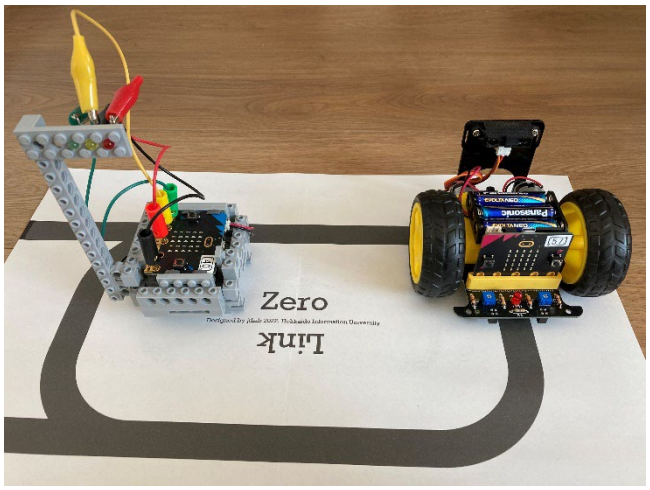
2022 年 10 月 13 日　クラーク記念国際高等学校 旭川キャンパス

レゴで体験するロボットプログラミング

北海道情報大学　棚橋研究室



今日の目標：

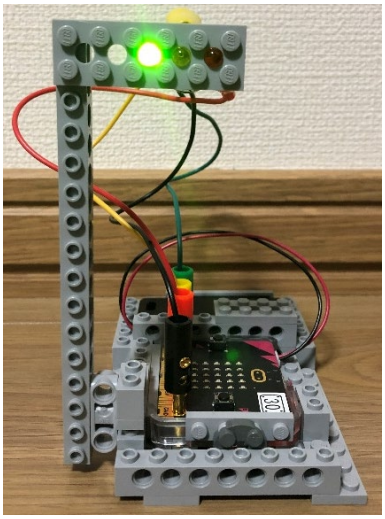
✧ micro:bit で動作する自動運転車と信号機に対して、Minecraft でも使われている Scratch のようなブロック型言語「MakeCode」を用いてプログラミングします。

✧ 車はラインに沿って走りますが、交差点では信号を守らなければいけません。

✧ コースを連結して複数台の車をぶつからず走らせることができるでしょうか？　時間があればチャレンジしてみてください！

ミニ信号機の基本的な仕組み：

信号機の micro:bit には、GND(黒)に LED のマイナス電源、P0・P1・P2 に緑・黄・赤それぞれのプラス電源が繋がっています。



信号の光り方（変数 signal）には、４種類あります。

- 0 … 全ての LED が点灯する（電源投入時のみ・LED のチェック用）
- 1 … 緑色の LED（P0）だけ点灯する
- 2 … 黄色の LED（P1）だけ点灯する
- 3 … 赤色の LED（P2）だけ点灯する

実際の信号機では、長い時は数分経たないと次の色には変わりませんが、今日はそんなに長くすると大変なので、

- 信号を青(緑)にする
- ５秒間（5000 ミリ秒）待つ
- 信号を黄色にする
- ２秒間（2000 ミリ秒）待つ
- 信号を赤にする
- ５秒間（5000 ミリ秒）待つ

