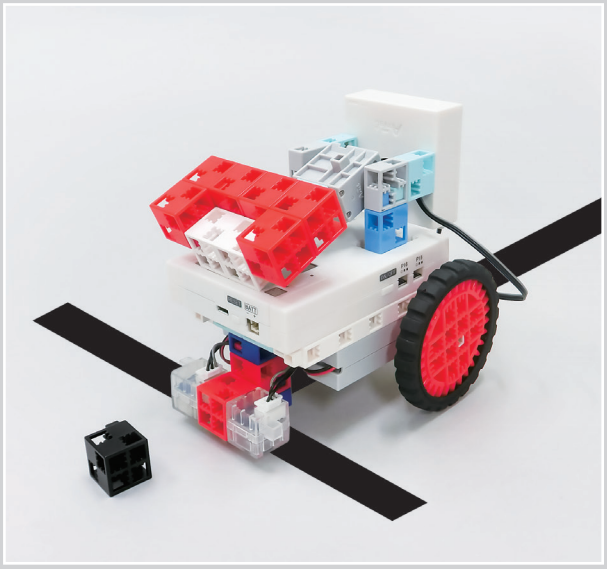
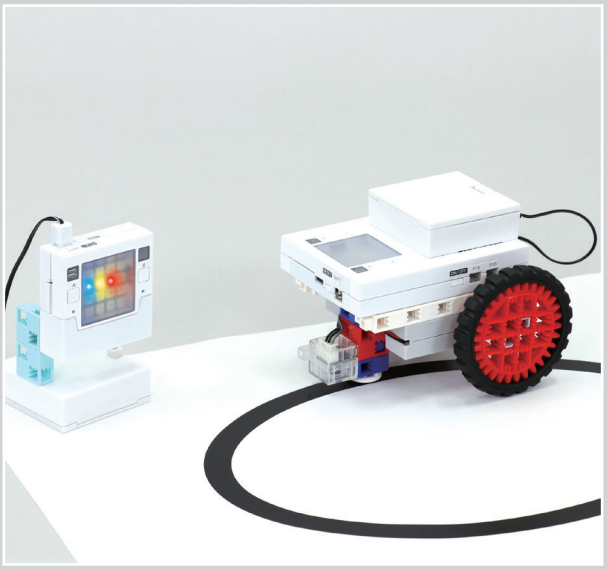
ArTeC® Robo 2.0
アーテックロボ

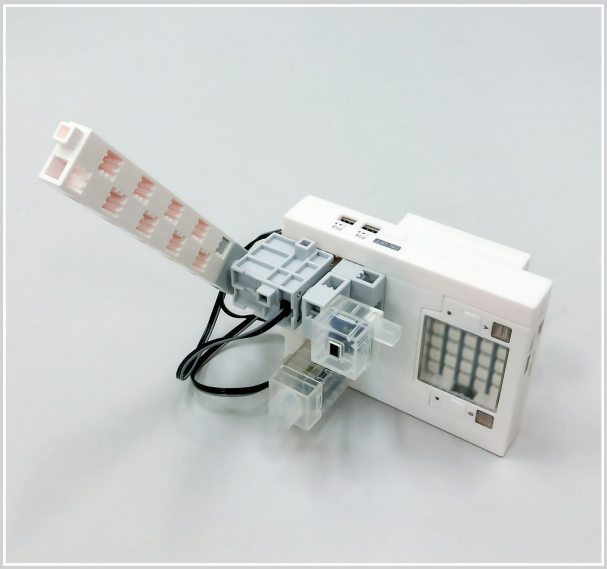
計測・制御のプログラミングによる問題解決

ベーシック/アドバンスキット用指導書

教員用







目 次

部分はシンプルキットの内容には含まれていません。

注意事項

端末と接続状態でのご使用時

Studuino:bit にプログラムを転送させずに、端末と Studuino:bit を USB もしくは Bluetooth で「接続」した状態で端末からプログラムを実行する場合、プログラムを Studuino:bit に転送して実行した場合にくらべて、プログラムの処理に遅延が生じる場合があります。この遅延はアプリ上でのプログラムの処理速度および端末と Studuino:bit 間の通信により発生します。センサーの値に応じて自動車を制御するといった、処理の遅延が動作に影響するようなプログラムを作成する場合は、プログラムを転送させて動作確認を行ってください。

乾電池のご使用時

乾電池が消耗してくると、基板にリセットがかかりやすくなります。とくに DC モーターを回転させる瞬間に、最も電圧降下が激しくなり、基板にリセットがかかって意図しない動作をしたり、Bluetooth の接続が切れたりすることがあります。また、電源を入れたまま保管してしまうと、待機電力により意図せず電池の消耗が起こることがあります。保管時は電池ボックスのコネクタを抜いておくようにしてください。

iPad/Android/Chromebook用Bluetooth通信対応のアプリご使用時

Bluetooth 通信対応アプリにおいては、USB での端末と Studuino:bit の通信に対応していないため、以下の点を注意いただく必要がございます。あらかじめご理解ください。

①ファームウェア更新について

ファームウェア更新時、端末と Studuino:bit が Wi-Fi 接続されます。
Studuino:bit は Wi-Fi による更新時に下記の状態で動作します。

Studuino:bit 内アクセスポイント SSID: Stu:bit0192837465
Studuino:bit 内 WEB サーバー IP: 192.168.4.1

端末の管理システム等により Wi-Fi の接続先やサーバーのアクセス先に制限がかかっている場合は、ファームウェア更新が正常に行えません。
アクセスを許可していただくか、Windows 用ソフトウェアを用いて USB ケーブル経由でのファームウェア更新を実施してください。

②Bluetooth接続時の機能制限について

右記の無線ブロックについてはプログラムを Studuino:bit へ転送することで機能しますが、端末と Studuino:bit が Bluetooth 接続されている状態では機能しません。

ArtecRobo2.0サポートガイドはこちら

https://www.artec-kk.co.jp/artecrobo2/pdf/jp/artecrobo2_supportguide.pdf

導 入

身近なプログラミング

＜学習内容＞

●身近な計測・制御システム

●プログラミング環境の基本操作

1．身の回りの計測・制御

私たちの身の回りにある電気製品は、計測・制御の仕組みを使って自動的に様々な仕事をしています。決まった動作を繰り返したり、外部の情報を元に柔軟に対応したりすることができるのは、コンピューターやセンサーが使われているからです。このような計測・制御システムは、センサー／コンピューター／アクチュエータの三つの要素から構成されています。エアコンの例を見てみましょう。

例 エアコン

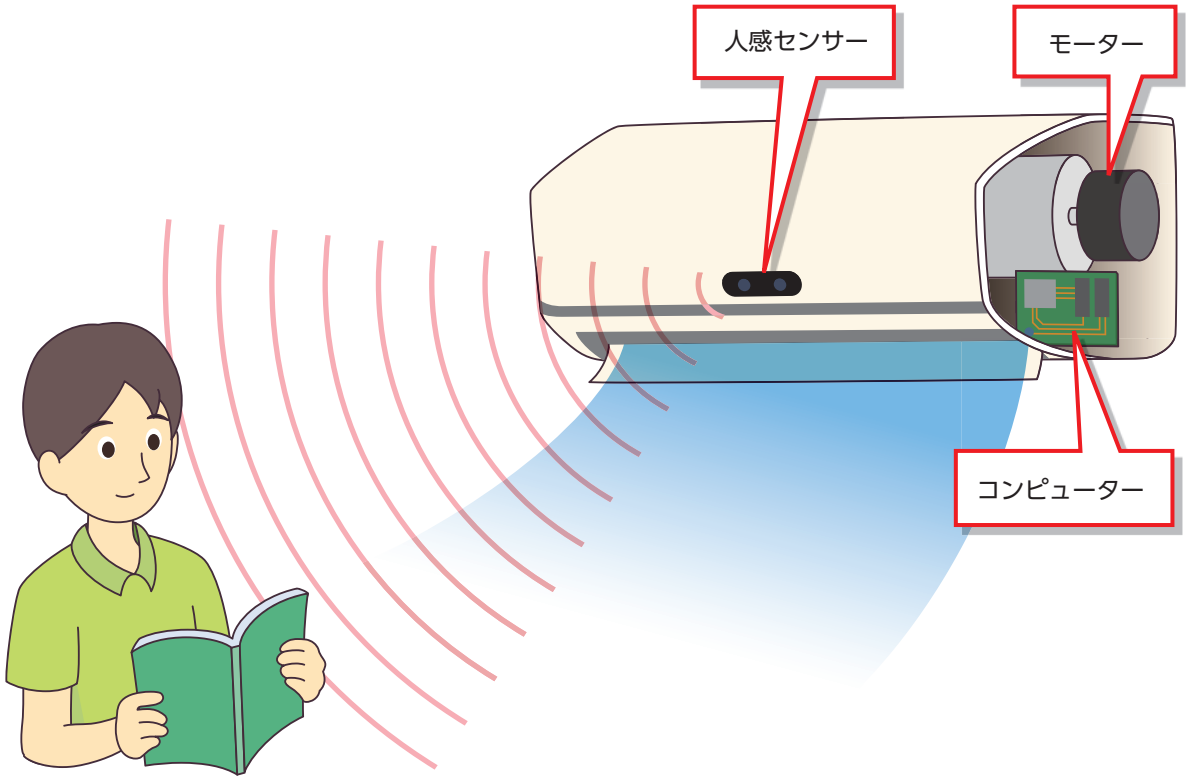
センサー
人感センサー（赤外線センサー）・温感センサーで人の位置や体温・室内の温度を計測する。

→

コンピューター
センサーで感知・計測した情報を処理・判断する。それを元に温度や風力を設定しモーターへ指示を出す。

→

アクチュエータ
コンピューターからの指令を受けてモーターの回転数を変え温度の調整をする。



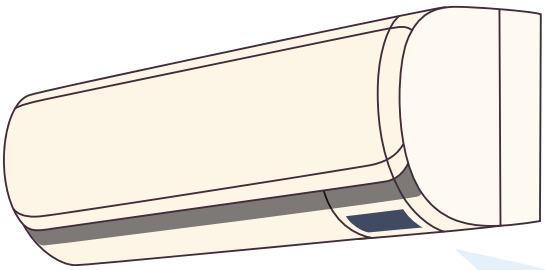
身近な製品でどういうものがコンピューターやセンサーを活用しているか探してみましょう。

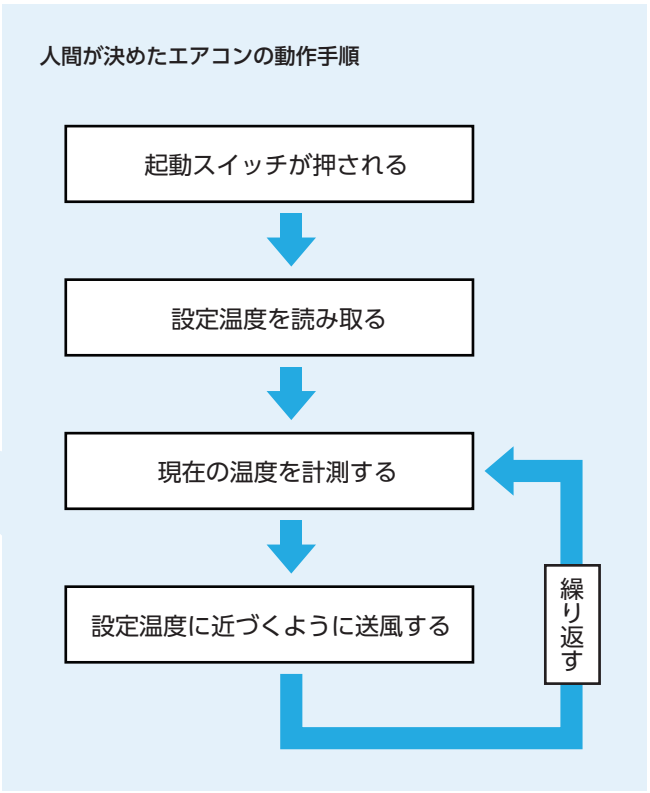
製品名	センサーで計測しているもの	計測結果からコンピューターが行う動作
風呂給湯器	水量	給湯する / 給湯を止める
お掃除ロボ	壁や障害物の有無	壁や障害物を避けるように動く

2．プログラミングとは

エアコンの例のように、コンピューターが使われている機械はセンサーから得た情報を元に、あらかじめ人間が決めた手順通りにアクチュエータを動かします。

例 エアコンを動かすとき





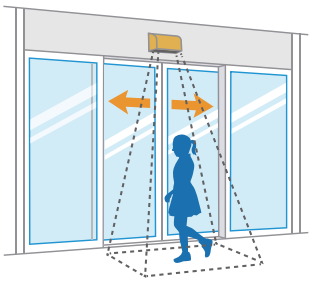
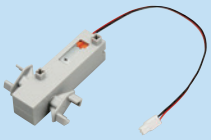
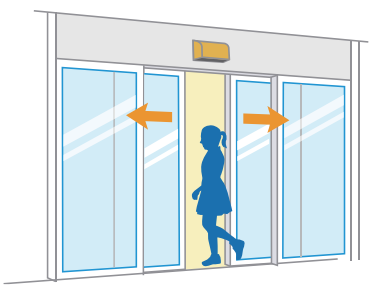
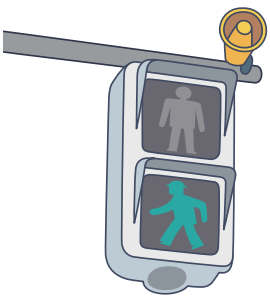
このようにコンピューターの動作手順を示したものをプログラムと言います。プログラムは人間が話すときに使う言葉とは違う、特別な言葉を使って表します。この言葉をプログラミング言語といい、プログラミング言語でプログラムを書くことをプログラミングと言います。

```
from pystubit import *
import machine

if lightsensor.get_value() > 300:
    xon = False

while True:
    if button_a.is_pressed() is True:
        bcount = bcount+1
        if bcount == 4:
            bcount = 1
    elif button_b.is_pressed() is True:
        bcount = bcount-1
        if bcount == 0:
```

▲プログラミング言語で書かれたプログラムの画面

3．計測・制御システムとArtecRobo2.0パーツの対応

ArtecRobo2.0のパーツは以下のようなコンピューターによる計測･制御システムで使われる各要素に対応しています。

	自動ドアの動作の流れ	押しボタン式信号の動作の流れ
		
センサー 周囲の情報を計測する	 **赤外線フォトリフレクタ** 物体の有無を調べる	 **ボタン** 押されたことを調べる
コンピューター センサーで得た情報を判断しアクチュエータに動作命令を出す	 **メインユニット(Studuino:bit)**	
アクチュエータ コンピューターからの命令に従って動作する	  **DCモーター**　**サーボモーター** 電気を回転する力に変える	 **LEDライト**　**ブザー** 電気を光に変える　電気を音に変える
		

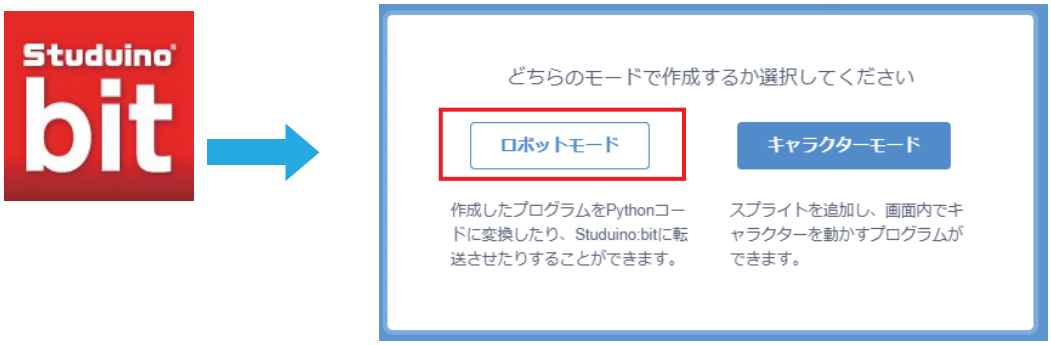
※アクチュエータとはエネルギーを動きに変えるものを指すため、動きのないLEDやブザーはアクチュエータに含まれません。

4. プログラミング環境

この授業では、文字の代わりにブロックのような絵をつないでコンピューターへの命令をプログラミングできる「ビジュアルプログラミング言語」を使います。

①ソフトウェアの立ち上げ

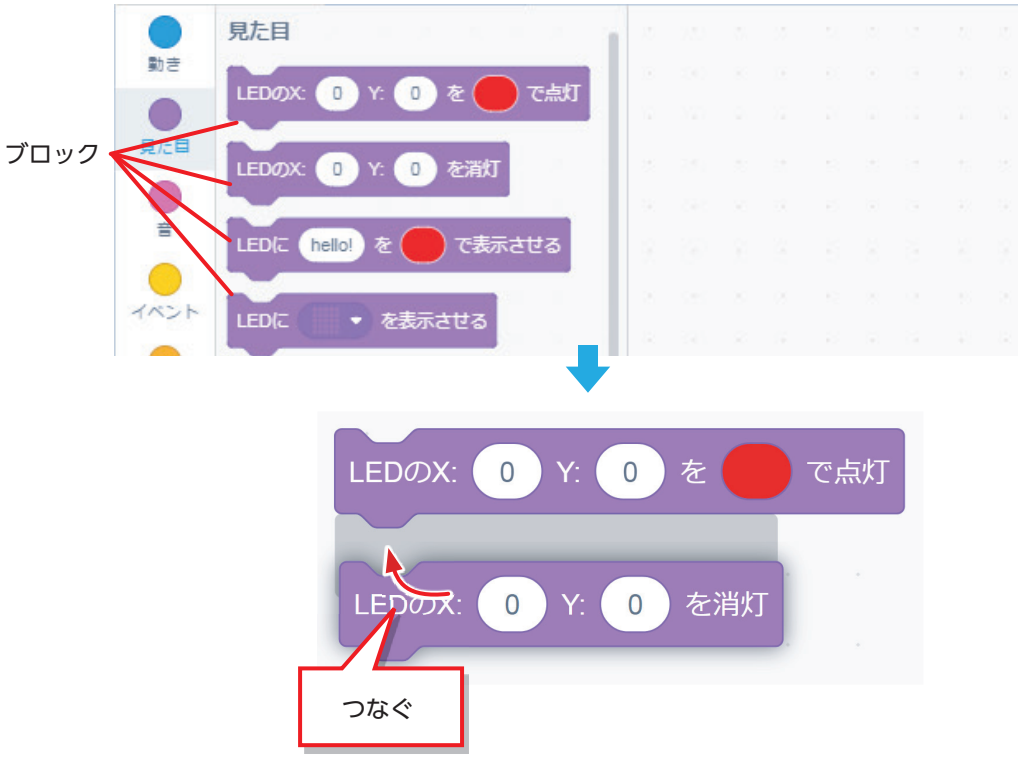
ロボットモードを選択してください。



②操作方法

◆ プログラムの作成

ブロックパレットにある命令をおもちゃのブロックのようにつなぐことでプログラムをつくります。この命令のひとつひとつを「ブロック」と呼びます。



◆ ブロックの削除

削除するブロックをブロックパレットにドラッグ＆ドロップします。

カテゴリーの種類について

カテゴリーは「動き」「見た目」「音」「イベント」「制御」「調べる」「無線」「演算」「変数」「関数」の
10種類があります。それぞれのアイコンをクリックすることで、カテゴリーを選択することができます。

DC モーターやサーボモーターなど、アクチュエータの動きを命令するブロックです。
起動時、ブロックは表示されていません。 後述の「入出力設定」で使用可能になります。

