色は、赤色のままにしておく。

＊「Neopixel」－「strip black 色に点灯する」 black では、消灯になる。

【例題 1-2】フルカラー表現（プログラム prei6-5）

「Neopixel」で、「RGB（赤 緑 青）」色に点灯する」に変更し、フルカラーで表現してみよう。

右図では、シアン（水色に近い青緑色）になる。

【例題 1-3】Neopixel の点灯（プログラム prei6-6）

Neopixel を 8 色で点灯させてみよう。

「最初だけ」ブロック

＊「Neopixel」－「変数 strip を 端子 P0 に接続している LED 個… にする」

ここで、LED 24 個→ 8 個」に変更しておく。

「ずっと」ブロック

＊「Neopixel」－「strip の 0 番目 色に設定する」 0~7 番目を図の色（赤・・・紫）にする。

＊「Neopixel」－「strip で設定した色で点灯する」

2. レインボーパターン

【例題 2-1】 レインボーパターンの点灯（プログラム prei6-8）☆

Neopixel のライブラリの中には、「レインボーパターンに点灯する」という命令がある。レインボーパターンを使って、虹色で光らせてみよう。

「最初だけ」ブロック 例題 1-1 に同じ（図は、省略）。

「ボタン A」

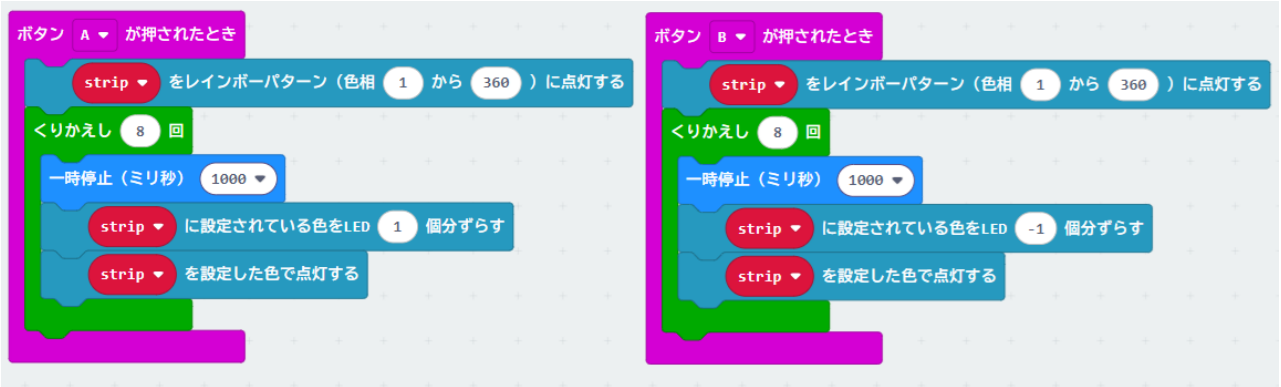
＊「レインボーパターン」(色相 1 から 360)に点灯する」を選択する。

＊下へ移動させるために、「strip に設定されている色を 1 個ずらす」を選択する。

「ボタン B」

＊「レインボーパターン」(色相 1 から 360)に点灯する」を選択する。

＊上へ移動させるために、「strip に設定されている色を－1 個ずらす」を選択する。



【例題 2-2】Neopixel の点灯（虹色）（プログラム prei6-9）☆

レインボーパターンを使って、リング状の Neopixel を虹色で光らせてみよう。ただし、A+B ボタンを押していると 10 倍の速さ（100ms）で順番に点灯し、押すのをやめると 1 秒で点灯するようにしよう。レインボーパターンの箇所は、「関数 ルーレット」とする。