というような動作をさせています

最初のプログラムは３ページにあります。

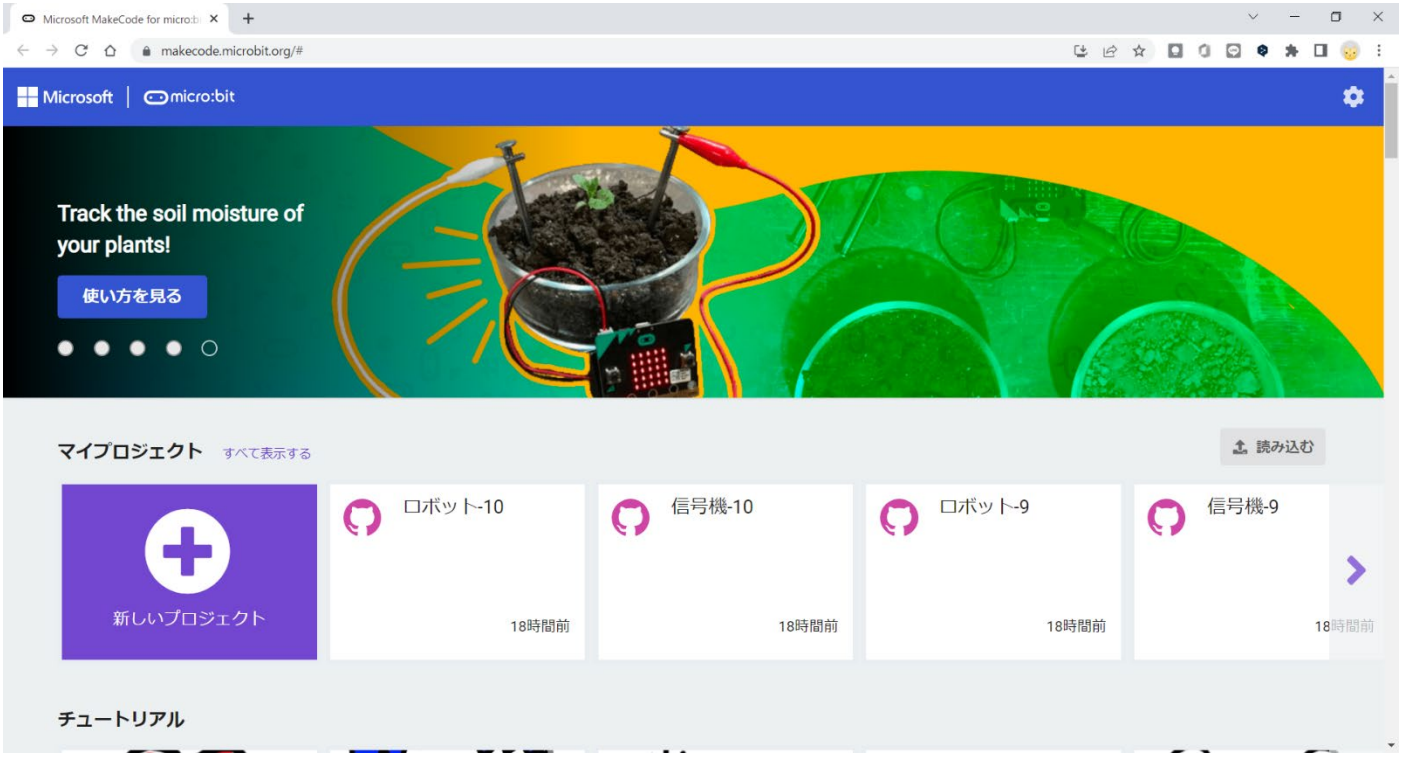
各 LED の明るさは 1024 段階で調整できます。

- 青（緑）　…　端子 P0　値 1000
- 黄　　…　端子 P1　値　500
- 赤　　…　端子 P2　値　100

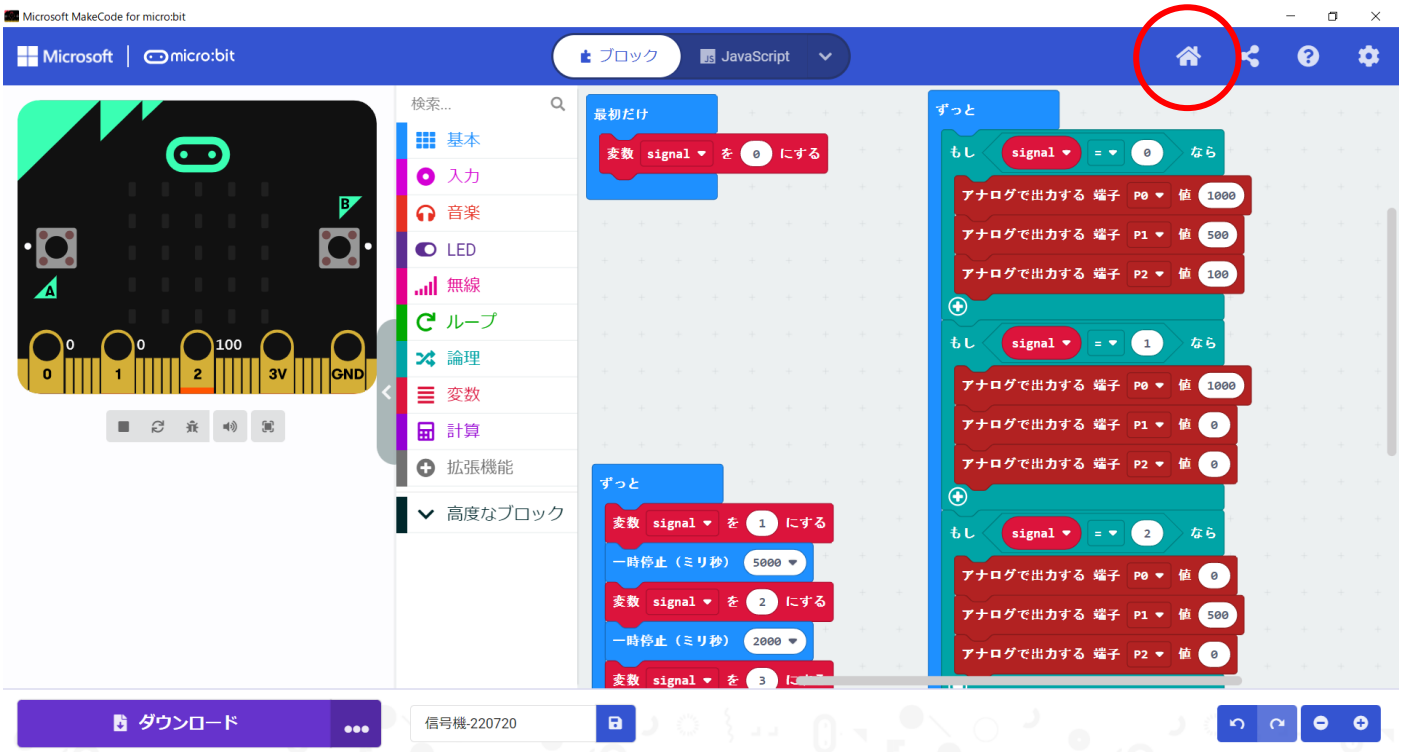
上記の数字を指定すると**点灯(オン)**、