LEDの点灯・消灯の命令をするブロックです。

ブザーを鳴らす命令をするブロックです。

プログラムスタートを行うブロックです。

命令の実行順を制御するブロックです。

センサーを指定して周りの情報を調べるブロックです。

無線で送受信を行うブロックです。

計算の命令を出したり、条件式を作成するブロックです。

数値情報の記録をする変数やリストブロックを作成します。

関数としてブロックを作成します。

基本学習 テーマ 1

LED の制御

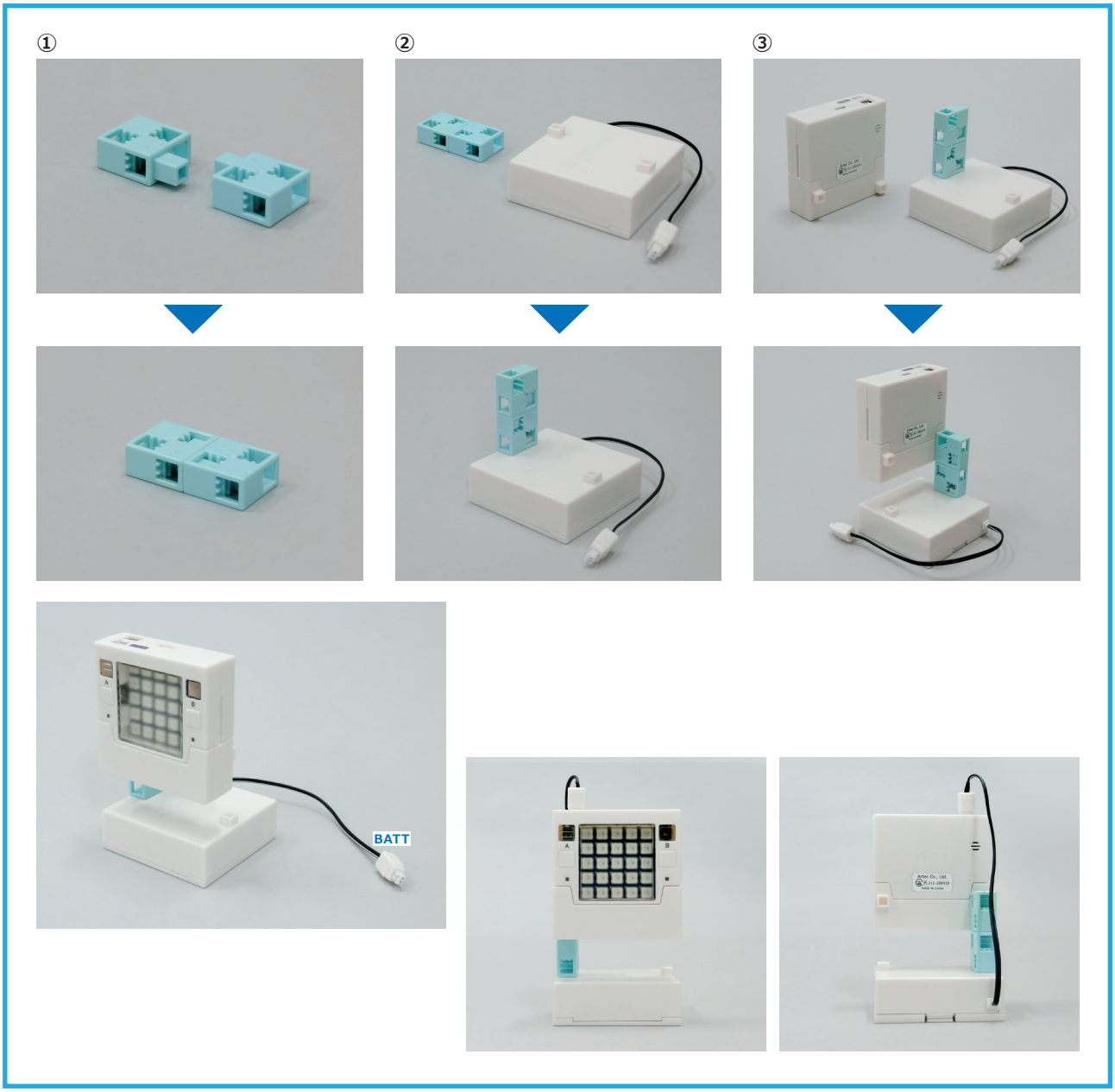
＜学習内容＞

- ●LED の制御 / ブザーの制御
- ●順次処理
- ●繰り返し処理
- ●条件分岐処理

1. 順次処理のプログラム

命令を順番通りに行う処理のことを「順次処理」といいます。LED を使った信号機をつくる中で、順次処理のプログラムを学びましょう。

①「信号機」の組み立て



②コンピューターとの通信

Windows/Mac の場合

USB ケーブルによる通信

①コンピューターとメインユニット
(Studuino:bit) を USB ケーブルで接続します。

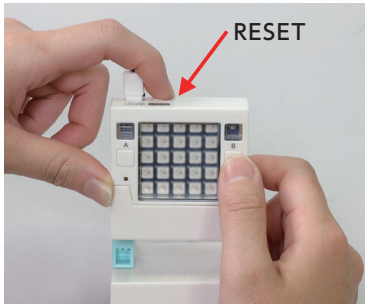
②「編集」から接続を選択します。

Android/iPad/Chromebook の場合

Bluetooth による通信

①「編集」から接続を選択します。

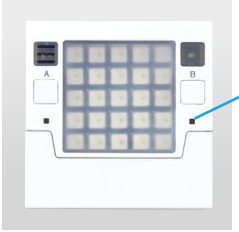
②表示されるメッセージに従い、
メインユニット(Studuino:bit) の B ボタンを
押しながらリセットボタンをおしてください。



③メインユニット(Studuino:bit) の
LED に表示された点灯パターンと同じ
デバイスを選択してください。

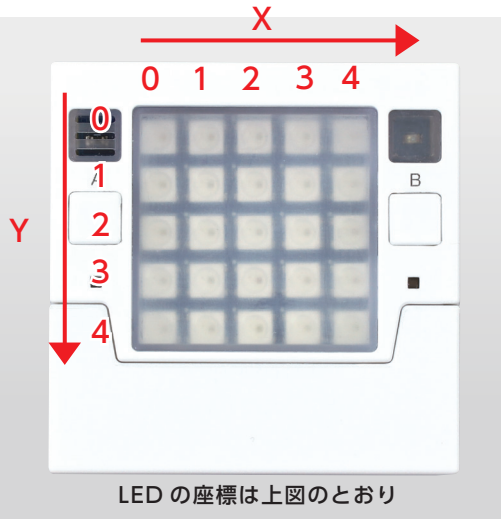
右図のようなセンサーボードが表示されると、
通信が正常に行われています。

Bluetooth による通信時は
通信ランプが青に点灯します。



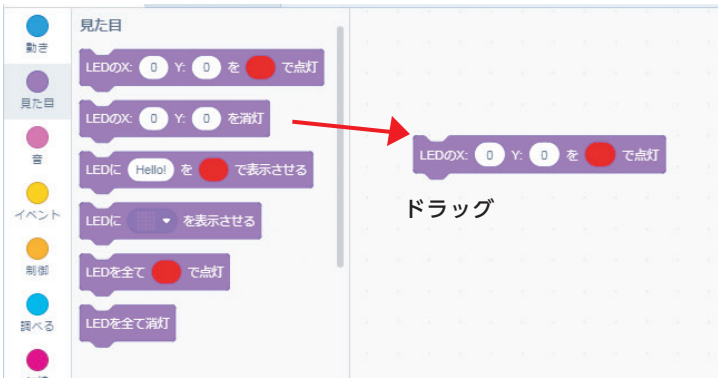
③LED を点灯させるプログラム

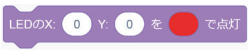
LED を光らせる命令を出すには次のブロックをつかいます。



LED の座標は上図のとおり

ブロックをドラッグして光らせたい LED の座標と色を指定します。





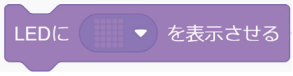
をクリックして LED が点灯することを確認してください。

やってみよう!

その他のブロックについてもどのように LED が点灯するか同様に確認してください。

……指定した座標の LED を消灯します。

……指定した文字をスクロール表示します。(半角英数字)

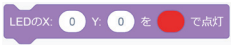


……全ての LED の点灯色を個別に指定します。

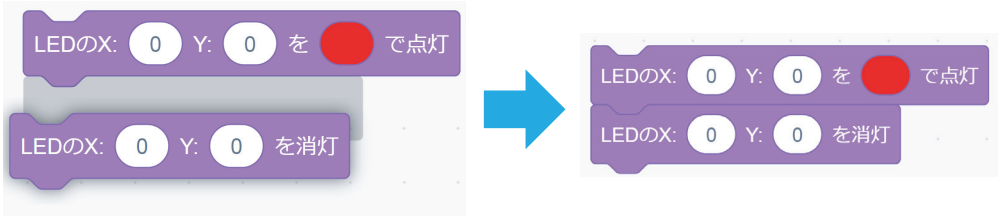
……全ての LED の点灯色を同時に指定します。

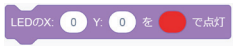
……全ての LED を消灯します。

④LED を 1 秒間だけ点灯させるプログラム

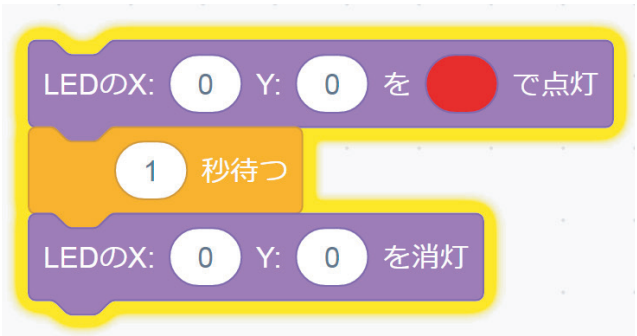
 のブロックと  を順番につなげましょう。
このようにブロックをつなげると、上から順番に命令が送られて、プログラムが実行されます。

ドラッグしたブロックを影ができたところではなすと、ブロックどうしをつなぐことができます。



 ブロックを  と  の間につなぎましょう。

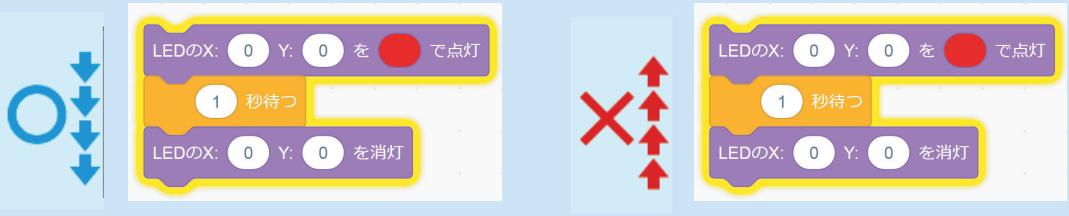
並べたブロックをクリックするとプログラムが転送されて実行されます。
LED が 1 秒間光って消えることを確認しましょう。



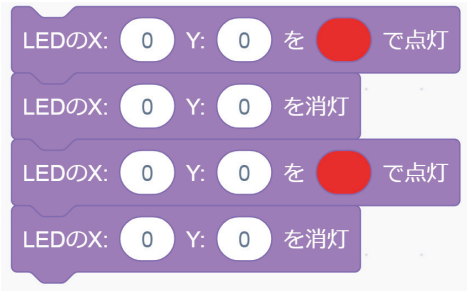
 の数値を変えることで、LED の点灯時間を設定することができます。
また、時間は少数点も設定することができるので、自由に変えて動かしてみましょう。

※コンピュータ内の通信処理速度により設定した時間より少し長い時間点灯する場合があります。

プログラムは必ず上から順番に実行されます。



「1秒待つ」のブロックを入れないとどうなるのか?



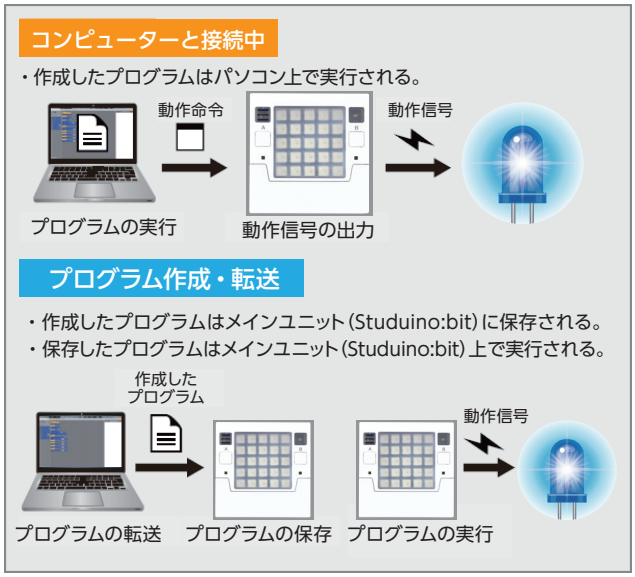
左のプログラムで動作をさせた場合、LED はほとんど点灯しません。
これはコンピューターがプログラムを非常に高速で処理していることが原因です。
つまり、「LED 点灯」の命令の直後に、「LED 消灯」の命令が行われてしまうため、LED を点灯している時間がありません。
※一瞬点灯する場合がありますが、これはコンピューターとの通信の際のわずかな遅延によるものです。

⑤接続を解除する

プログラムの動作が確認できたら、接続を解除しましょう。
画面上方の編集より、接続解除を選びます。

⑥プログラムを転送する

ここまではコンピューターとメインユニット(Studuino:bit)が通信し、コンピューターからの命令でプログラムを動かしていました。
作成したプログラムをメインユニット(Studuino:bit)に転送することで、メインユニット(Studuino:bit)のなかにプログラムが保存され、コンピューターから切り離してもプログラムを実行することができるようになります。



プログラムを作るたびに転送を行う必要がありますが、USB ケーブルや Bluetooth の通信を切断してもプログラムを実行できたり、プログラムの実行や読み込みが早くなるというメリットもあります。

プログラムの転送方法

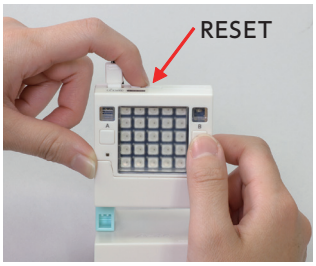
①作成したプログラムの一番上に 

をつなぎます。

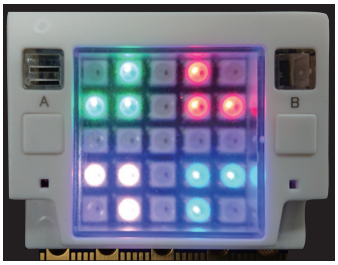
②編集から「転送」を選びます。

Android/iPad/Chromebook の場合　Bluetooth による通信

①表示されるメッセージに従い、メインユニット(Studuino:bit) の B ボタンを押しながらリセットボタンを押してください。



②メインユニット(Studuino:bit) の LED に表示された点灯パターンと同じデバイスを選択してください。



③プログラムを転送する先を指定してください。
※0 ～ 9 まで選択できます。

④転送が完了すると、自動的にプログラムが実行されます。
再度動かしたい場合はリセットボタンを押してください。

複数のプログラムを転送した場合の実行方法

複数のプログラムを 0 ～ 9 の番号を指定して転送することができます。

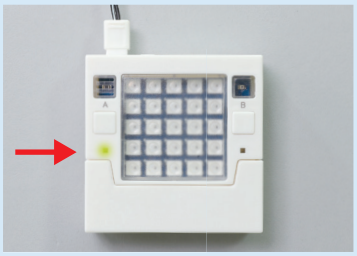
通常は最後に転送したプログラムが実行されますが、以下の方法で番号を指定して転送済みのプログラムを実行することができます。

①電源を接続する（通常起動）

(1) 電池ボックスもしくは USB ケーブルをメインユニットに接続します。

(2) 電源ランプ（緑）が点灯し、電源が自動的に ON になり起動します。

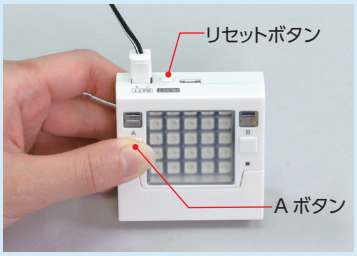
通常起動時は前に実行したプログラムが自動的に実行されます。



②プログラム選択モードへの移行

(1) A ボタンを押したまま、リセットボタンを押してください。
リセットボタンを押すと再起動のため一時的に電源ランプが消灯します。

(2) リセットボタンを離して 3 秒後に電源ランプが再点灯したら、
A ボタンを離します。



(3) 緑の LED で 0 が表示されるとプログラム選択モードへ移行しています。

③プログラムの選択・起動

(1) プログラム選択モード時に A ボタンを押すと、表示が 0・1・2・・・と
切り替わります。9 まで切り替わると 0 に戻ります。

(2) 選択した番号で B ボタンを押すと、各番号に割り振られたプログラムが
実行されます。

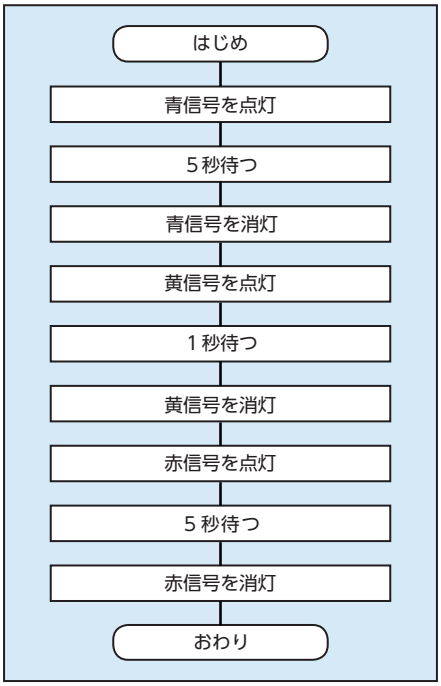
次回より通常起動時に③で選択したプログラムが自動で実行されます。

※工場出荷段階で動作確認用に各番号にはサンプルプログラムが書き込まれています。
※サンプルプログラムはソフトウェアを使用して新たなプログラムを書き込むと上書きされます。

練習課題　自動車用信号機のプログラムの作成

次の動作の手順をフローチャートにまとめたあと、プログラムをつくりましょう。

プログラム例



ブロックの代わりに

を使用しても同様のプログラムを作成することができます。

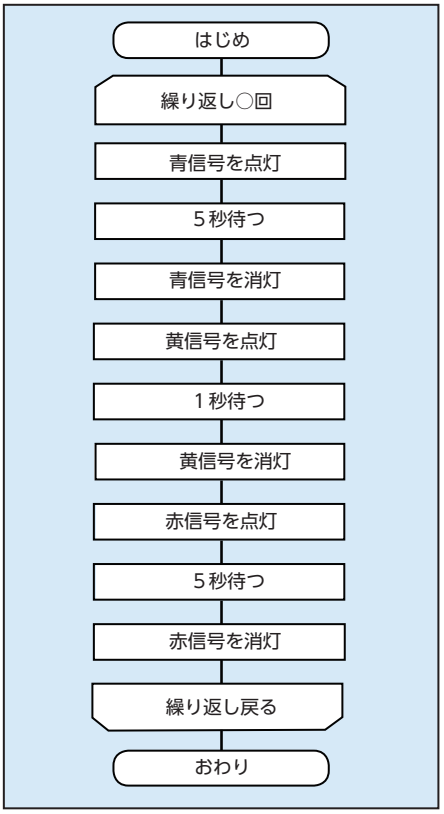
2．繰り返しのプログラム

ある命令を繰り返し行う処理のことを「繰り返し処理」といいます。前のページで作成した信号機のプログラムは赤信号の点灯が完了すると消灯し、その後はプログラムは動きません。実際の信号機のように繰り返し動作させるには、この「繰り返し処理」が必要になります。

①フローチャート

同じ動作を複数回繰り返す場合、以下の記号を使います。
信号機の動作を表すフローチャートをまとめましょう。

繰り返し○回	繰り返し処理の開始
繰り返し戻る	繰り返し処理の終了

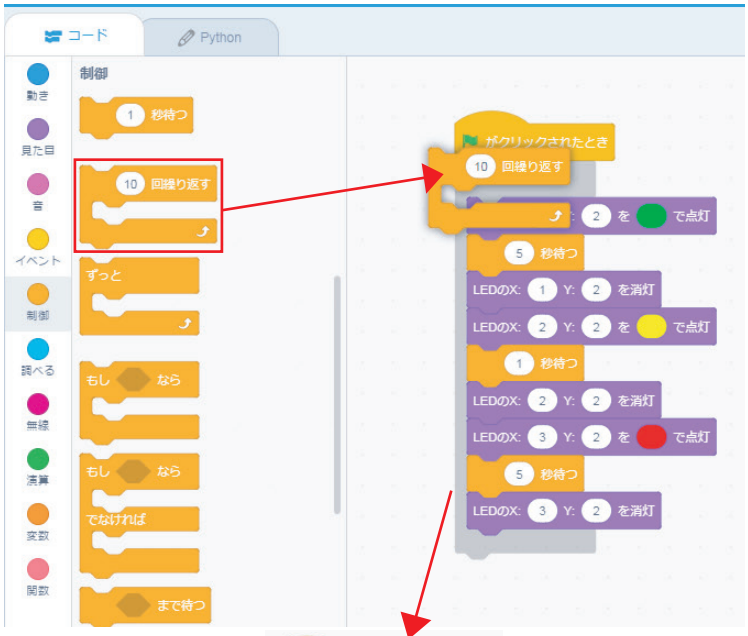


「終了」がなくなり、繰り返し続けることは下のフローチャートのように矢印で表すこともできます。

②プログラムの改造（繰り返し処理の追加）

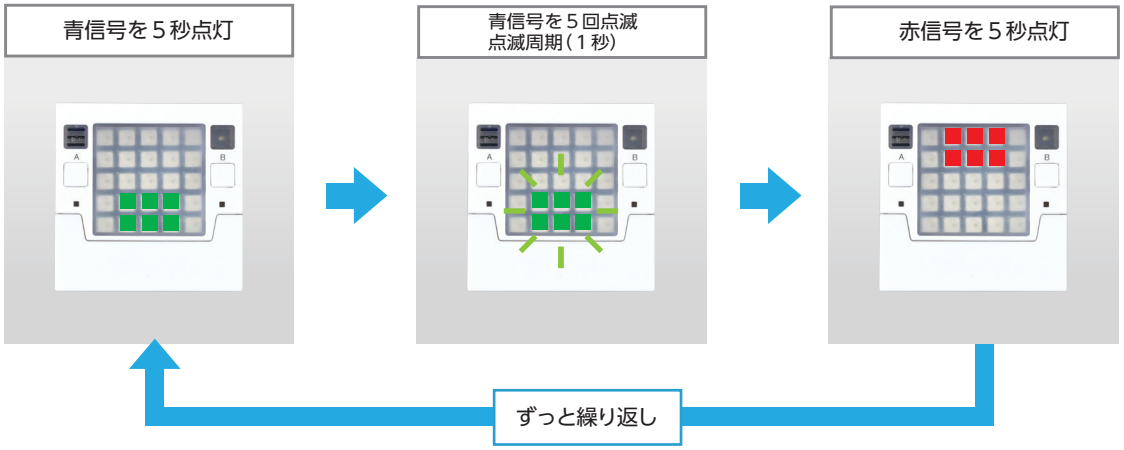
同じ動きを指定回数繰り返させたい場合は、「○回繰り返す」ブロックを、ずっと繰り返させたい場合は「ずっと」ブロックを利用します。