「最初だけ」ブロック

使用するリング状の Neopixel は、LED は 12 個なので、今までの「LED 8 個の･･･」の箇所は、LED 12 個に変更する。

注）画面のシミュレータは、リング状で表示はされないが棒状で表示され、LED の個数は変更されている。

「ずっと」ブロック

＊「入力」から、「ボタン A が押されている」を選択し、A+B に直す

＊変数 s に、一時停止の時間を入れて、「関数 ルーレット」の引数とする

＊A+B が、押されているときは、s＝100 とし、そうでないときは、s＝1000 とする

「関数 ルーレット」

例題 4−4 のプログラム（A ボタン）を利用して、LED の個数、繰り返しの回数を 8 から 12 に変更し、停止時間を s にする。

関数作成手順の概略は、以下のとおりである。

＊ツールボックスの「高度なブロック」−「関数」を選択する。

＊「関数」−「関数を作成する･･･」が表示される。

＊「関数を作成する」を選択すると、「関数の編集」ダイアログが表示される。

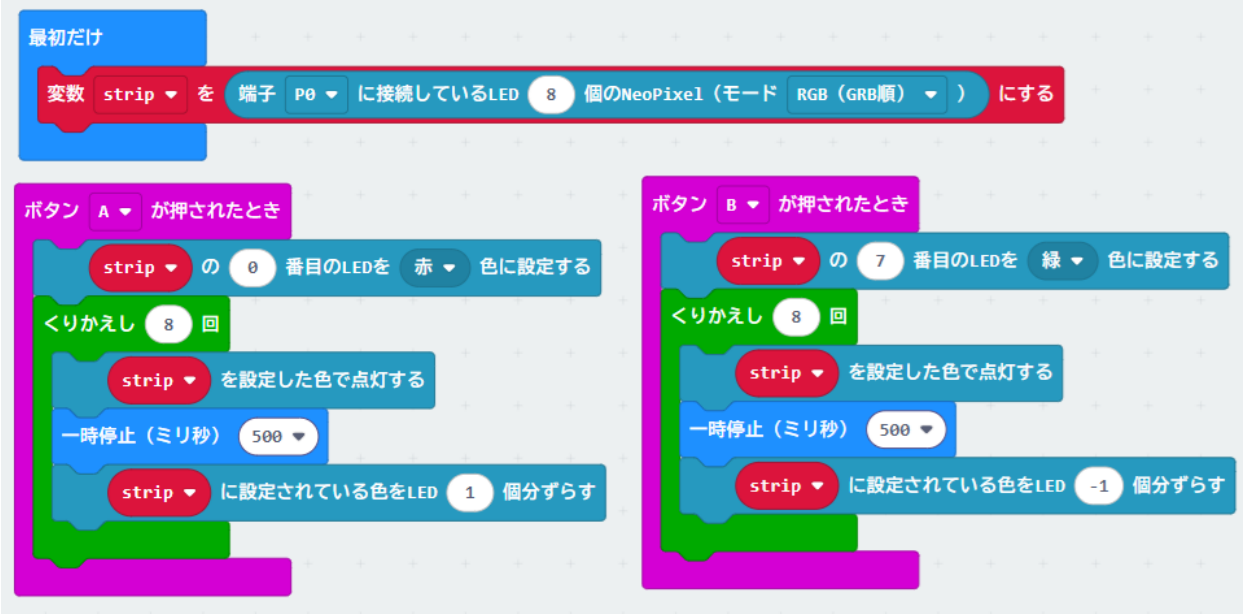
＊関数（function）の名称（doSomething の箇所）に名前を記入する。

「関数」は「function」と記載

3．エレベータのシミュレーション

【例題 3-1】Neopixel2 色の上下移動（プログラム prei7-1）

Neopixel を 2 色で点灯させ、2 色の上下移動してみよう。



A ボタンで、赤色が上から下へ、B ボタンで、緑色が下から上へ移動するように変更する。

「ボタン A」

＊「Neopixel」－「strip の 0 番目 赤 色に設定する」

＊「ループ」－「くりかえし 8 回」にする。

＊「Neopixel」－「strip で設定した色で点灯する」

＊「Neopixel」－「strip 設定されている色を 1 個ずらす」

「ボタン B」

＊「strip の 0 番目 緑 色に設定する」

＊「strip で設定した色で点灯する」

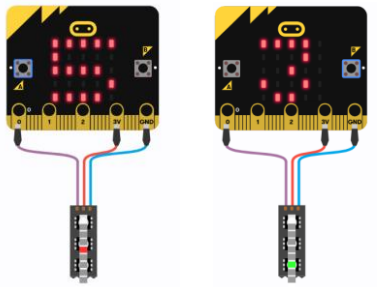
＊「strip 設定されている色を LED -1 個ずらす」とする。

【例題 3-2】Neopixel によるエレベータの表示（プログラム prei7-2）

エレベータの動きを Neopixel の上下移動を使って表示するプログラムを考えてみよう。

1 階から 8 階までの建物で、エレベータで移動できるものする。上行きの移動は、赤で表示し、下行きの移動は、緑で表示するものとして、つぎのような場合のプログラムを考えてみよう。