0(ゼロ)を指定すると**消灯(オフ)**します。

MakeCode の基本的な使い方：

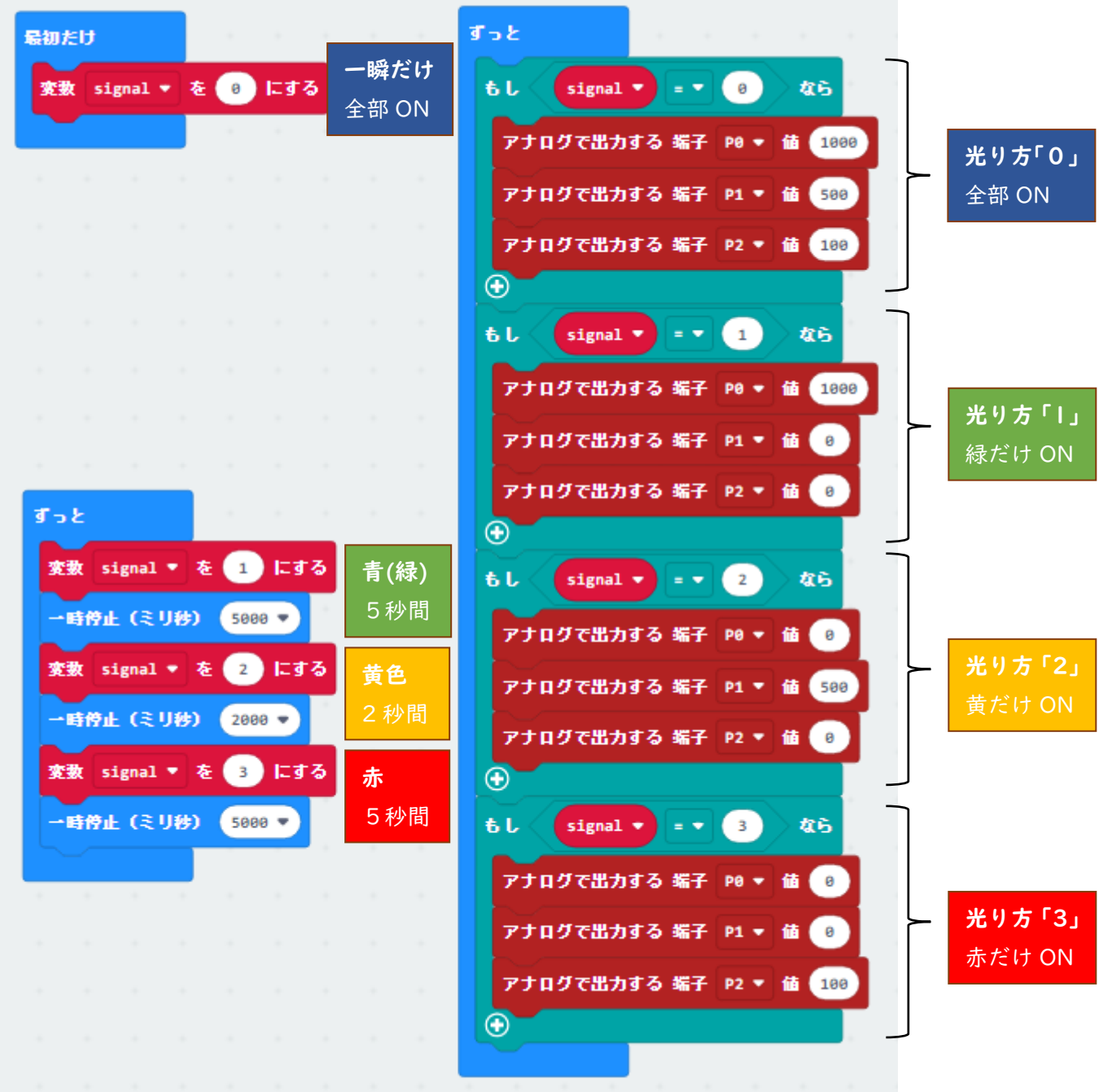


トップ画面はこのような構成になっています。「読み込む」から「https://github.com/clark-share/robot チーム番号・signal チーム番号」を読み込んで、「**ロボット**-チーム番号」と「**信号機**-チーム番号」を用意してください。