下図のように自動車用信号機のプログラム全体を囲むように  または  を追加してください。

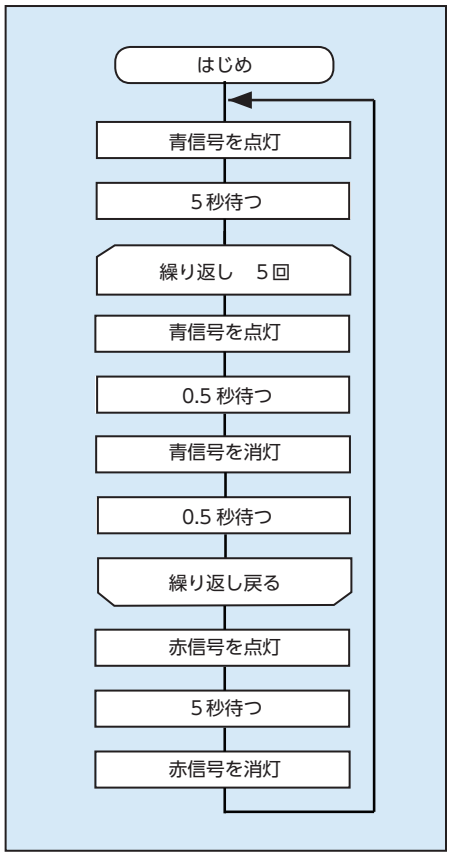


練習課題　　歩行者用信号機のプログラムの作成

次の動作の手順をフローチャートにまとめたあと、プログラムをつくりましょう。



プログラム例



ブロックの代わりに


を使用しても同様のプログラムを作成することができます。

3．条件分岐のプログラム

条件によって動作を分ける処理のことを「条件分岐処理」といいます。押しボタン式信号機のプログラム作成を通して条件分岐のプログラムを学びましょう。

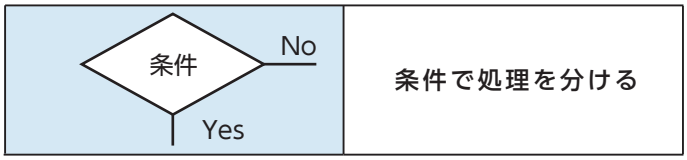
①ボタンの数値確認

センサーの情報は数値で表されます。
メインユニット（Studuino:bit）をコンピュータと接続中に表示されるセンサーボードでプッシュボタンが押されているときと押されていないときの数値の変化を確認しましょう。

ボタンが押されているときの数値	ボタンが押されていないときの数値
0	1

②フローチャート

条件分岐を表す場合、以下の記号を使います。



条件は  と  を組み合わせてつくることができます。

作成した条件は右図の各種制御ブロックの空欄に入れて使うことができます。

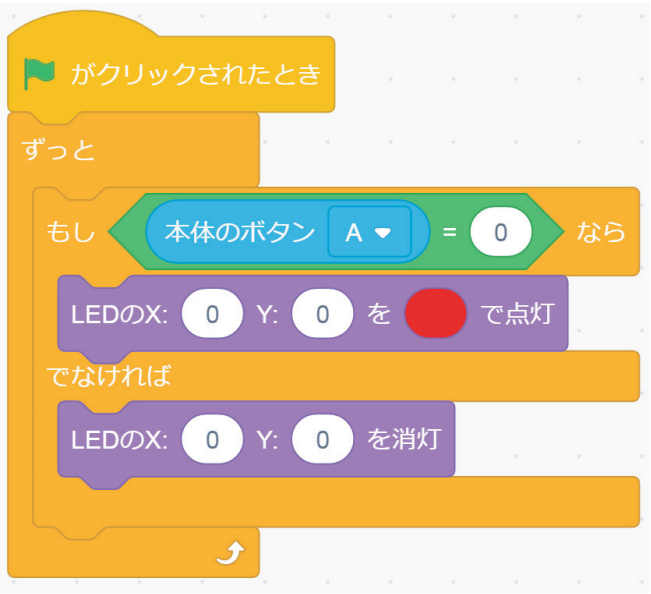
…… 条件が成り立つときだけ、囲まれたプログラムが処理されます。

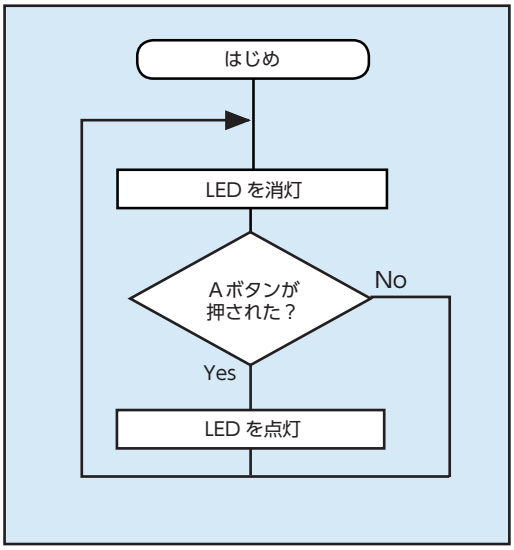
…… 条件が成り立つときだけ、上段で囲まれたプログラムが処理され、
条件が成り立たないときは、下段で囲まれたプログラムが処理されます。

…… 条件が成り立つまで、次に続くプログラムを処理せずに待ちます。

…… 条件が成り立つまで、囲まれたプログラムを繰り返し処理します。

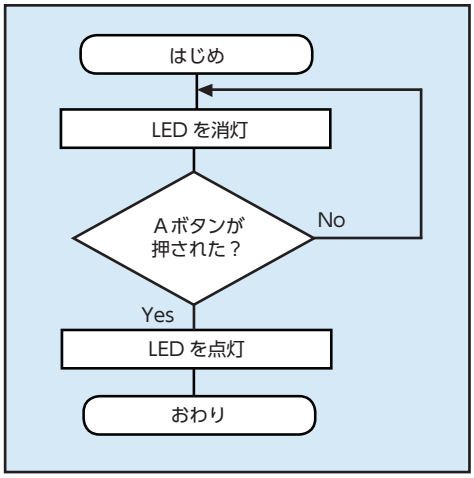
③Aボタンを押している間 LED が点灯するプログラム



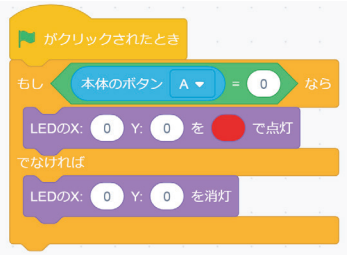


「ずっと」のブロックを入れないとどうなるのか？

「ずっと」を入れない場合のフローチャートとプログラムは以下のようになります。



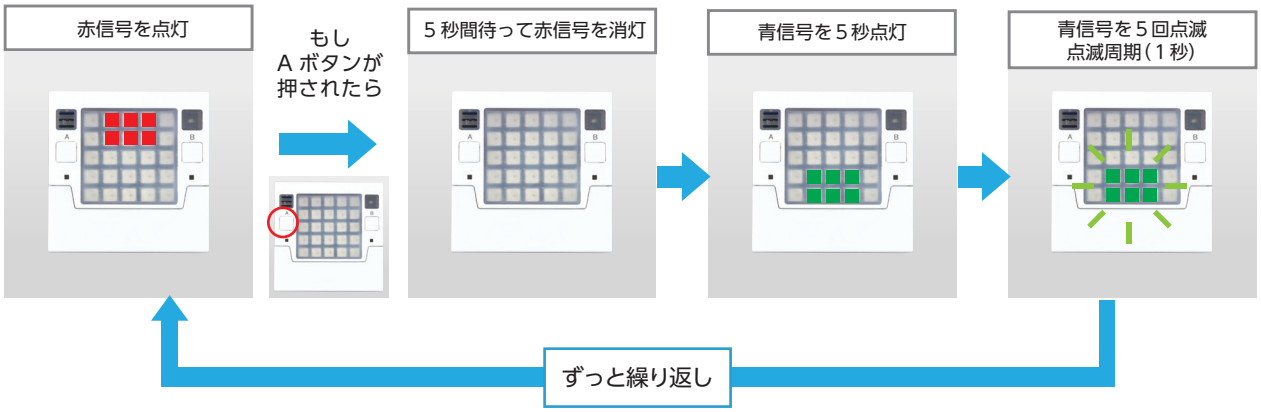
このプログラムを動作させると、A ボタンを押しても全く反応しなくなります。
これは、プログラムが非常に高速で処理されることで、プログラムを動作させた瞬間にプログラムが終了してしまうからです。
「ずっと」のブロックを入れることで、A ボタンの値を常に確認するようになるため想定通りの動作をするようになります。



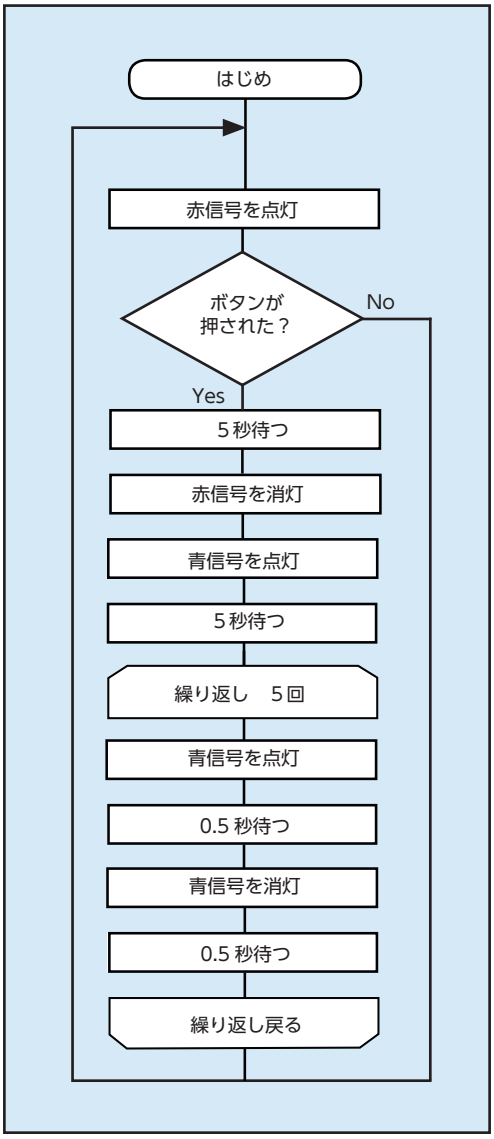
動作させた瞬間にプログラムが終了。
プログラム終了後に A ボタンを押しても反応しない。

練習課題

＜押しボタン式信号機のプログラム＞
次の動作の手順をフローチャートにまとめたあと、プログラムをつくりましょう。



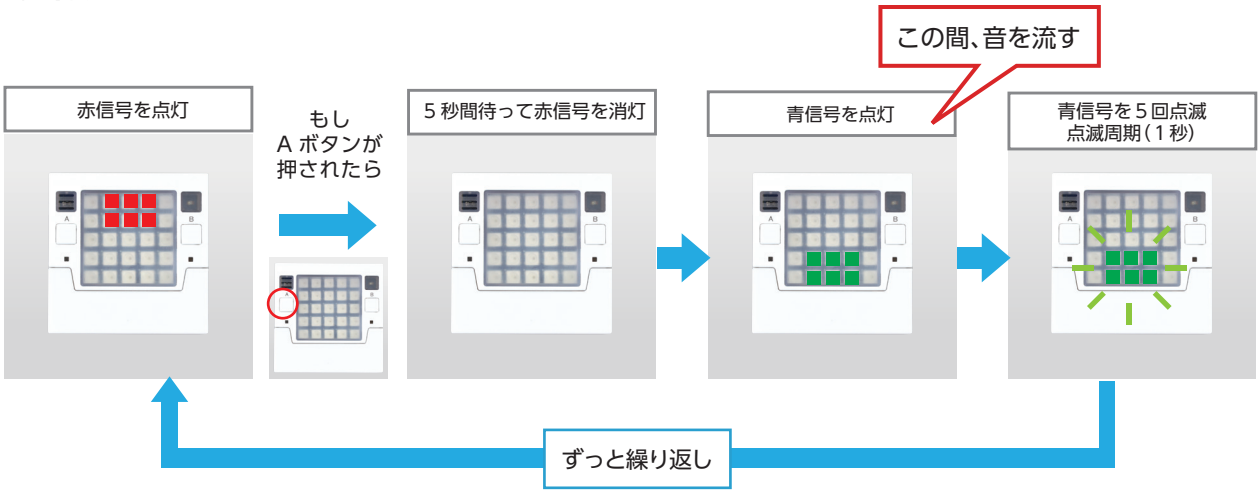
プログラム例



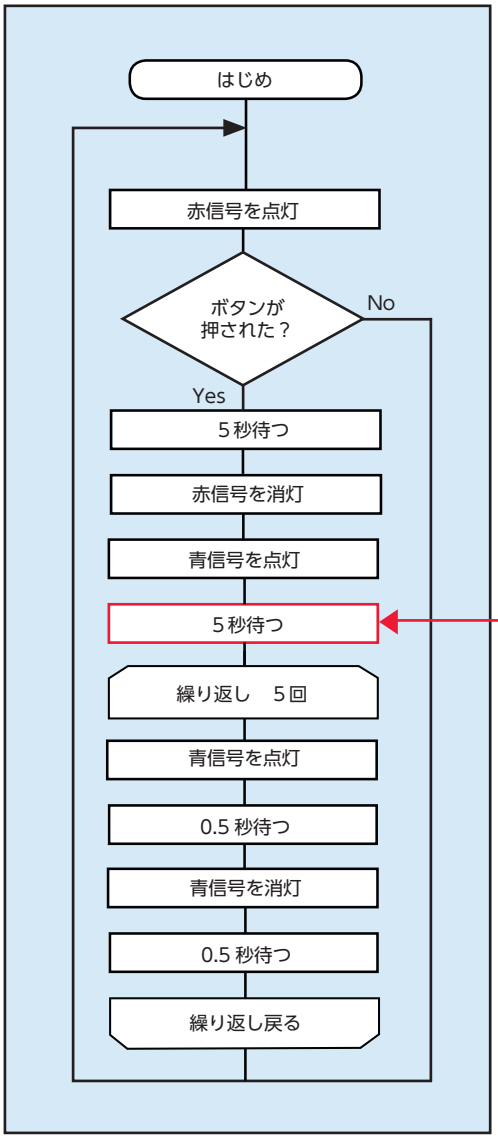
発展学習①　ブザーをつかった制御

＜音響装置付信号機のプログラム＞

目の不自由な人のために、信号がかわったことを音で知らせてくれる信号機があります。
これを音響装置付き信号機といいます。
押しボタン式信号機のプログラムで青信号が点灯している間、ブザーの音を鳴らすプログラムを追加して音響装置付き信号機にしてみましょう。

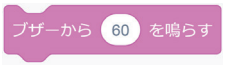


プログラム例



この処理の間、ブザーを鳴らすプログラムを追加します。

①ブザーの音を鳴らすプログラム

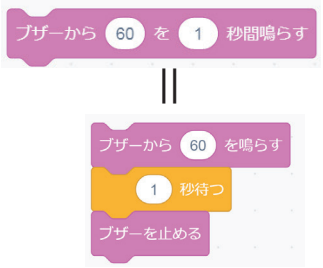


……指定した高さの音を鳴らします。

……指定した高さの音を指定時間鳴らす。

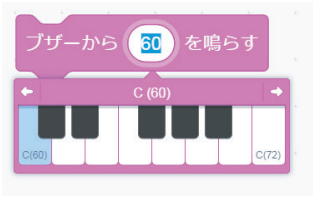
……ブザーからの音を止める。

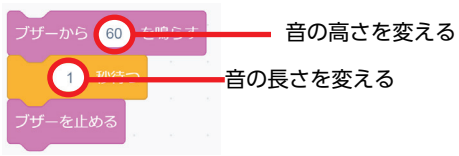
プログラム例
1 秒間音を鳴らしてとめるプログラム



やってみよう！

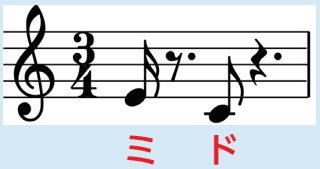
音の高さや長さを変えてみよう。





②カッコウの音をつくろう

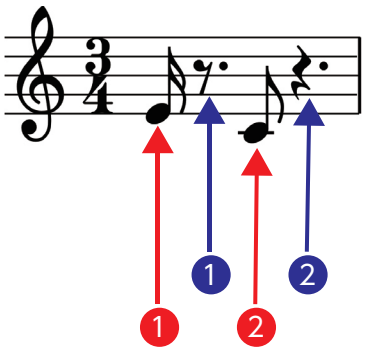
「カッコウ」の音は音ぷにすると
「ミ(64)・ド(60)」です。



「カッコウ」の音の長さの決め方

音楽には音の長さや音同士の間隔などが決められた「楽譜」があります。カッコウの曲にも楽譜があり、今回のプログラムの 2 つの音と 2 回の休みの長さも実際の楽譜に合わせて設定しています。

「カッコウ」の音の楽譜



「カッコウ」の音のプログラム

カッコウの音を 5 回繰り返す場合は、右図のように「5 回繰り返す」ブロックで囲みます。

③青信号が点灯している間音が鳴るようにしよう

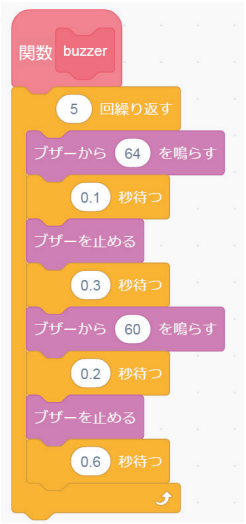
関数ブロックを作成します。

関数名を入力します。
（ここでは buzzer とします）

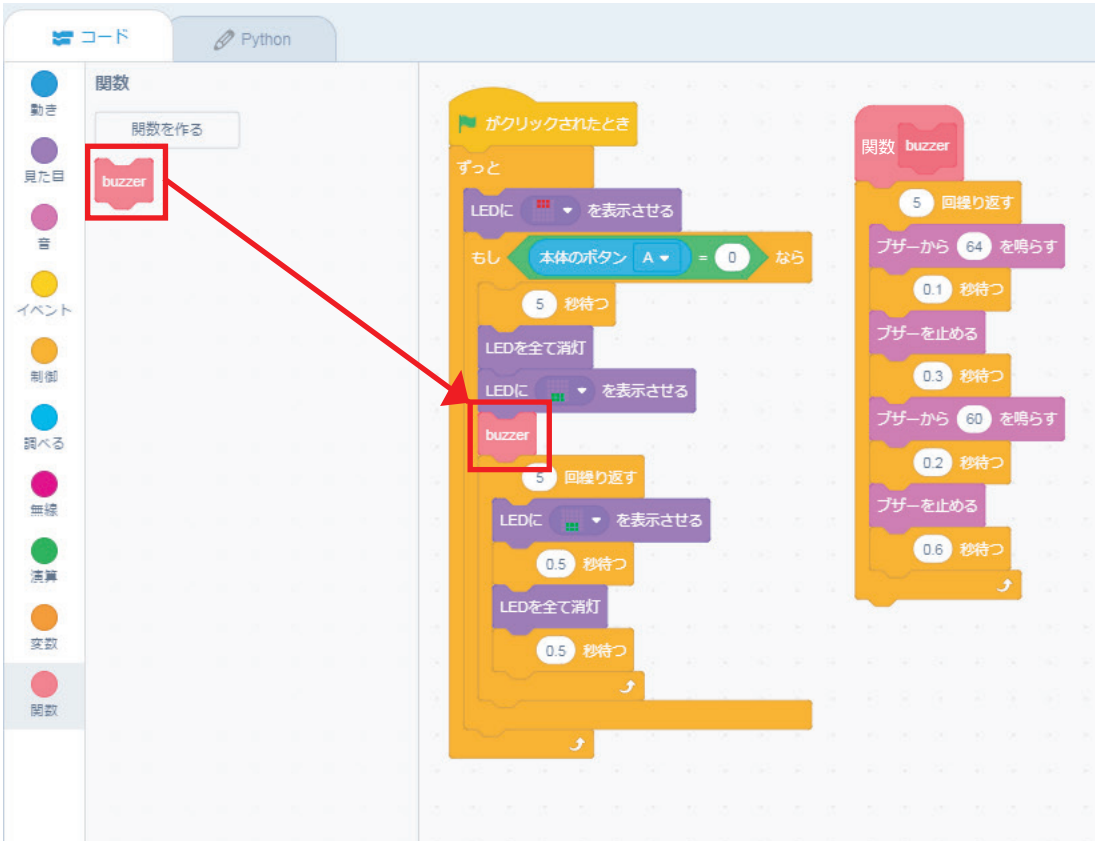
関数名を入力
（半角英数字）

クリック

スクリプトエリアに表示される
関数の定義ブロック 関数 buzzer に②で
作成したプログラムをつなげます。



練習課題で作成した押しボタン式信号機の青信号が点灯した後の 5 秒待つ と、関数の
呼び出しブロック buzzer を入れ替えます。

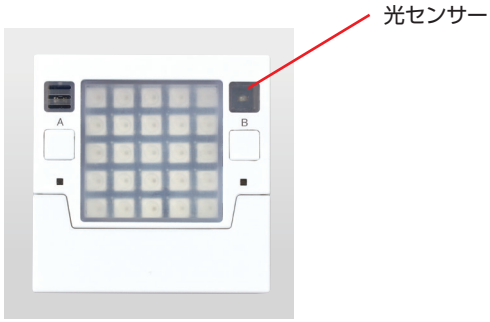


発展学習②　光センサーをつかった制御

＜センサーライトのプログラム＞

周囲が暗くなったら自動で照明が点灯する、センサーライトのプログラムを考えましょう。

①「編集」より接続を選択しコンピュータと接続してください。

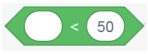


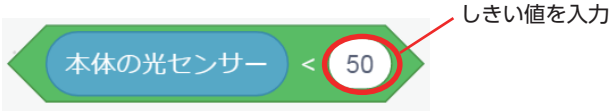
②センサーボードで光センサーの値を確認します。
光センサーを手で覆って暗くするとセンサーの値がどのように変化するか確認しましょう。