<条件>
・エレベータは、最初は 1 階（もしくは、8 階）にある。
・1 階で A ボタンを押すと、1 階から上へ移動して、5 階で止まる。
・8 階で B ボタンを押すと、8 階から下へ移動して、3 階で止まる。



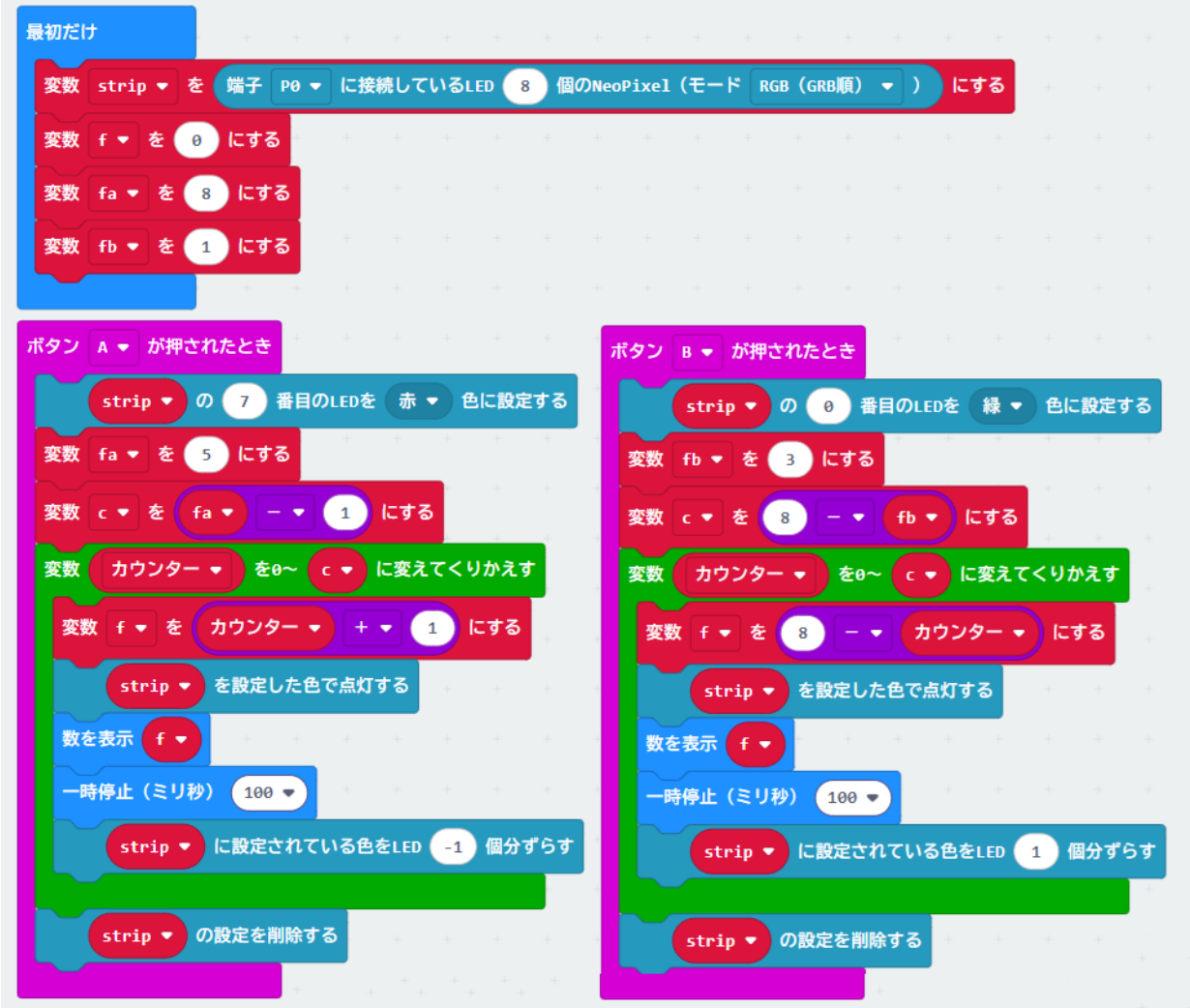
ここでは、例題 3-1 の考え方を利用して作成する。

「最初だけ」ブロック

f、fa、fb は、エレベータの位置に関係する変数である。

＊初期設定では、変数 f は 0 とする。

＊変数 fa は最上位の階で 8、変数 fb は最下位の階で 1 とする。



「ボタン A」

A ボタンを押すと、赤が上へ移動する。

＊「strip の 7 番目の LED 赤色に設定する」では、一番下の LED を赤に設定する。

＊fa を 5 にして、変数カウンターを 0 から 4 までとして、5 回繰り返す。

＊変数 f で、現在のエレベータの位置（カウンター ＋ 1）を表示する。

＊strip で赤色を点灯させ、設定されている色を LED −1 個分ずらすことにより、上に移動する。

「ボタン B」

B ボタンを押すと、緑が下へ移動する。

＊「strip の 0 番目の LED 緑に設定する」では、一番上の LED を緑に設定する。

＊fb を 3 にして、変数カウンターを 0 から 5 までとして、6 回繰り返す。

＊変数 f で、現在のエレベータの位置(8 − カウンター)を表示する。

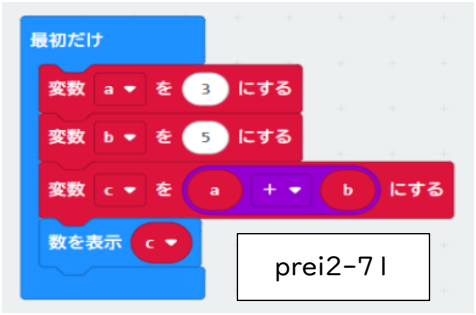