例えば信号機のプログラムはこのように、ブロックが積み重ねられたり、口の中にくわえられていたりしている構造になっています。プログラムを切り替えるには、ホームボタンを押して戻ってください。

信号機のプログラム



左上の「最初だけ」のブロックは、電源が入ったとき、最初に一回だけ実行されます。LED が故障していないか確認するため、signal を 0 にして一瞬だけ全て点灯させます。

左下の「ずっと」ブロックでは、1 ページ目で説明したとおり、青（1）にして、5 秒待って、黄色（2）にして、2 秒待って、赤（3）にして、5 秒待つ、という基本的な動作が並んでいます。この様に、上から下に順番に実行する**順次処理**を、「ずっと」**反復処理**させることで、信号が変わり続けます。

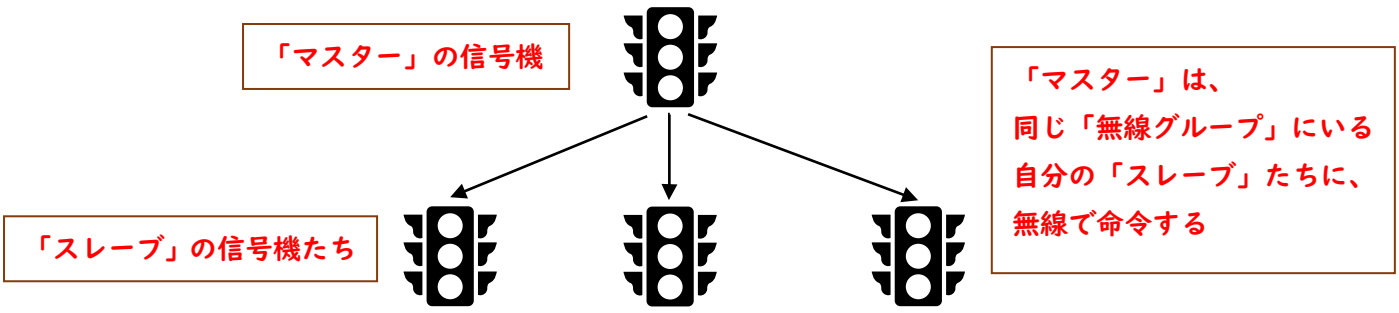
右側の「ずっと」ブロックでは、変数 signal を通じて命令された 0 ～ 3 の数字に従って、それぞれの時にどの LED を点灯させたり消灯させたりするのかを、「もし～なら」の**分岐処理**を用いて実行しています。

ここを変えると、レースのスタートシグナルとかも作れそうですね。トライしてみてください。

信号機をすべて同じ色にするための考え方：

例えば向かい合った信号機を同じ色にしようと思ったら、人間なら「いっせーの！」で同じ色にするとよいです。でも、コンピュータはそういうのが苦手で、命令されたことをそのままやる方が得意です。

そこで、それぞれの信号機を、「命令を出す信号機（**マスター**）」と「命令に従う信号機（**スレーブ**）」の２つの役割に分けて、スレーブの信号機はマスターの信号機が指示した色を点灯させるだけにします。