③しきい値の決定
どの程度暗くなったら LED を点灯させるかは②でしらべた光センサーの値で決めます。

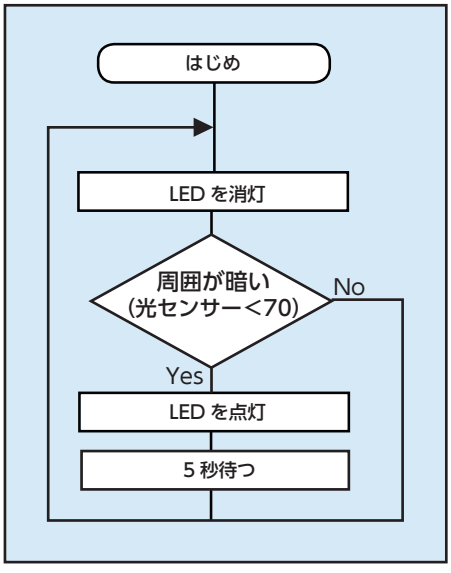
このような点灯させる条件となる値のことを「しきい値」といいます。

④光センサーの値が一定以上小さくなったら
5 秒間 LED を点灯させて消灯させるプログラムを考えましょう。

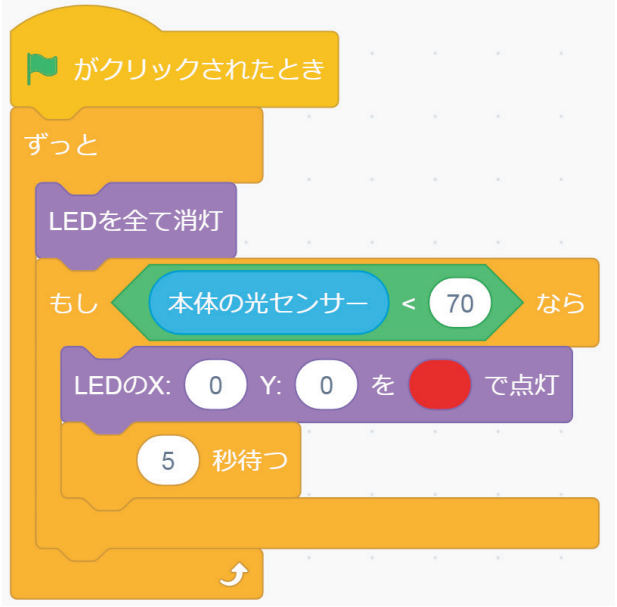
条件は  と  を組み合わせてつくることができます。



フローチャート



プログラム例



基本学習 テーマ2

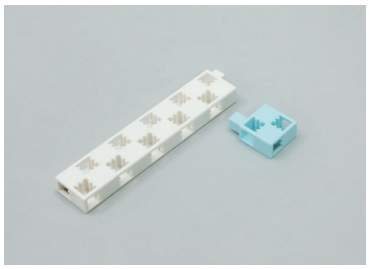
ロボットカーの制御

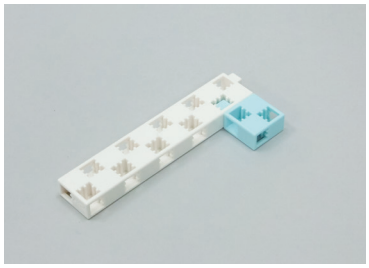
＜学習内容＞

- ●DCモーターの制御
- ●車の制御
- ●赤外線フォトリフレクタ
- ●衝突回避カー
- ●落下回避カー
- ●ライントレースカー

①組み立て

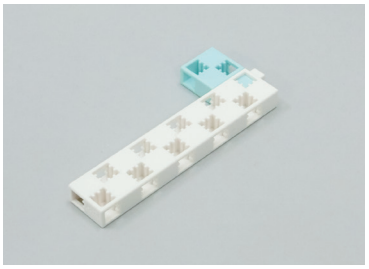
①





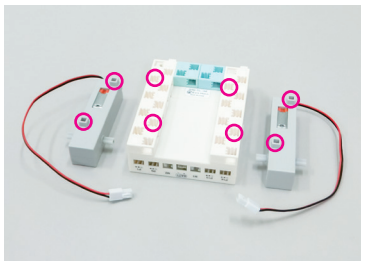
②

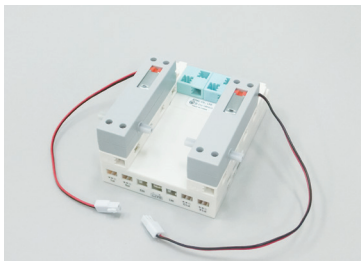
③



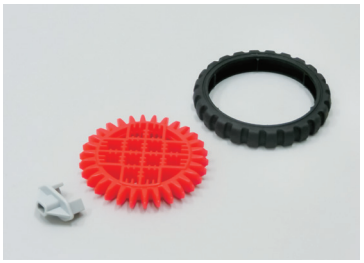
④

⑤

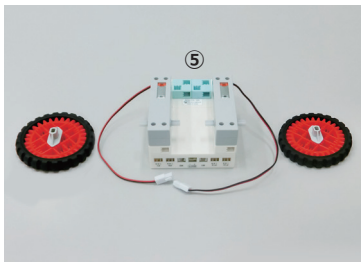


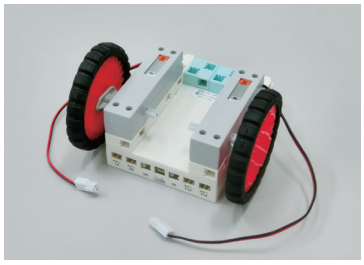


⑥ ×2

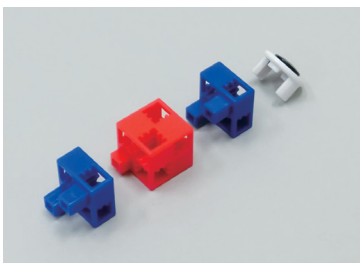


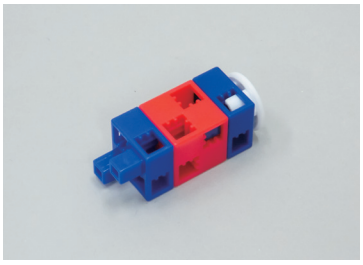
⑦



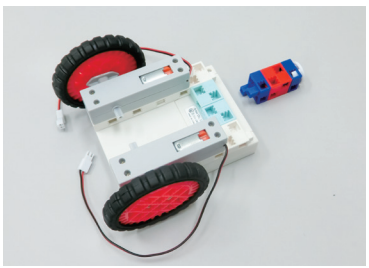


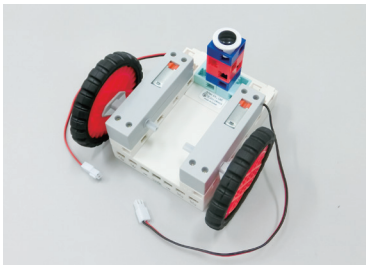
⑧



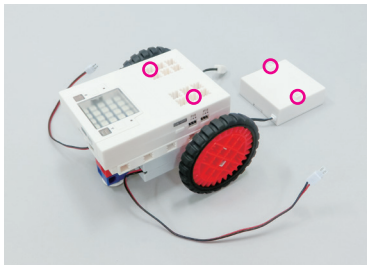


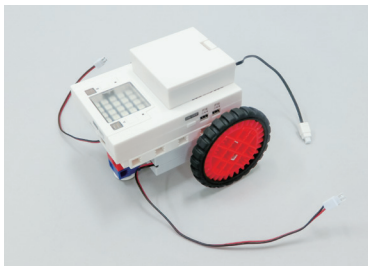
⑨

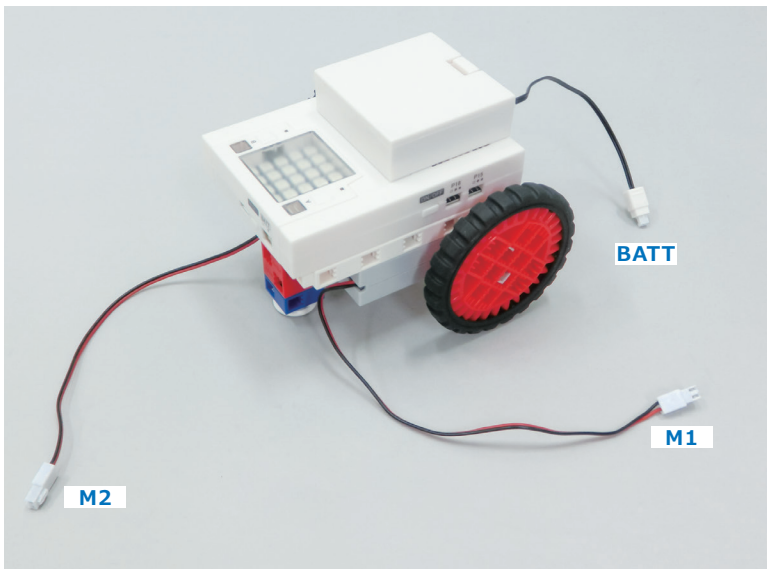


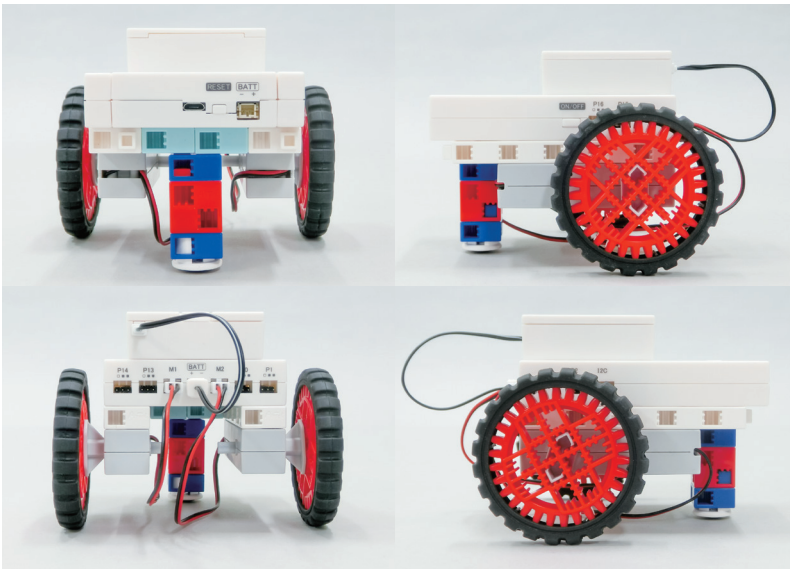


⑩



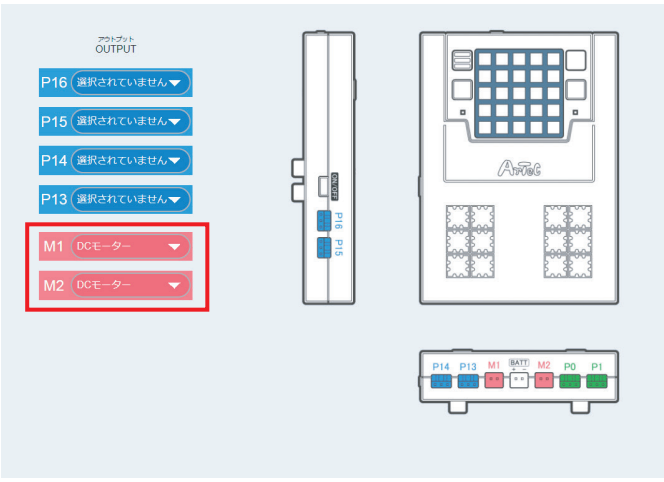






②入出力設定

「編集」より「入出力設定」を選択し、
M1 と M2 に DC モーターを選択します。

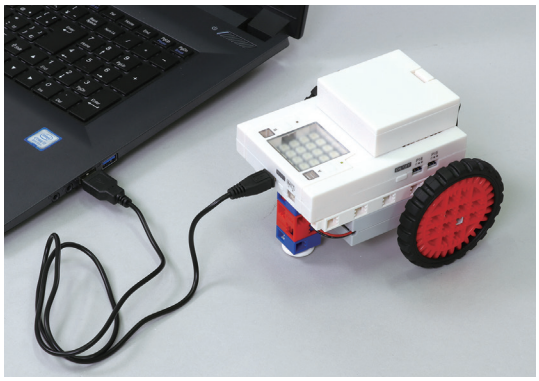


③コンピューターとの通信

Windows/Mac の場合

USBケーブルによる通信

①コンピュータとメインユニット(Studuino:bit)を USB ケーブルで接続します。

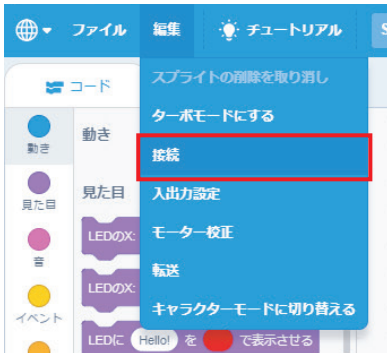


②「編集」より接続を選択します。

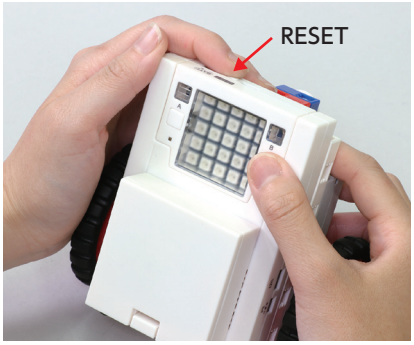
Android/iPad/Chromebook の場合

Bluetoothによる通信

①「編集」より接続を選択します。



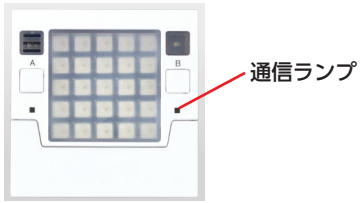
②表示されるメッセージに従い、メインユニット(Studuino:bit)の B ボタンを押しながらリセットボタンを押してください。



③メインユニット(Studuino:bit)の LED に表示された点灯パターンと同じデバイスを選択してください。

右図のようなセンサーボードが表示されると、通信が正常に行われています。

Bluetooth による通信時は通信ランプが青に点灯します。

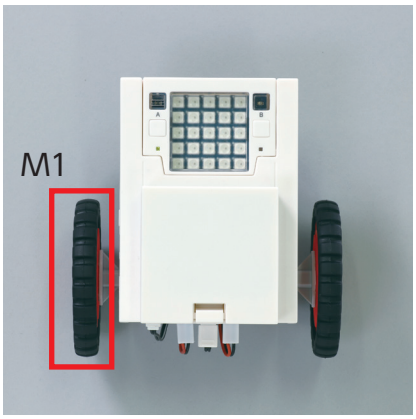


1. DCモーターの動作確認

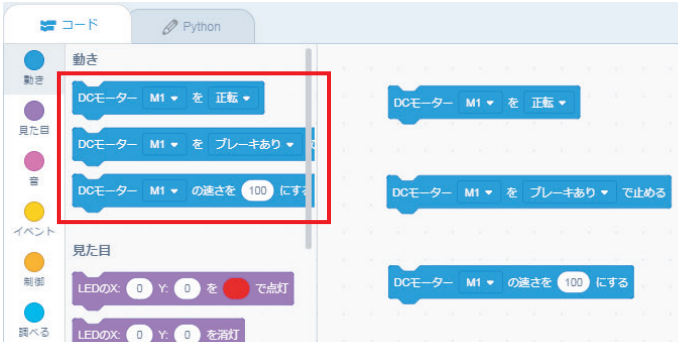
DC モーターを動かして、プログラムと動作の関係を調べましょう。

M1 につないだ左の DC モーターのタイヤに注目しましょう。

※DC モーターを動かす場合は必ず電池ボックスをロボット拡張ユニット側に接続するようにしてください。



DCモーターは回転の速さと向きを設定するプログラムで制御されています。どちらか一方でもプログラムが抜けているとDCモーターは動きません。DCモーターのブロックをスクリプトエリアに並べましょう。