＊strip で緑色を点灯させ、設定されている色を LED 1 個分ずらすことにより、下に移動する。

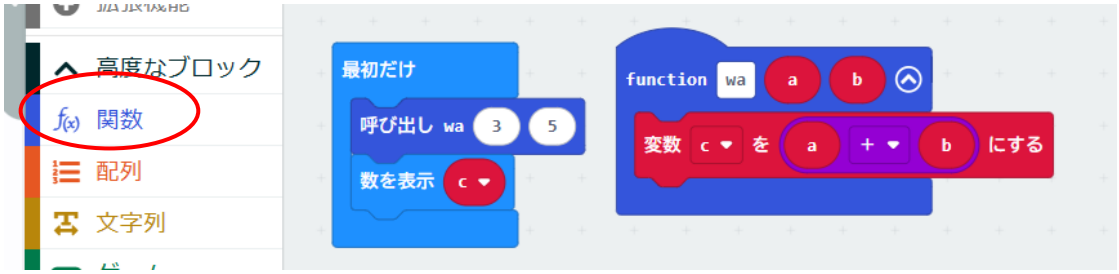
<参考> プログラムの応用（関数）

関数は、一定の処理をまとめたもので、プログラムの中で必要に応じて呼び出される。関数には、プログラミング言語に初めから用意されている組み込み関数(例えば、乱数)とプログラムの作成者が用意するユーザ定義関数がある。ここでは、後者のユーザ定義関数を関数という。

【例題 2-7】足し算（関数）（プログラム prei2-7）☆

ａ（３）と ｂ（ ５ ）を足して、その和を ｃ（ ８ ）とするプログラム（prei2-71）を、関数を利用したプログラムにしてみよう。





「関数 wa」を下記の作成手順に従って作成すると、関数のブロックが作成される。また、作成後のツールボックスには、右図のように、「戻る」と「呼び出し」が作成されている。

<関数の作成手順>

＊ツールボックスの「高度なブロック」－「関数」を選択する。

＊「関数」－「関数を作成する…」が表示される。

＊「関数を作成する」を選択すると、「関数の編集」ダイアログが表示される。

＊関数（function）の名称（doSomething の箇所）に名前（ここでは、wa）を記入する。

＊パラメーターを追加する箇所（ここでは、数値）を選択して記入(ここでは、a と b)し、完了する。

<関数の引数と戻り値>

「関数」ブロックの「wa」を関数名、変数 a、b は関数に引き渡されるパラメータ（この例では数値）で、引数という。また、関数が実行されたときに返される値を戻り値という。この例では、戻り値はない。