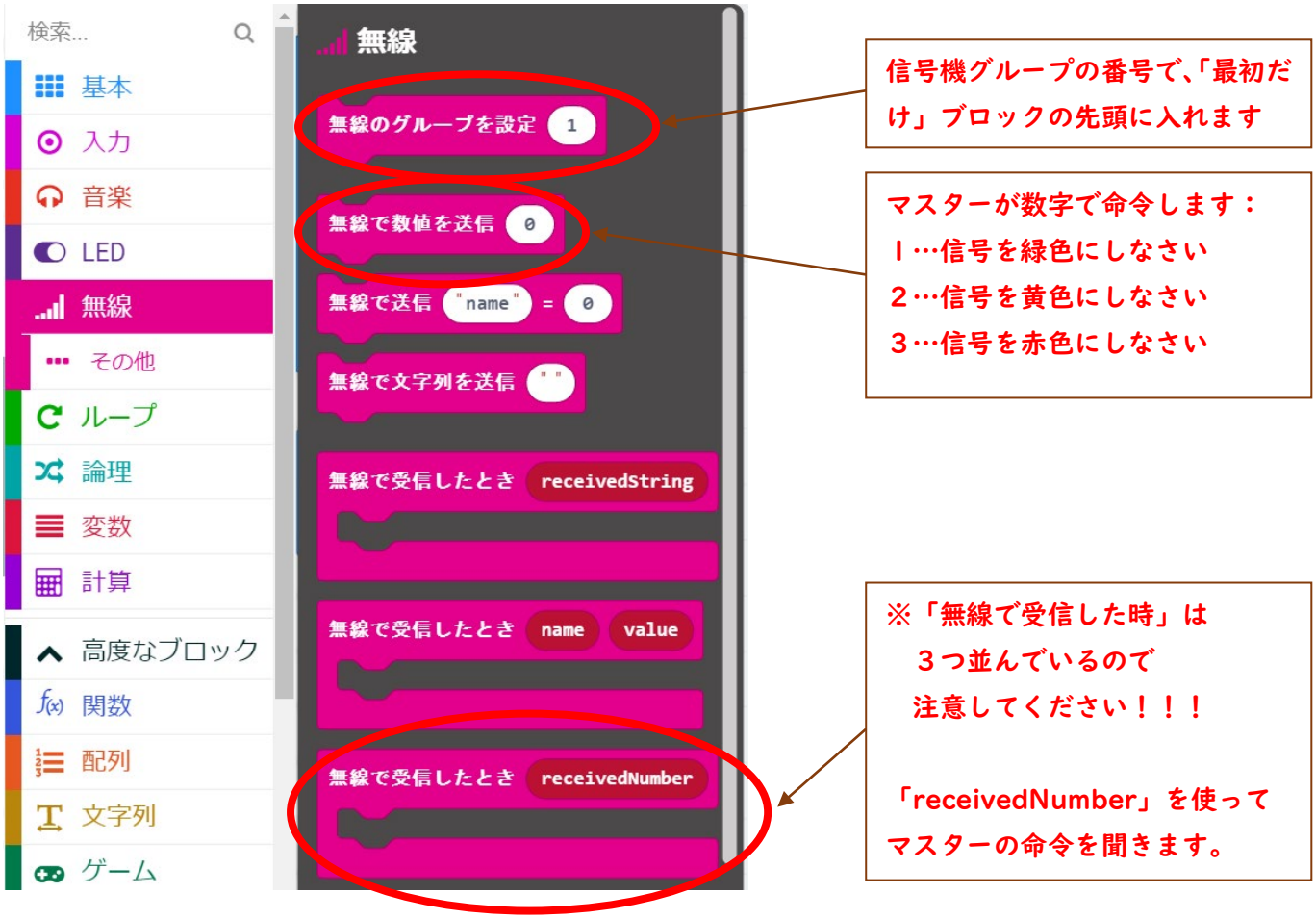


「無線」ブロックを使う：

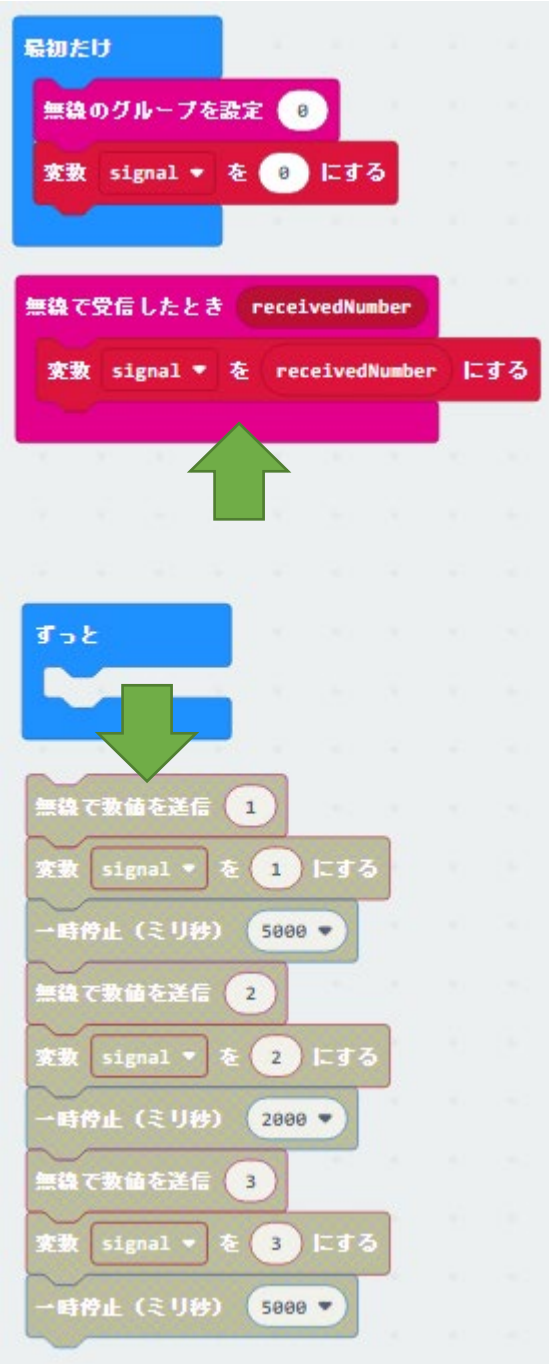
まずは先生の信号機を「マスター」、みなさんの信号機をグループ０の「スレーブ」として試してみます。