各ブロックをクリックすると、そのブロックの処理が行われます。

DCモーターの速度設定

DCモーターの回転する速さを100に設定するために、

をクリックしましょう。

回転する速さはブロックの数値に対応しています。

※数値を変更した場合、再度 

をクリックしないと、変更した数値が反映されません。

DCモーターの回転

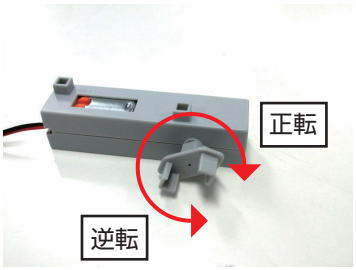
をクリックした後、


をクリックすると DC モーターが回転します。

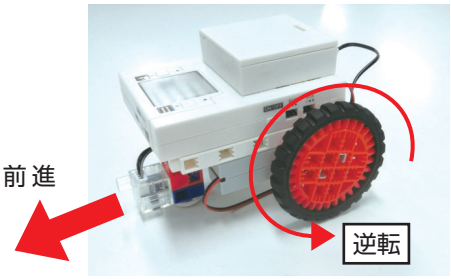
※ 速さを設定した後でないと、


をクリックしても DC モーターは動きません。

DC モーターの回転の向きは以下のようになります。

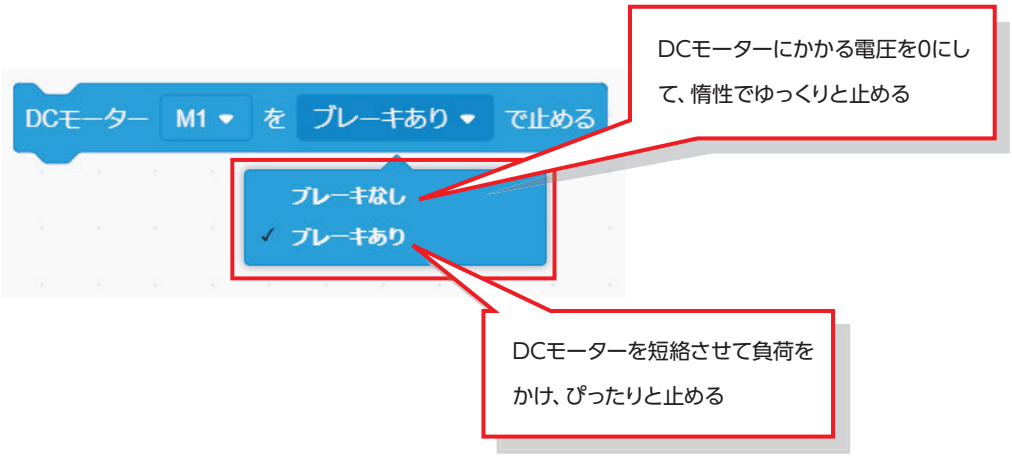


ロボットカーのタイヤは左右両方の DC モーターを逆転させる方向に回転させたときに下図のように前進します。



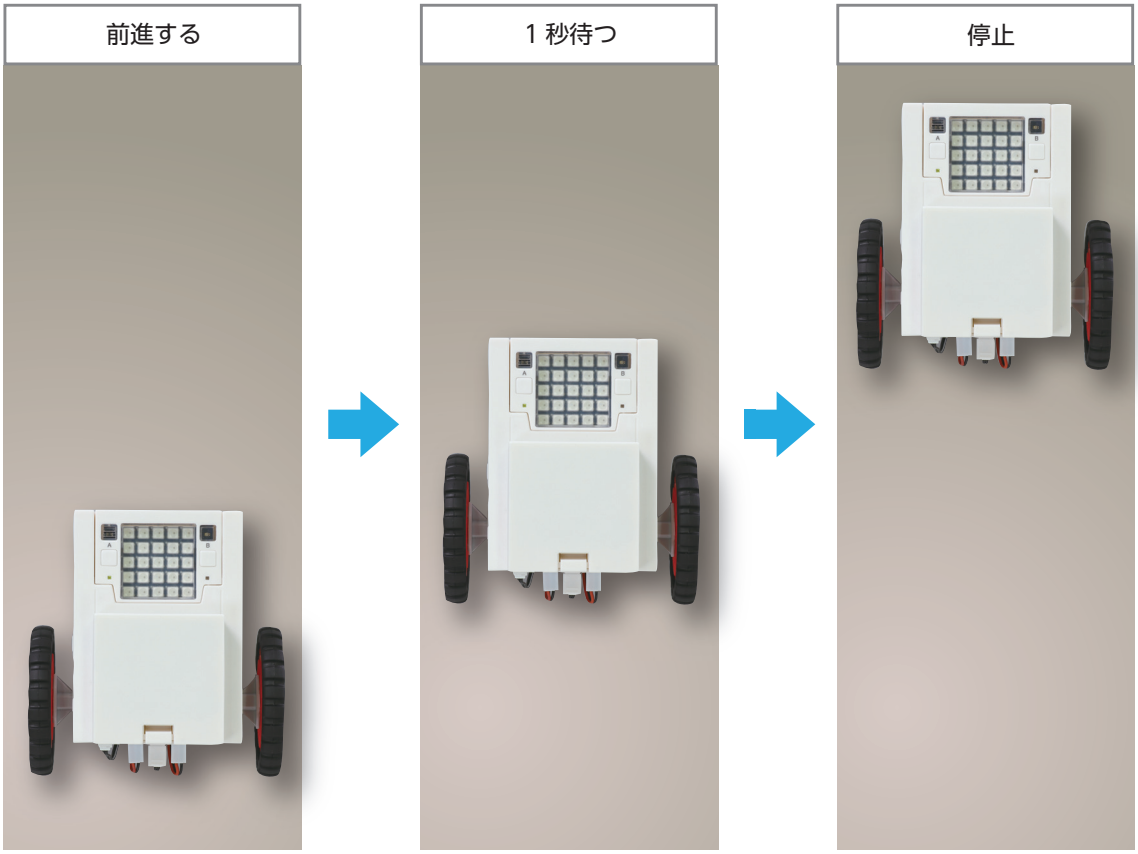
DCモーターの停止

をクリックして、DC モーターの回転を停止させましょう。DC モーターの止め方は「ブレーキなし」と「ブレーキあり」を選択することができます。



プログラミング

車を１秒だけ前進させるプログラムをつくりましょう。



両方のタイヤの DC モーターが同じ速さで回転すると、車は前に進みます。したがって、M1 と M2 の速さをともに 100 にして、逆転させましょう。

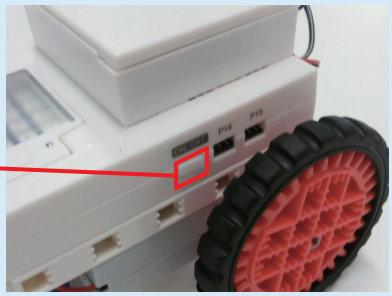
つぎに、制御カテゴリーにある「１秒待つ」のブロックを先ほどのブロックの下につなぎましょう。

プログラムが実行される流れ

最後に、DCモーターを止めるブロックをつなぎます。

プログラムを転送して動きを確認しましょう。

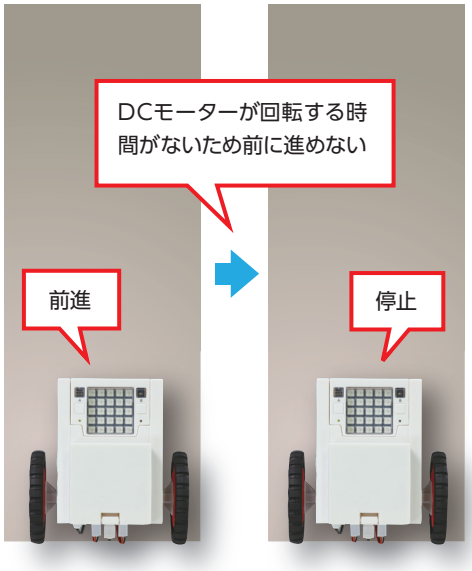
プログラム転送後、USB ケーブルを外して動かす場合、ロボット拡張ユニット側面の電源ボタンを長押し(2 秒)して電源を ON にしてください。



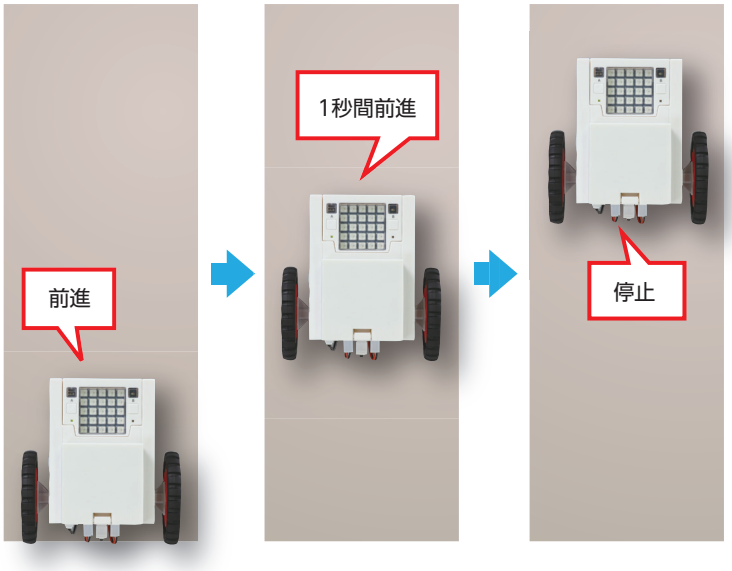
「1秒待つ」のブロックを入れないとどうなるのか?

左のプログラムで動作をさせた場合、車はほとんど動きません。これはコンピューターがプログラムを非常に高速で処理していることが原因です。つまり、「DC モーターを正転」の命令の直後に、「DC モーターを止める」の命令が行われてしまうため、車が前に進む時間がありません。

「1秒待つ」を入れない場合

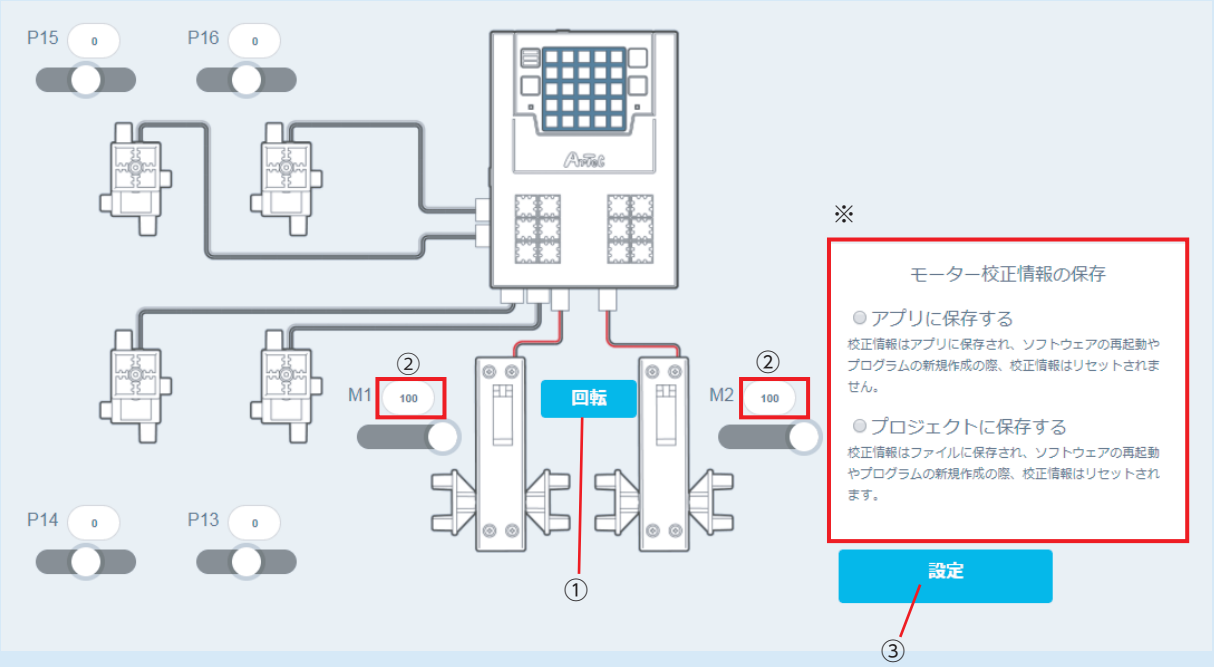


「1秒待つ」を入れる場合



モーターの校正

左右の DC モーターの個体差により、左右同じ速さで設定してもまっすぐに走らない場合があります。
「編集」の「モーター校正」からDC モーターの回転の速さを校正してください。



①回転ボタンをクリックすると接続された DC モーターが回転します。

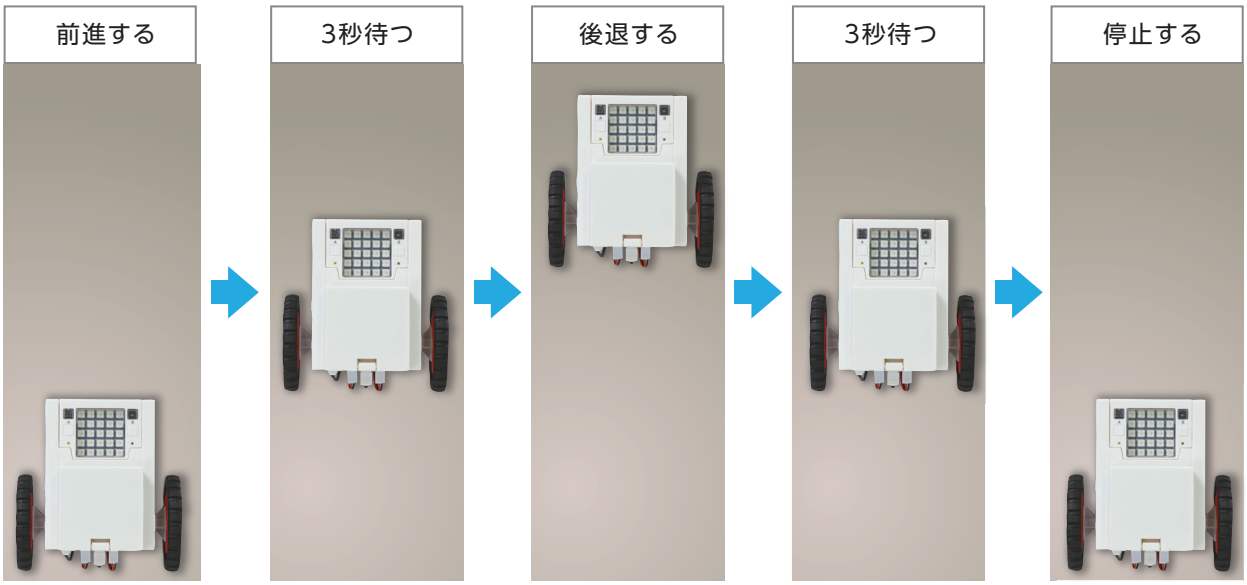
②M１・M２の値を小さくすると回転速度が制限されます。
M１・M２の値を調整して同じ速さで回転するように校正します。

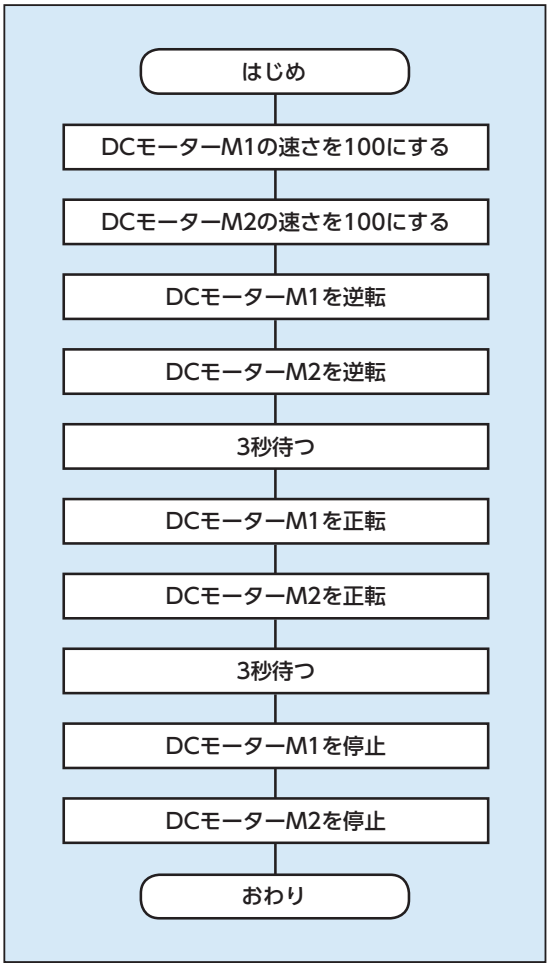
③設定をクリックして校正を完了します。

※モーター校正情報の保存先を選択してください。
1 台のコンピューターにつき 1 台のロボットを使う場合は「アプリに保存する」を選択してください。
1 台のコンピューターにつき複数のロボットを使う場合は「プロジェクトに保存する」を選択してください。

練習課題

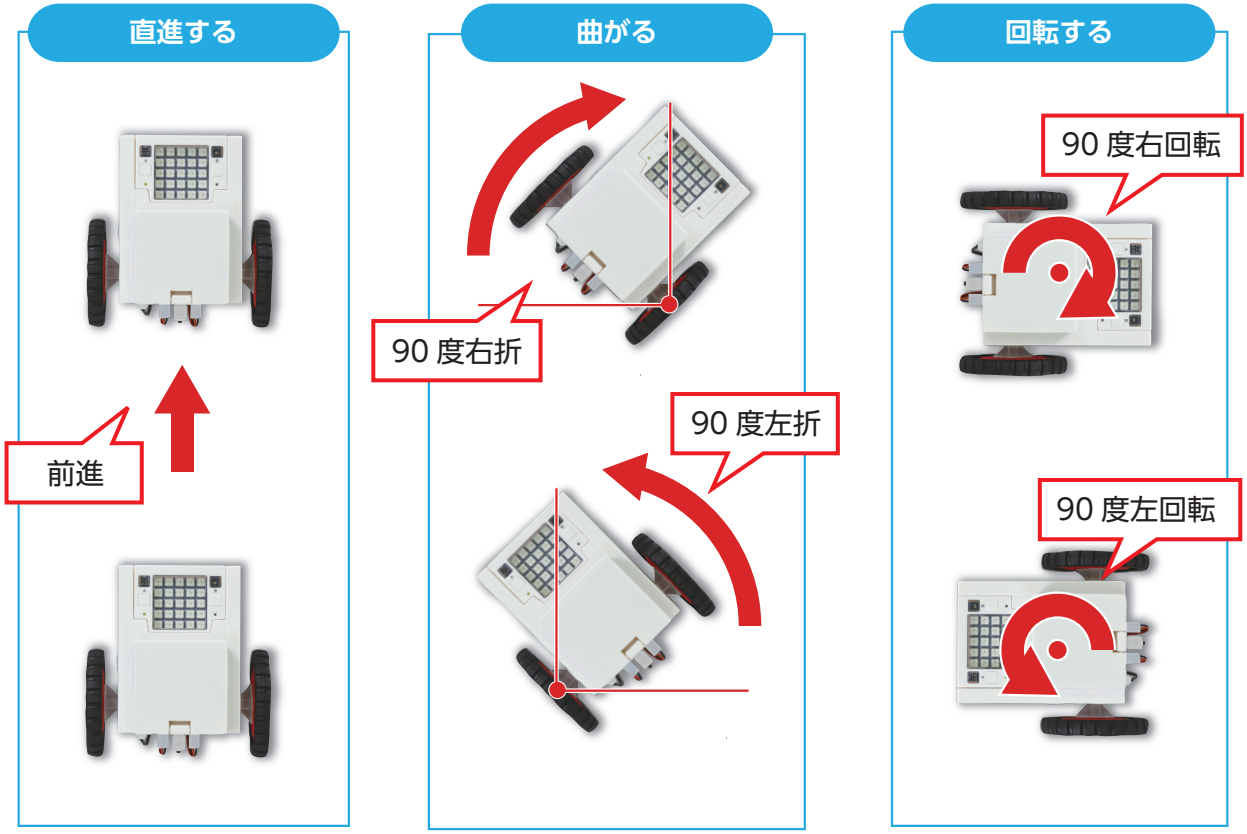
次の動作の手順をフローチャートにまとめたあと、プログラムをつくりましょう。





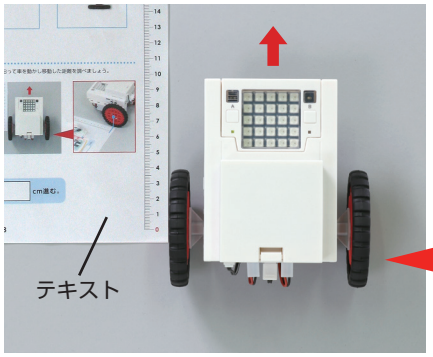
2．車の制御

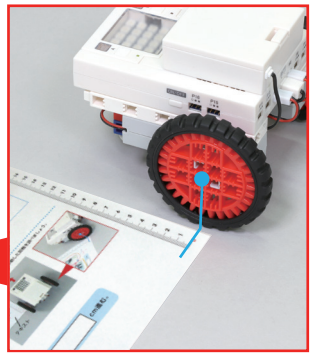
1 秒間前進させたときの距離を調べましょう。また、車をぴったり90度曲げたり、回転させるときに DC モーターを動かす時間を調べましょう。
※進む距離や曲がったり回転する時間は DC モーターの個体差や電池の残量によって異なる場合があります。