マスターは、「無線で数値を送信」すると、その数字をすべてのスレーブで受信できます。

スレーブは、「無線で受信した時 received**Number**」で、マスターからの命令を受信できます。



スレーブのプログラム



マスターのプログラム

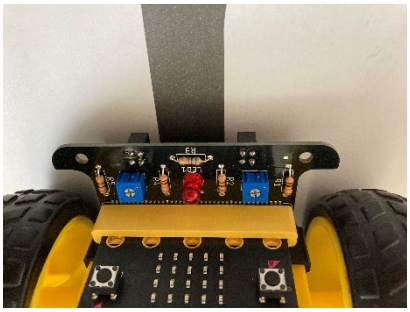
※指定された光り方によって LED を点灯させるための左側の「ずっと」ブロックは、掲載を省略していますが両方に配置されています。

スレーブは「無線の受信」側を、マスター側は「無線の送信」側をそれぞれ C 型ブロックの中に入れて有効にしたり、ブロックの中から出して無効にしたりして役割を切り替えてください。

ここまでできたら「ダウンロード」ボタンを押して、先生の信号と同じように光るかどうか確認してください。

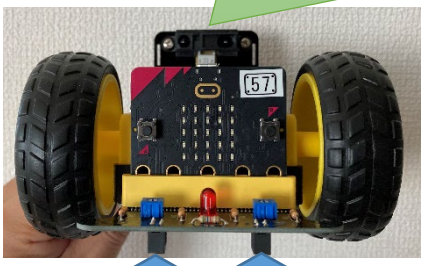
車の基本的な仕組みとプログラム：

車体前面には、１対のフォトリフレクタがついています。光の反射を利用して、白っぽい（良く反射してくる）か、黒っぽい（あまり反射してこない）かを判断します。