①前進

1 秒間前進させるプログラムをつくり、ページ右側の定規に沿って車を動かし移動した距離を調べましょう。





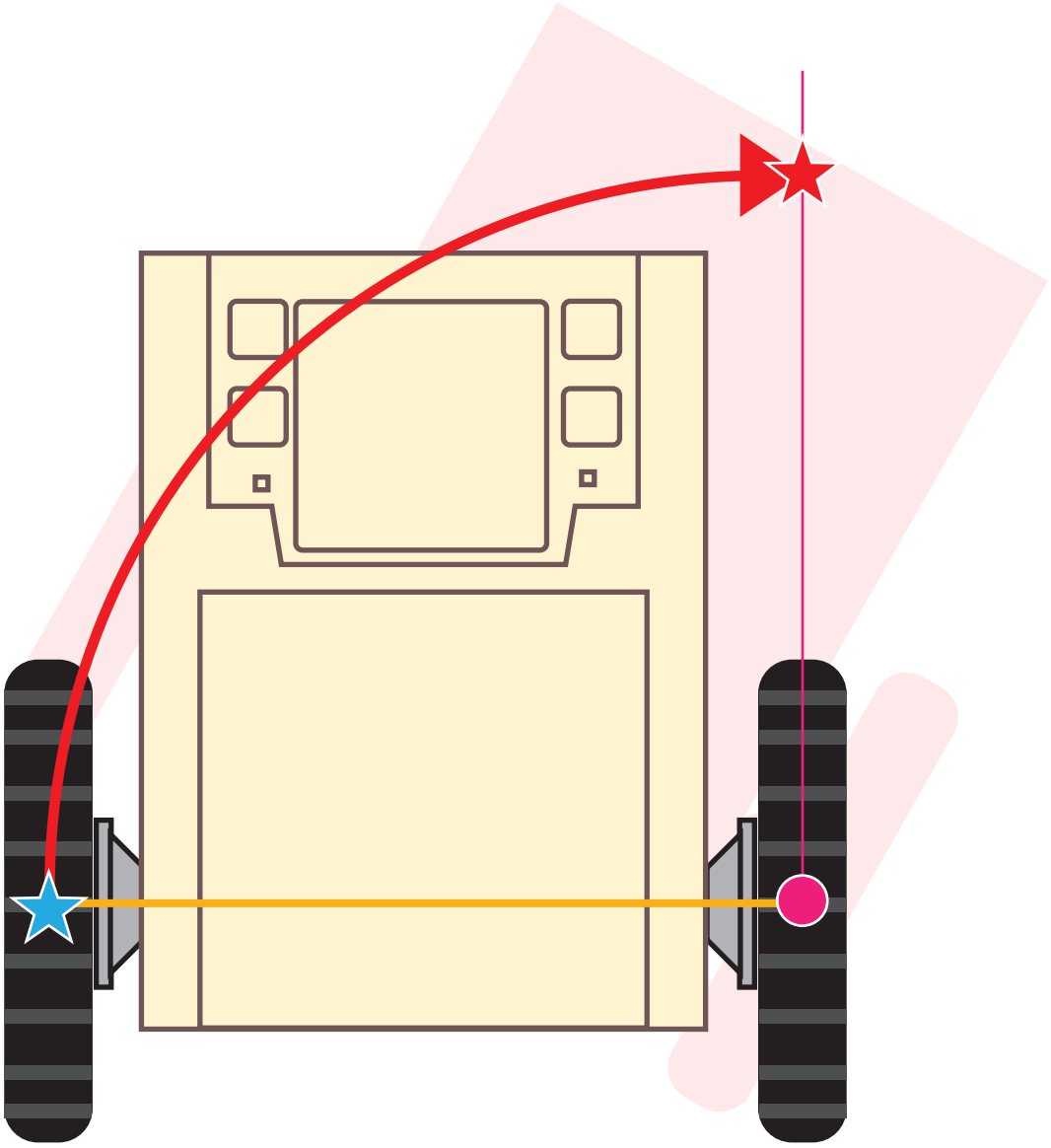
1 秒間で　　cm進む。

②右折

左(M1)のDCモーターだけを逆転させることで右折ができます。イラストに合わせて、ぴったり90度右折するために必要な時間を調べましょう。

左のプログラムをつくり、「待つ」ブロックの時間を調整しましょう。

［　　］秒で90度右折する。

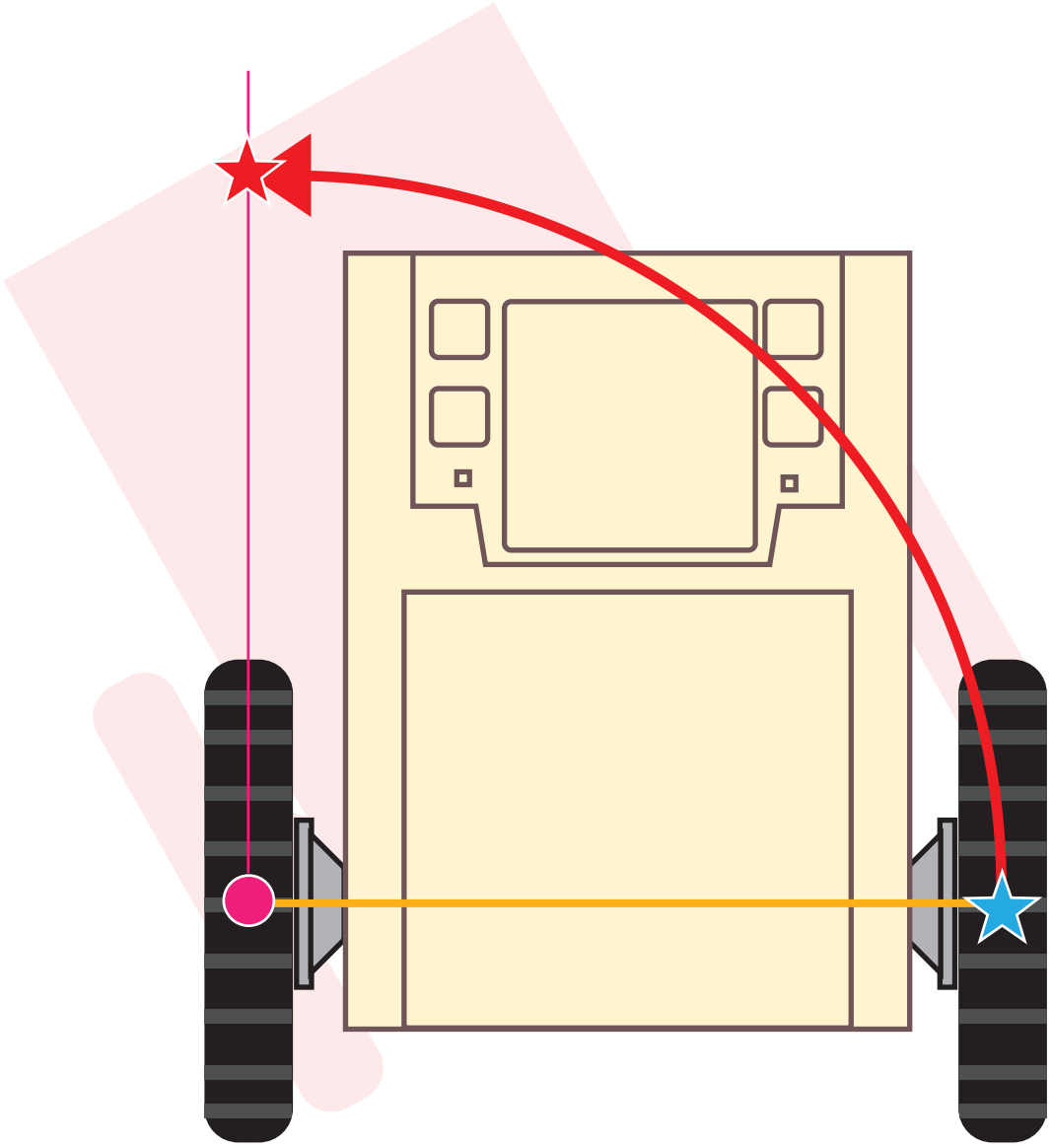


③左折

右(M2) の DC モーターだけを逆転させることで左折ができます。イラストに合わせて、ぴったり 90 度左折するために必要な時間を調べましょう

左のプログラムをつくり、「待つ」ブロックの時間を調整しましょう。

□ **秒で 90 度左折する。**

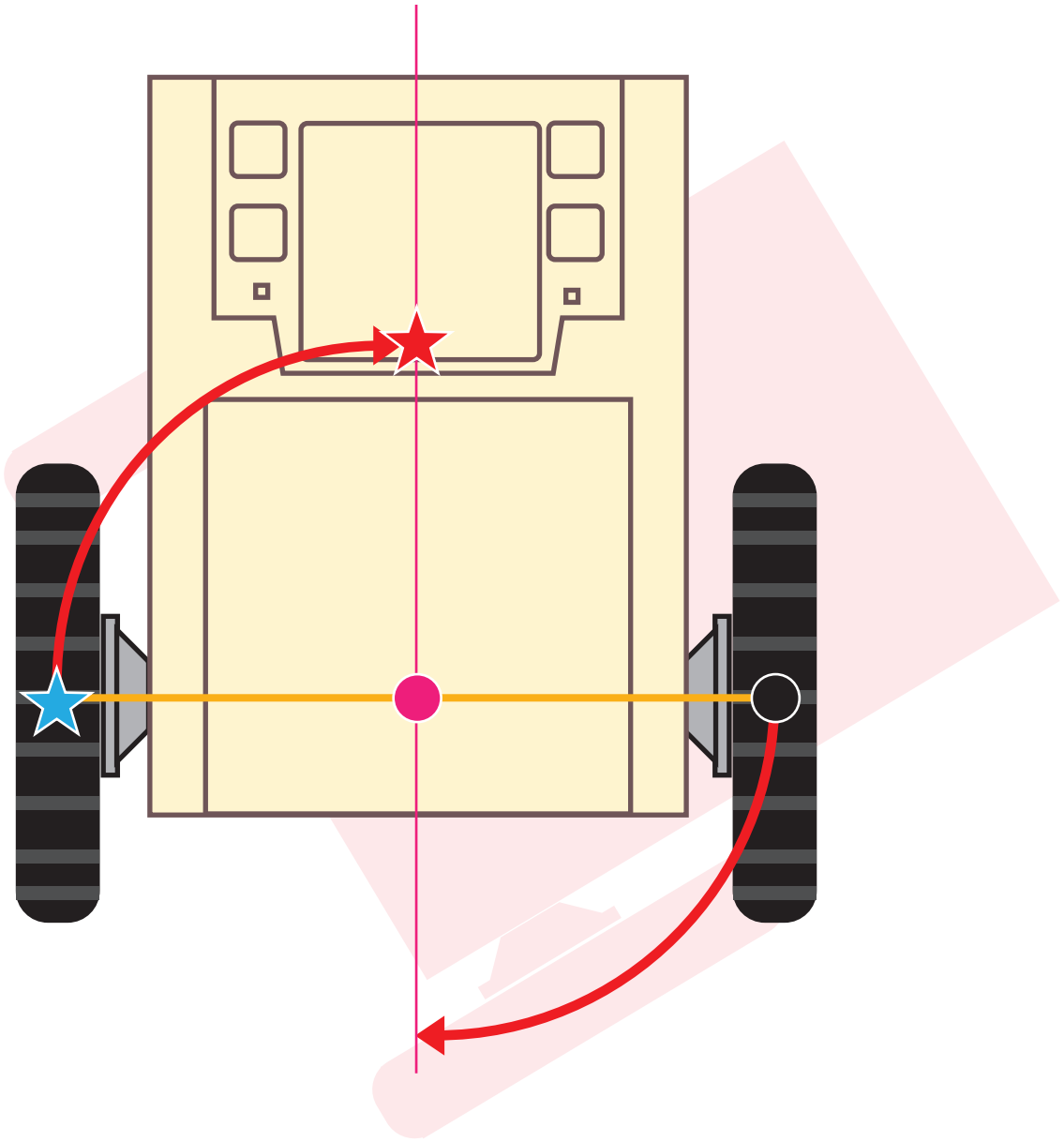


④右回転

左 (M1) の DC モーターを逆転させ、右 (M2) の DC モーターを正転させることで右回転ができます。イラストに合わせて、ぴったり 90 度右に回転するために必要な時間を調べましょう。

左のプログラムをつくり、「待つ」ブロックの時間を調整しましょう。

□ **秒で 90 度右に回転する。**

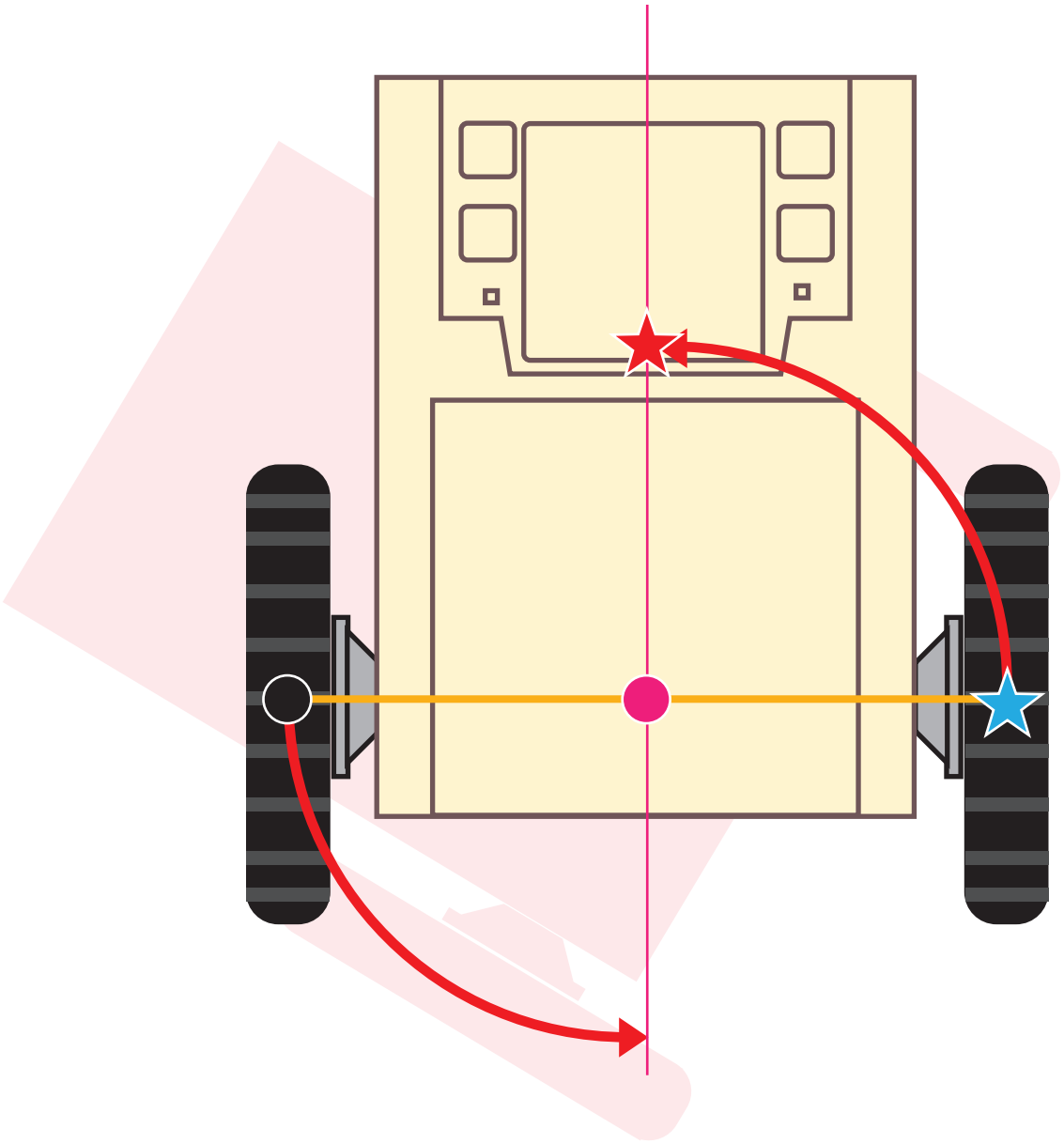


⑤左回転

右 (M2) の DC モーターを逆転させ、左 (M1) の DC モーターを正転させることで左回転ができます。イラストに合わせて、ぴったり 90 度左に回転するために必要な時間を調べましょう。

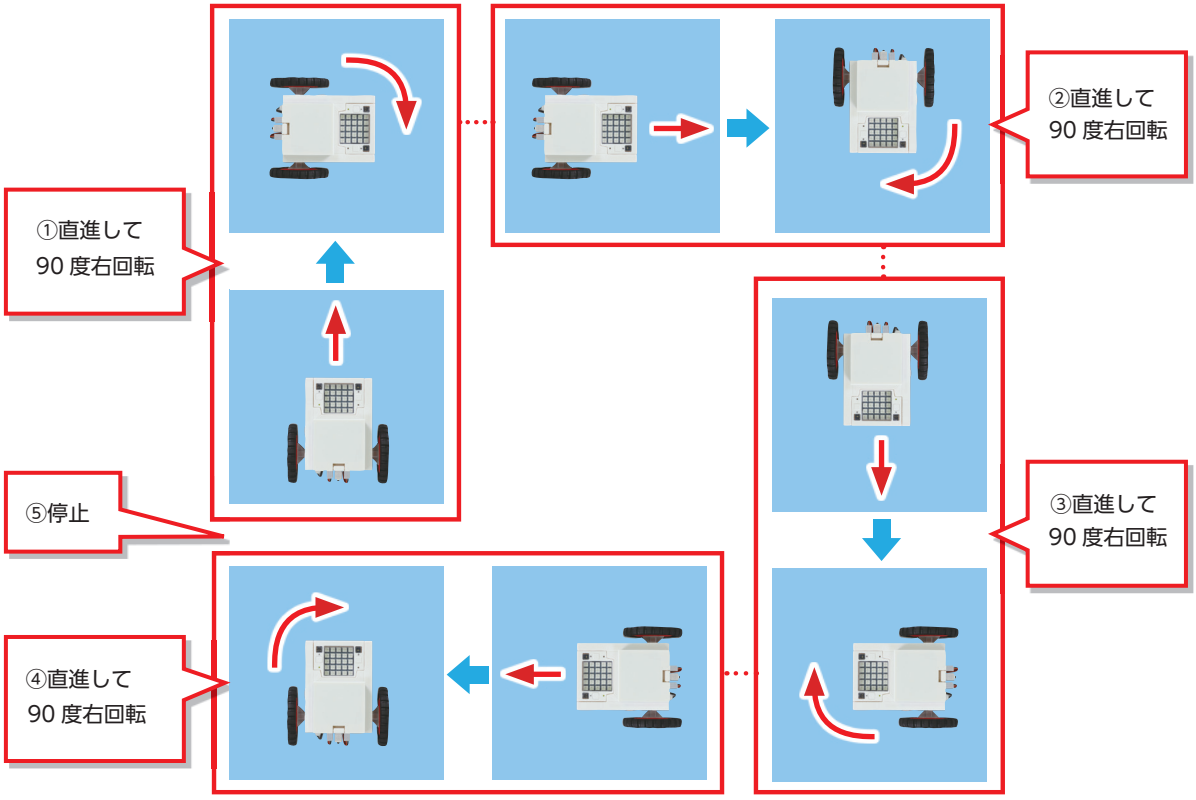
左のプログラムをつくり、「待つ」ブロックの時間を調整しましょう。

［　　］秒で 90 度左に回転する。



練習課題

直進と右回転を繰り返すことで、１周回るプログラムをつくりましょう。

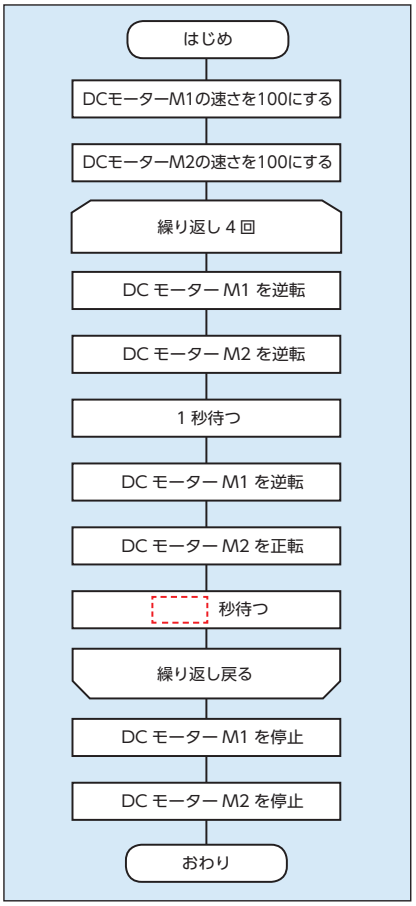


①フローチャート

直進と右回転を4回繰り返すと1周回ることになります。同じ動作を複数回繰り返す場合、以下の記号を使います。動作を表すフローチャートをまとめましょう。

繰り返し○回	繰り返し処理の開始
繰り返し戻る	繰り返し処理の終了

※右回転する秒数は41ページで確認した秒数を参考にしてください。



②プログラミング

まとめたフローチャートをもとにプログラムをつくりましょう。同じ動きを複数回繰り返す場合、 

を利用すると簡単にプログラムをつくることができます。 

に囲まれたブロックが指定した回数だけ繰り返し実行されます。

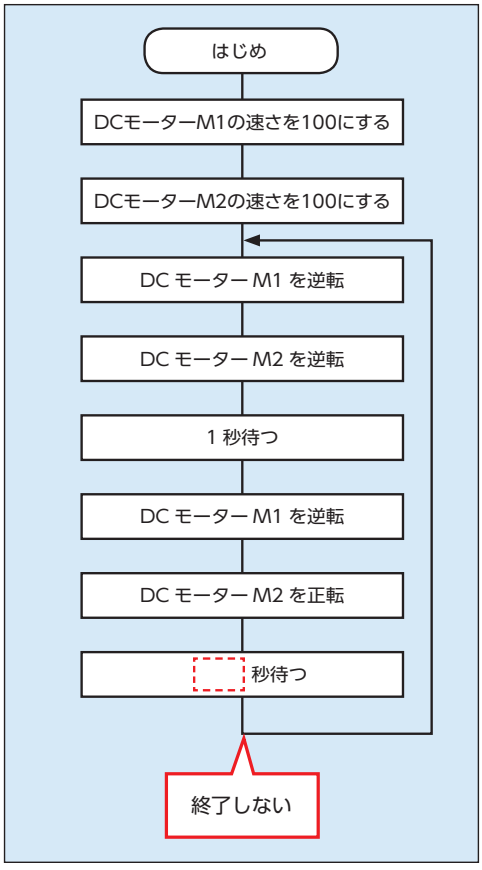
③無限に繰り返しをするプログラム

同じ動きをずっと繰り返させたい場合は、 

を利用します。

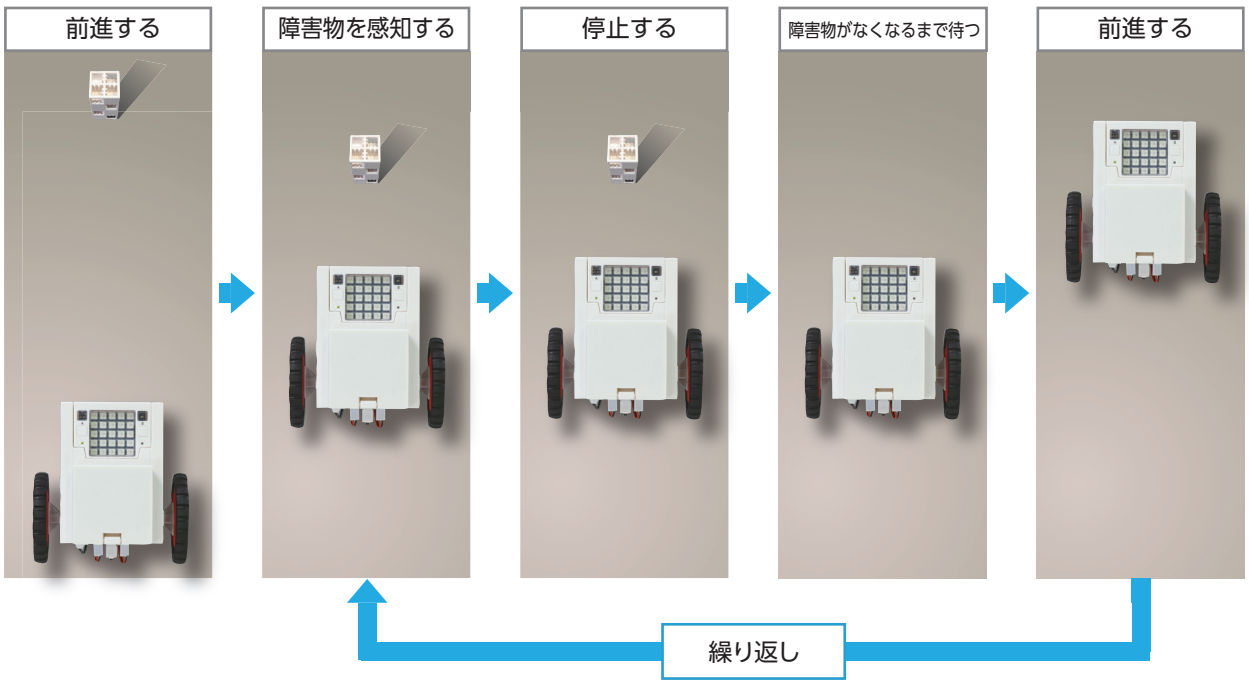
「４回繰り返す」を「ずっと」のブロックに変更しましょう。

「終了」がなくなり、繰り返し続けることは右のフローチャートのように矢印で表すこともできます。※停止する必要がなくなるので、停止ブロックは削除してください。



3．衝突回避カーの製作

赤外線フォトリフレクタを利用して衝突回避カーをつくりましょう。

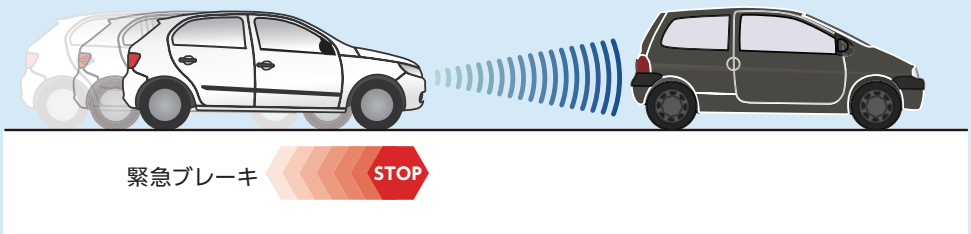


自動車の運転を補助するシステム

自動車にはセンサーを用いた運転を補助する様々なシステムが存在します。

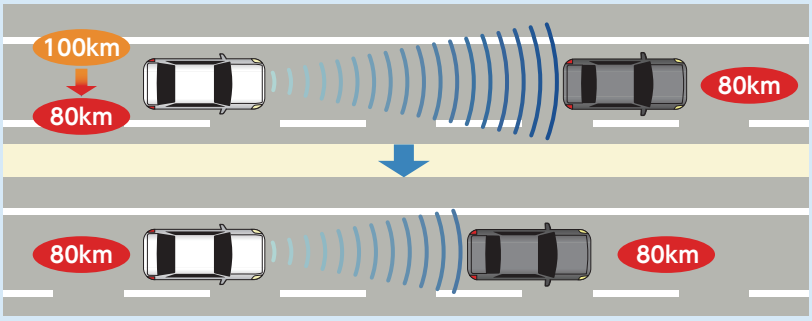
自動ブレーキシステム

前を走る自動車や歩行者、障害物にぶつかりそうになったときに自動でブレーキをかけて停止します。



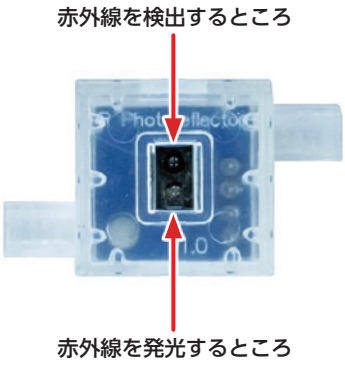
速度自動調整システム

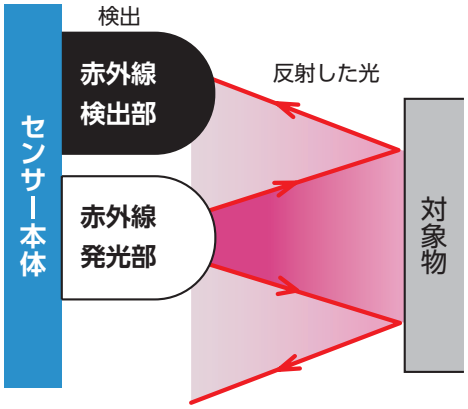
高速道路などで前を走る自動車と安全な距離を保ちながら、自動で速さを調整します。長い時間の運転や、渋滞中の負担を少なくすることができます。



①赤外線フォトリフレクタとは

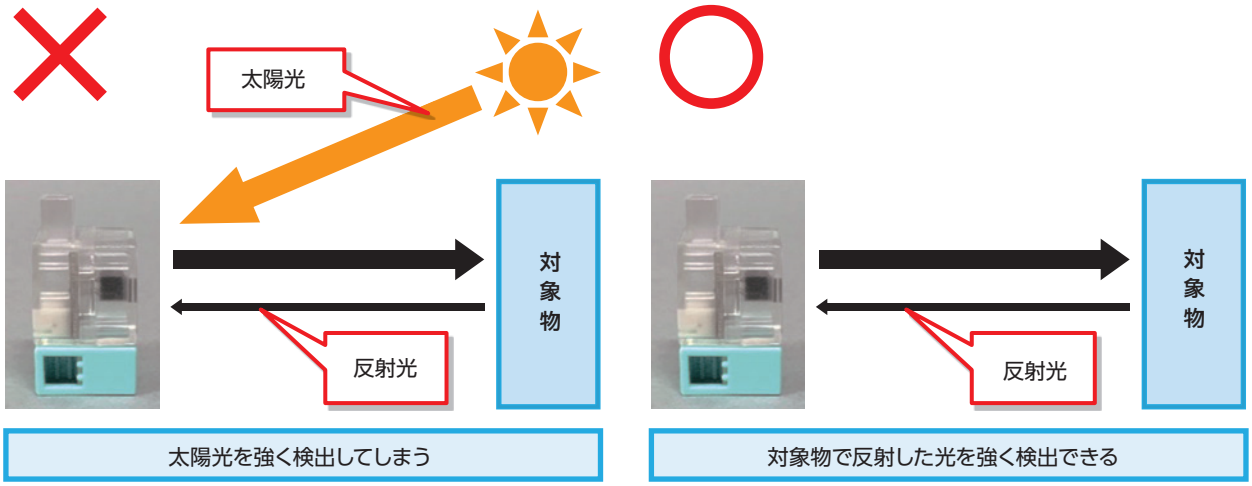
赤外線フォトリフレクタは「赤外線」という光の「反射」を利用したセンサーです。中央の２つの丸い部分は透明な方が赤外線を発光し、黒い方は赤外線を検出します。発光部から出た赤外線が対象物にぶつかって反射してきたところを検出部で読み取ります。そのときの赤外線の強さで、対象物が存在するかどうかを知ることができます。



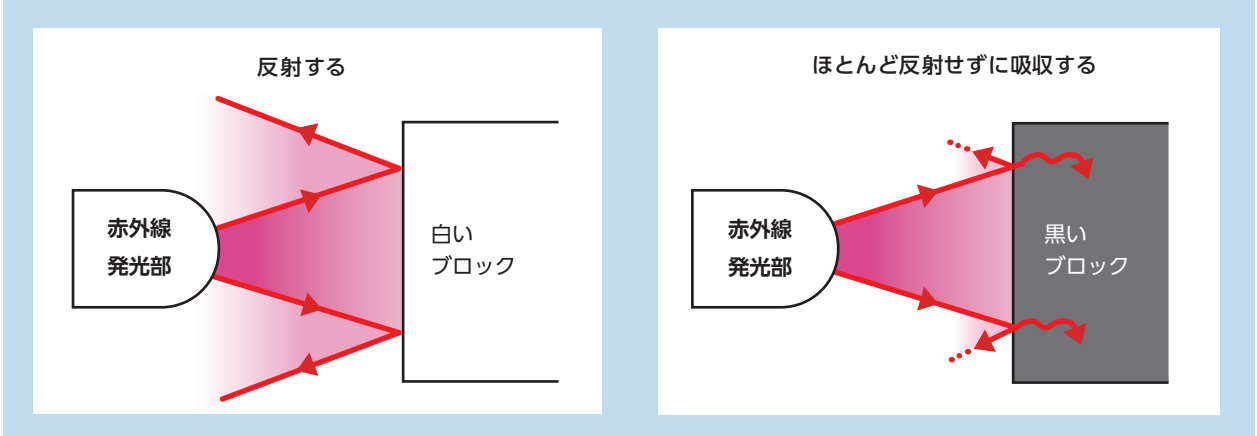


赤外線フォトリフレクタの使い方

・太陽光には赤外線が多く含まれています。太陽光が検出部に当たらないように注意してください。

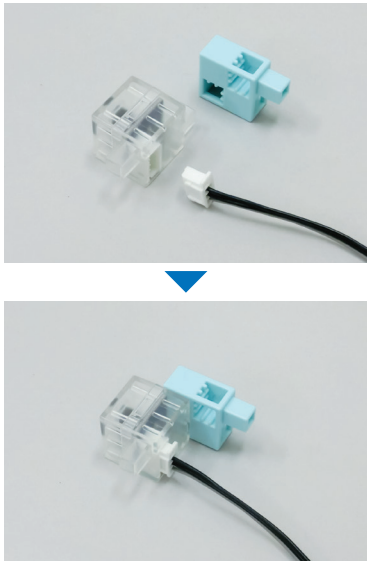


・黒系の色は赤外線を吸収しやすく反射しにくいため、反射光の検出量が少なくなります。

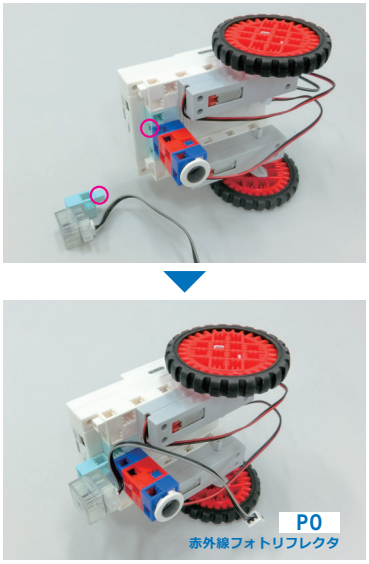


②「衝突回避カー」の組み立て

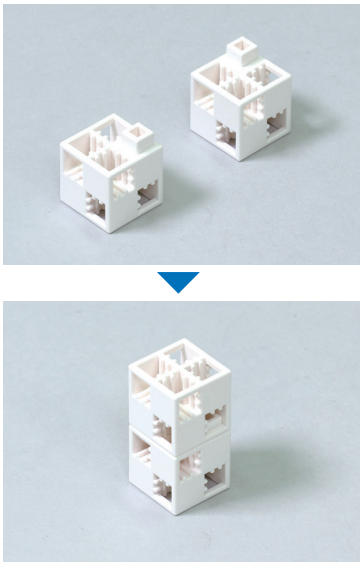
①

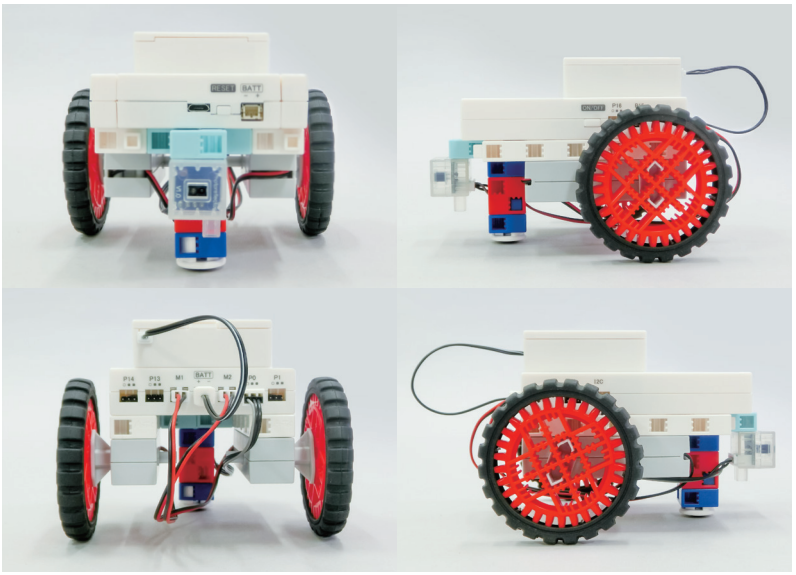


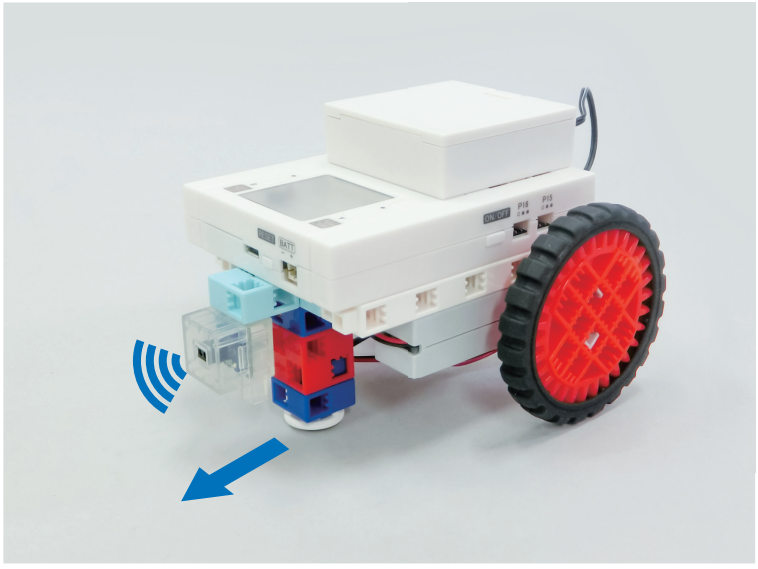
②



③

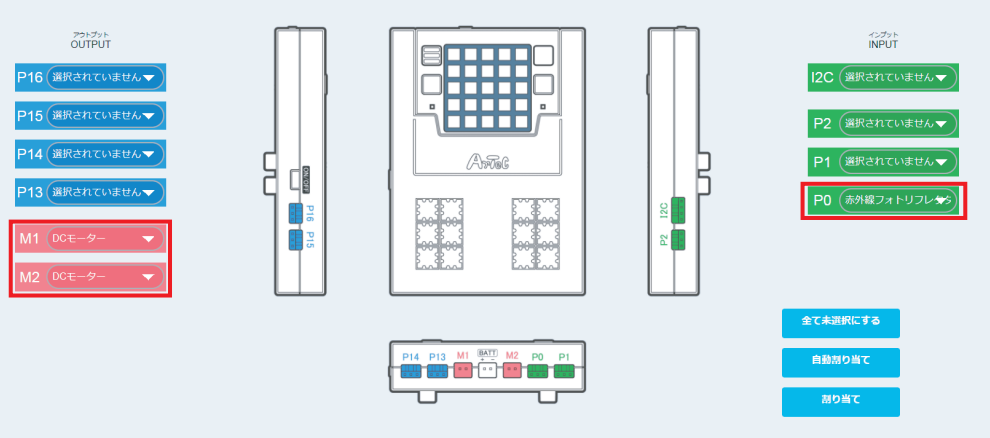






③入出力設定

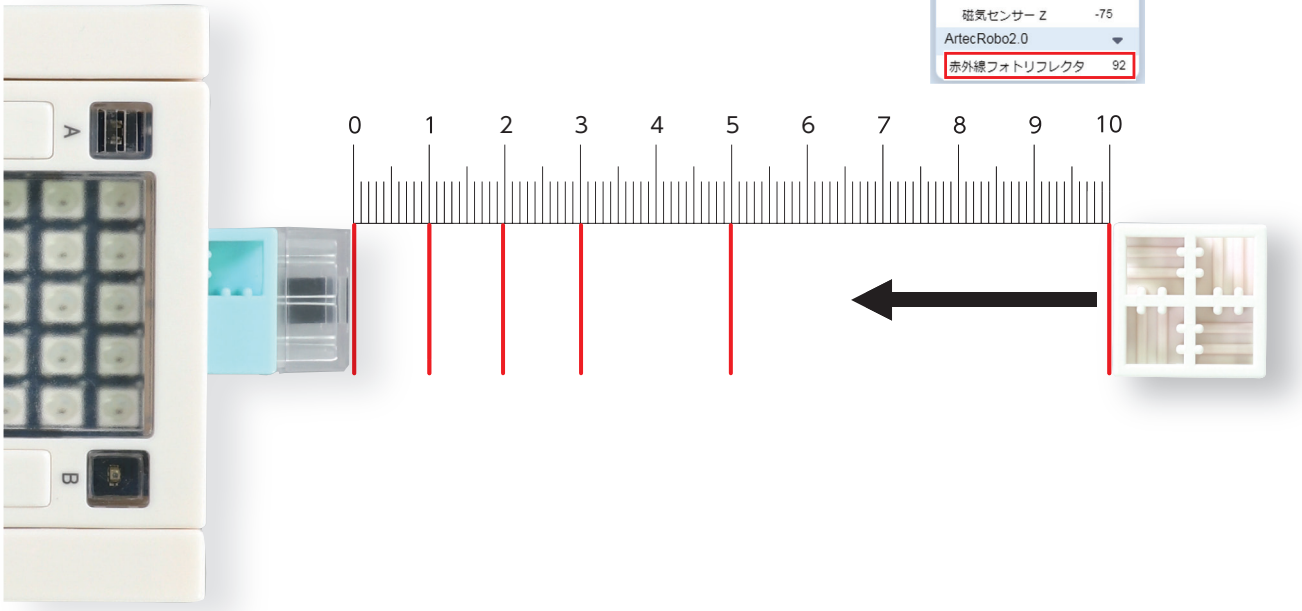
M1、M2にDCモーター、P0に赤外線フォトリフレクタを選択します。



④赤外線フォトリフレクタの数値確認

衝突回避カーは前に障害物が存在したときにセンサーで感知して衝突を避けるため自動で停止します。この「感知」の役割を赤外線フォトリフレクタで行います。テストモードにして、センサー・ボードで赤外線フォトリフレクタの値を確認しましょう。下のイラストに合わせて、車を置きます。赤外線フォトリフレクタの前 10cm、5cm、3cm、2cm、1cm、0cm に障害物（ブロック白四角）があるときの値を確認しましょう。