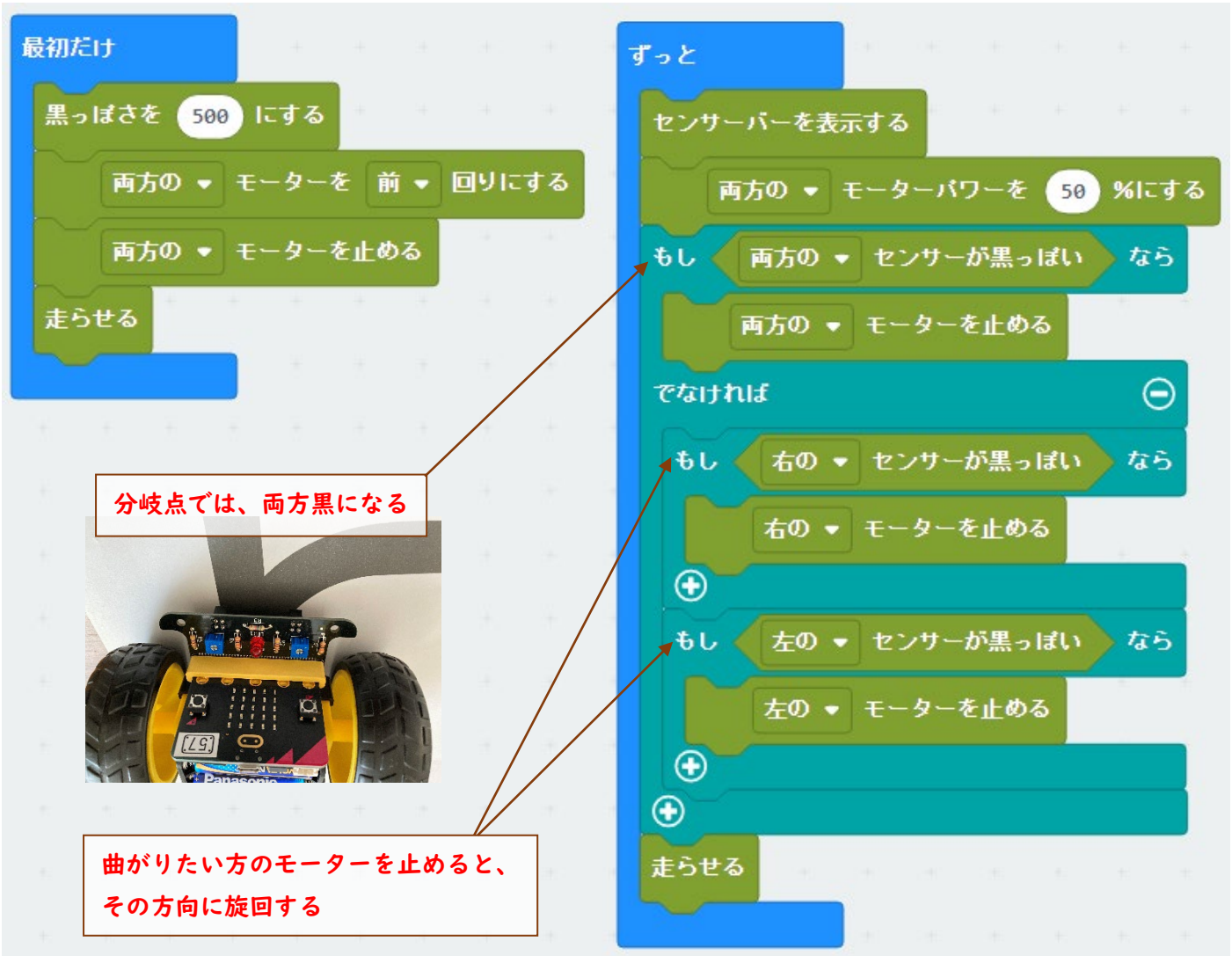
赤外線測距センサー（障害物センサー）

目に見えない光を出し、それが障害物に反射して戻ってくるまでの時間から、距離を計測します。

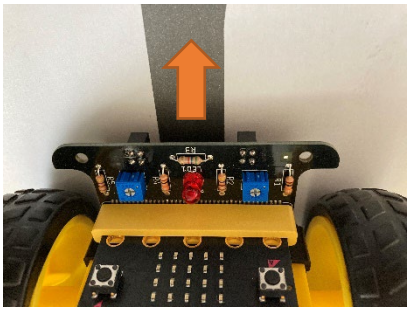


フォトリフレクタ（右センサー・左センサー）

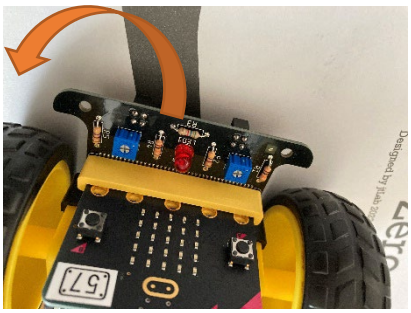
赤外線を床に出して、白っぽいか黒っぽいかを計測します。



ライン上を走る仕組み：

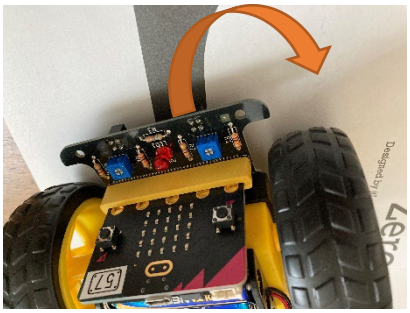


左の写真では左右のフォトリフレクタが両方白の上にいますので、この場合は直進します。



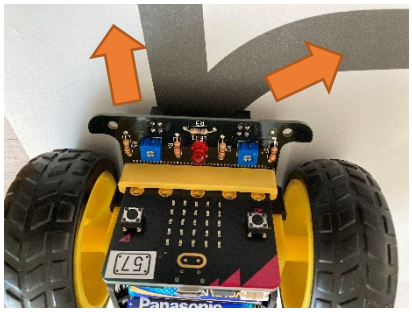
右の写真では、ちょっと傾いて、左フォトリフレクタがラインに掛かっています。

このままいくとラインを外れてしまうため、黒だと判断しているフォトリフレクタと同じ側、左に向かってちょっと旋回します。曲がりたい方のモーターを遅くする（このプログラムでは「止める」）と旋回します。



旋回しすぎると、今度は右フォトリフレクタが黒い線に乗ってしまいました。この場合は右に曲がります。

これを繰り返すと、車はラインに沿って走ります。



コースには分岐点があり、ここでは両方のフォトリフレクタが黒だと判断します。

分岐点を右に進み、回り続ける：

分岐点を越えて進み続けるためには、直進の場合は左センサーのみ、右折の場合は右センサーのみを使って走行しなければいけません。

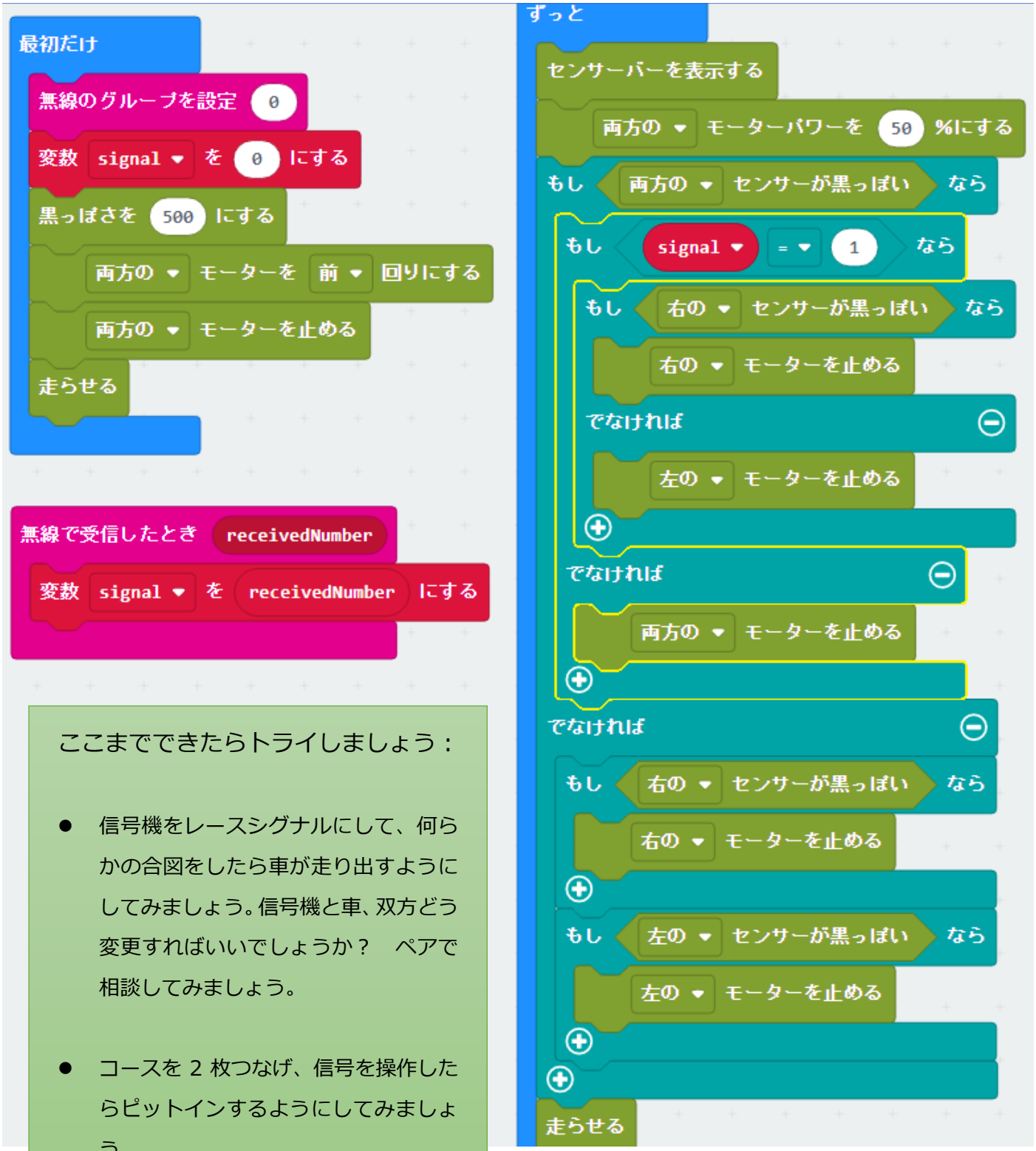
さきほどのプログラムでは、「両方のセンサーが黒っぽい」分岐点に差し掛かった場合、「両方のモーターを止める」にしていましたが、このブロックの代わりに「もし～なら～でなければ」ブロックを入れて、右センサーだけを使ってライン上を進むように変更します。

※この資料では時計回りに走行させています。
反時計回りに走行させる場合は
左右を逆にする必要があります。

信号機に従って止まるプログラム：

信号機と同じように、無線関係のブロックを追加し、青信号だった場合のみ右折する黄色枠で囲まれたブロックも追加します。

信号機側のプログラムは「**マスター**」に、信号機・自動運転車双方の「無線のグループを設定」は「**信号機と同じ番号**」を設定してください。



ここまでできたらトライしましょう：

- 信号機をレースシグナルにして、何らかの合図をしたら車が走り出すようにしてみましょう。信号機と車、双方どう変更すればいいでしょうか？　ペアで相談してみましょう。
- コースを 2 枚つなげ、信号を操作したらピットインするようにしてみましょう。