※赤外線フォトリフレクタの値は太陽光の影響を受けるため、日光のあたる窓に向けないようにしてください。

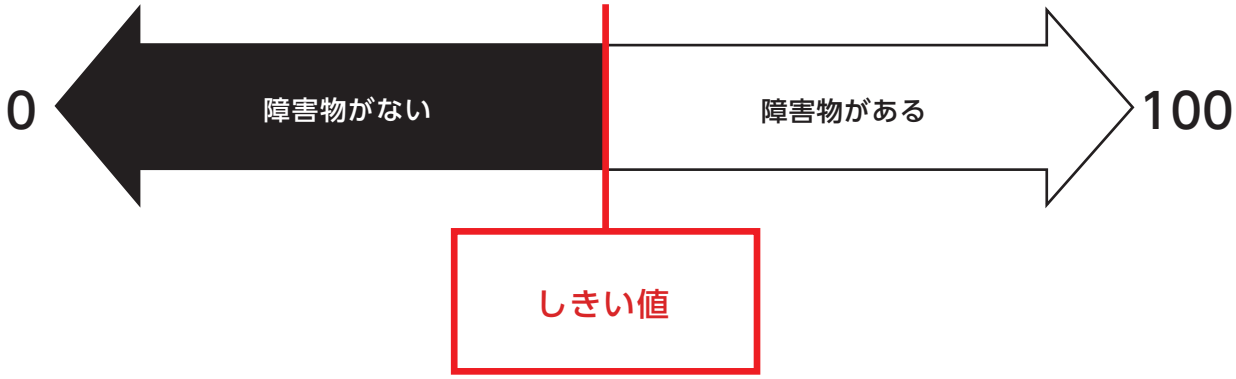


距離	0cm	1cm	2cm	3cm	5cm	10cm
値						

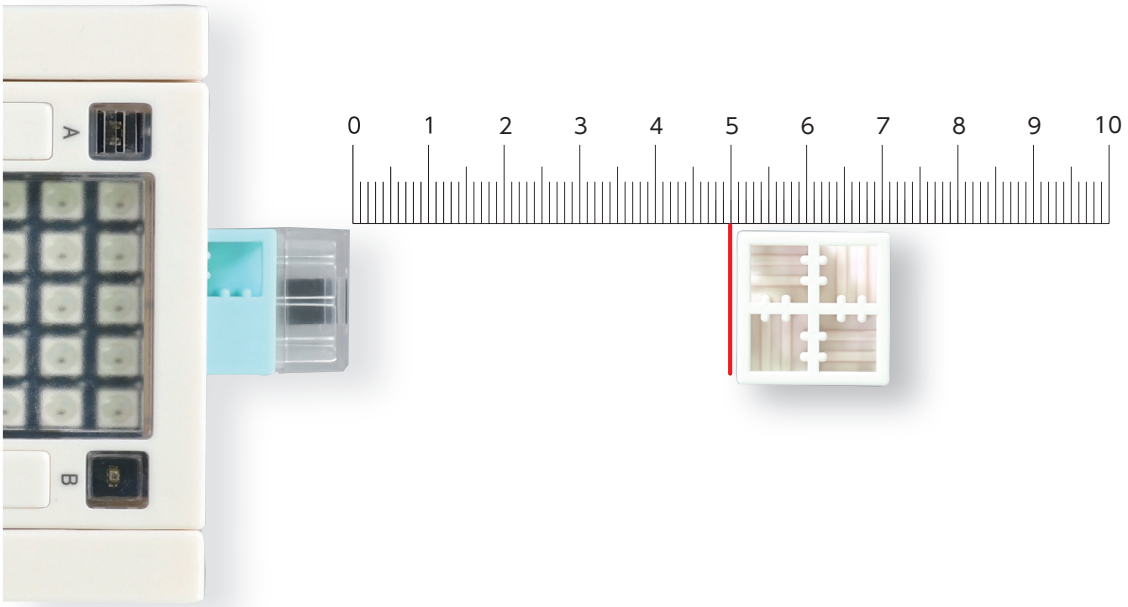
※赤外線フォトリフレクタの個体差で、同じ距離でも数値が異なる場合があります。

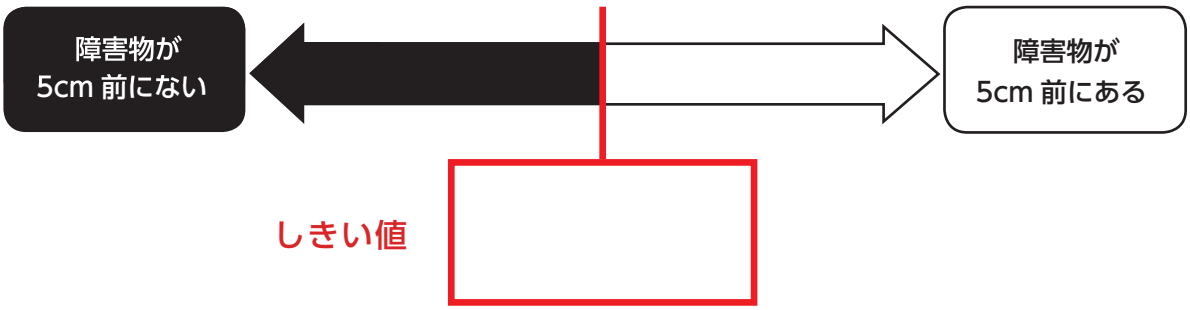
⑤しきい値の決定

赤外線フォトリフレクタの値で障害物があるときとないときを区別する場合は、２つの状態の境目となる値を決めて判断します。このような境目となる値のことを「しきい値」と言います。



今回は車の前５cmに障害物があるときとないときを区別するようにしきい値を決めましょう。



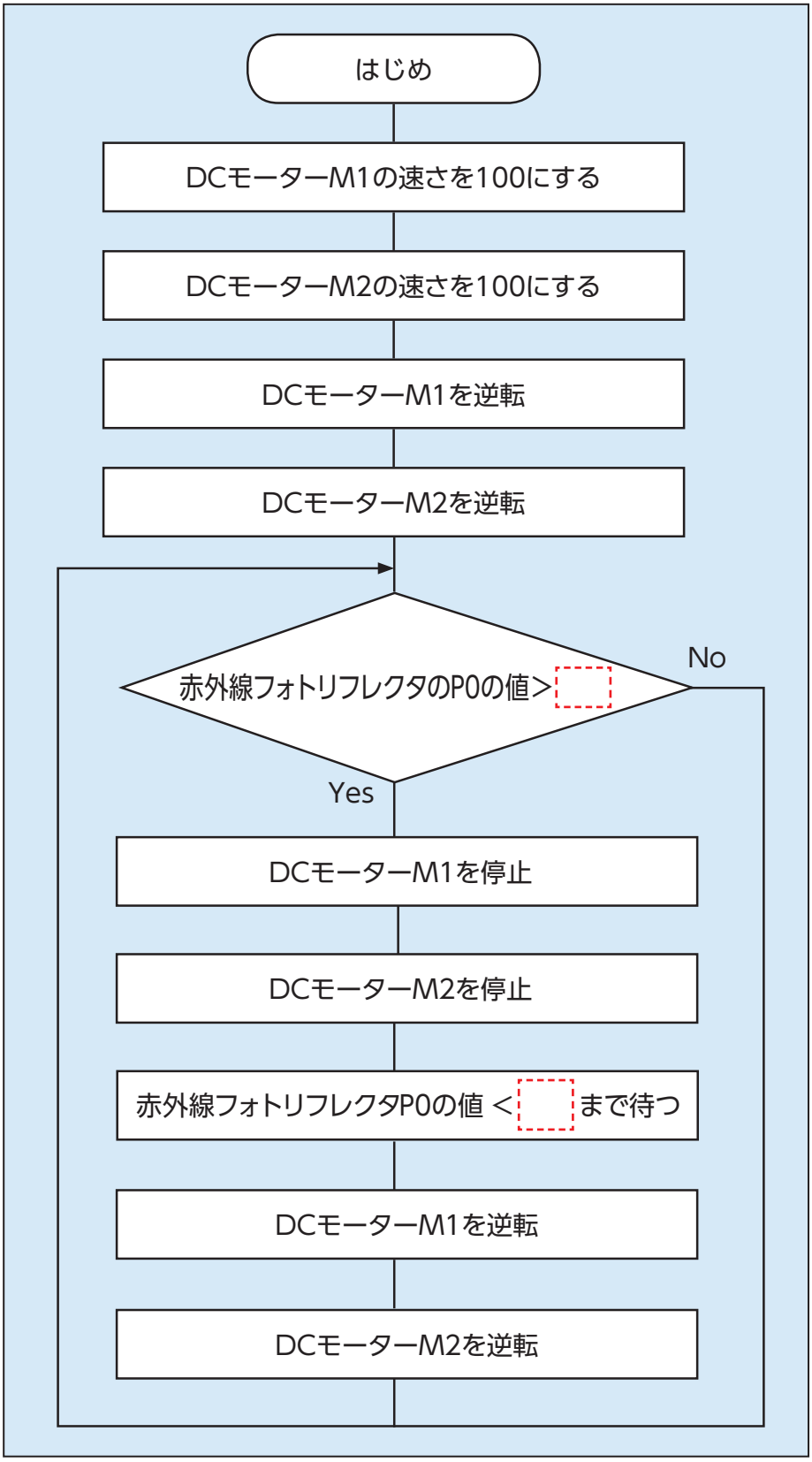


⑥フローチャート

45ページに示した動作を実現できるように手順を考えてフローチャートをまとめましょう。

条件の部分を決めたしきい値を利用して書き換えましょう。また、「障害物が無くなるまで待つ」は

赤外線フォトリフレクタの値 < [　] まで待つ　を使いましょう。



⑦プログラミング

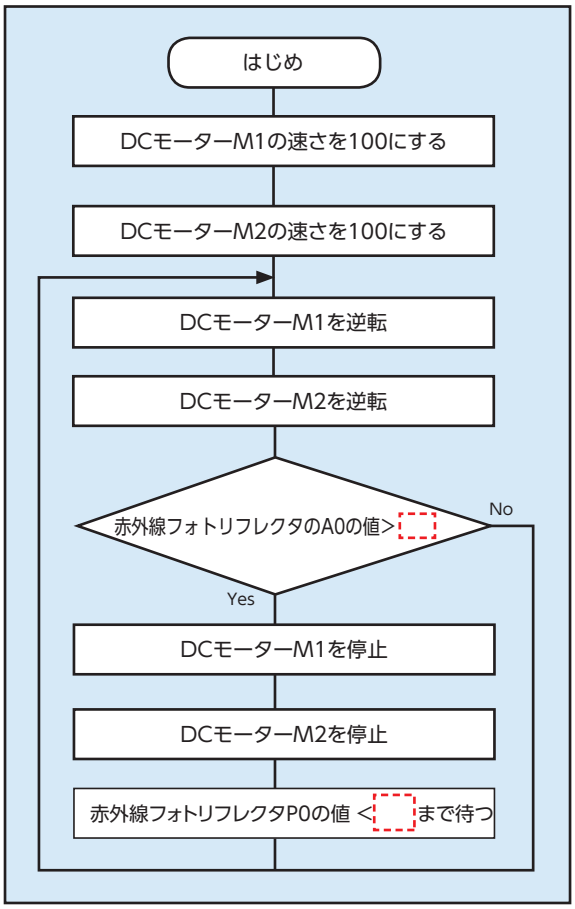
「制御」カテゴリーにある 

を使ってプログラムをつくりましょう。 

を使うと、条件を満たすまでプログラムの処理を待つことができます。

衝突回避カーの別解

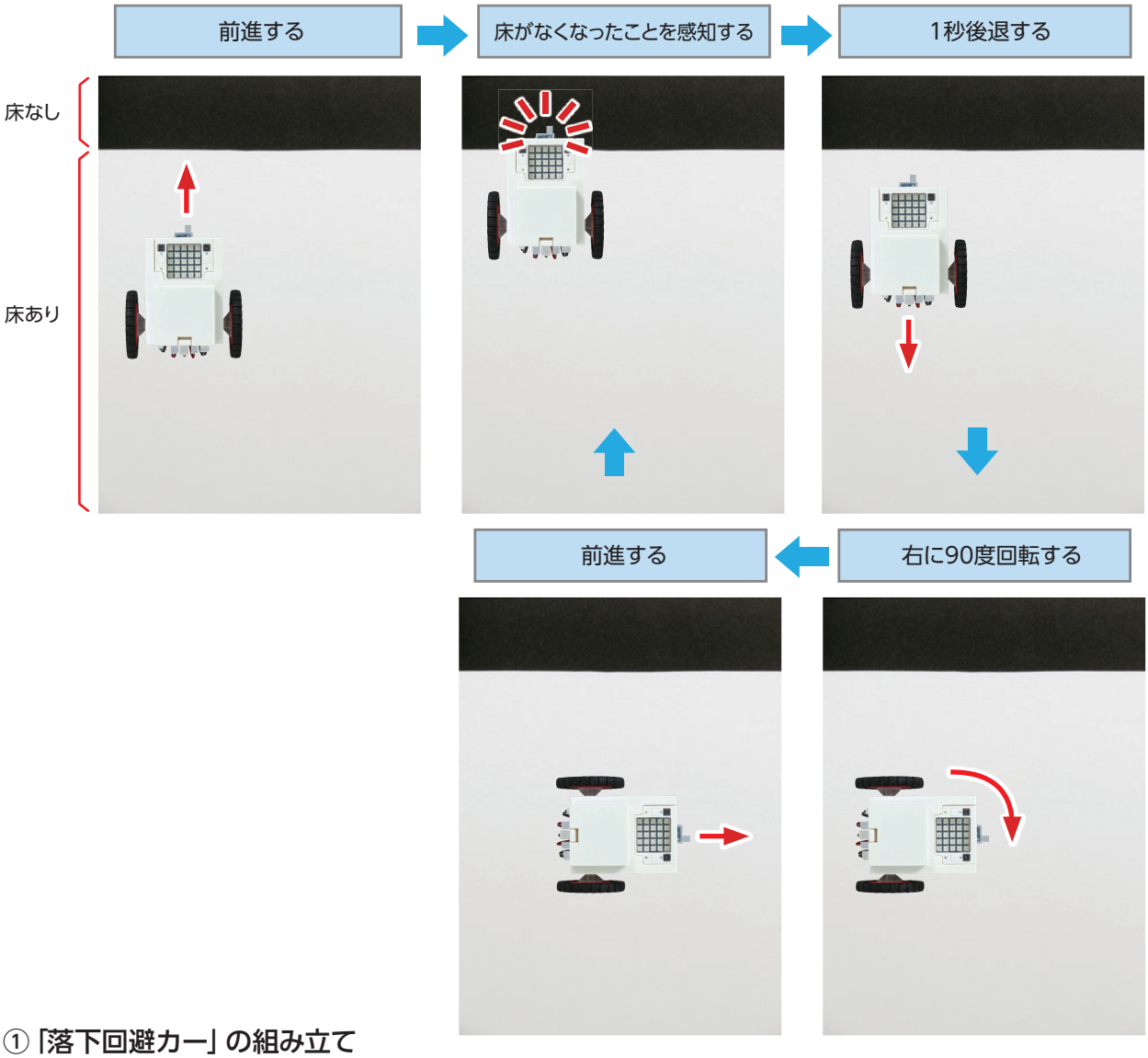
以下のプログラムでも衝突回避カーの動作が実現できます。



プログラムのつくり方は一つとは限りません。想定する動作が実現できればどのようなプログラムでも構いません。自分なりに考えてプログラムをつくることが大切です。

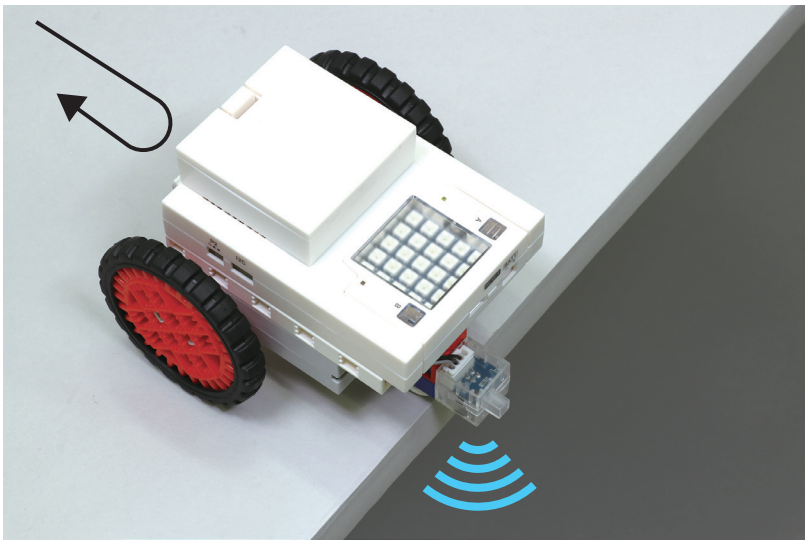
4．落下回避カーの製作

赤外線フォトリフレクタの使い方を工夫することで、机から落下しそうになったときに後退して落下を避ける車をつくることができます。

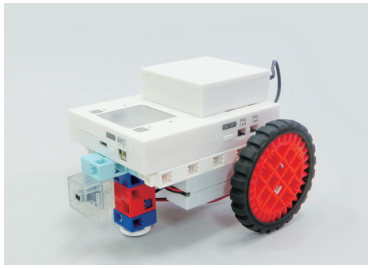


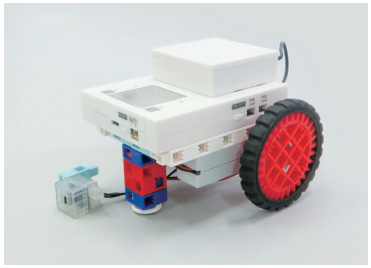
①「落下回避カー」の組み立て

今回は床の有無を感知するので、赤外線フォトリフレクタの向きを下向きに付け替えましょう。

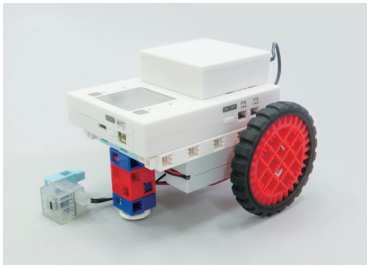


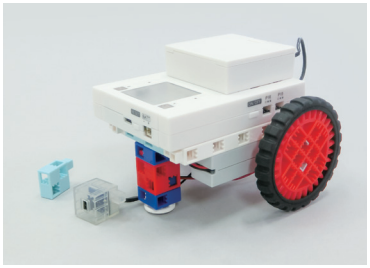
①



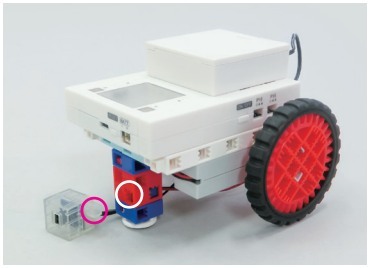


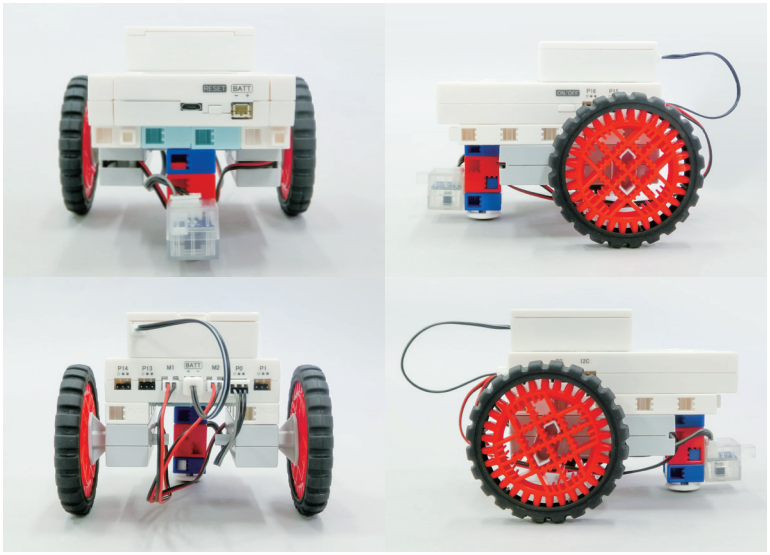
②

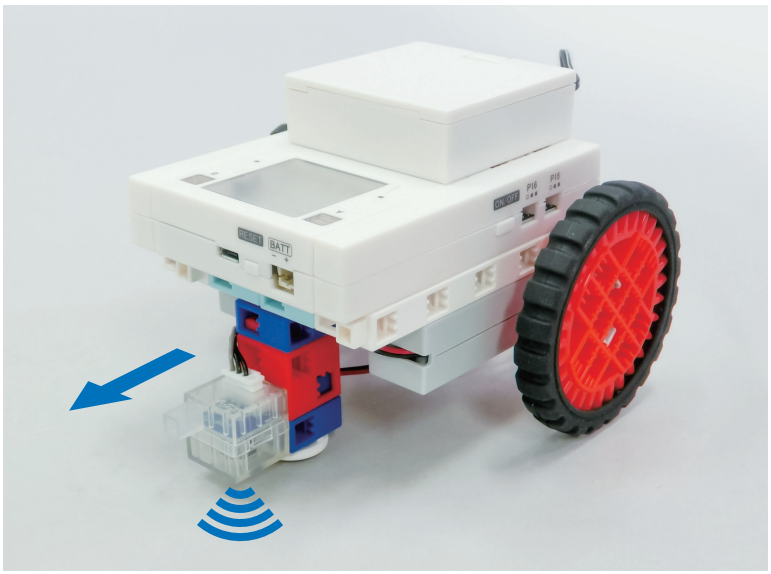




③

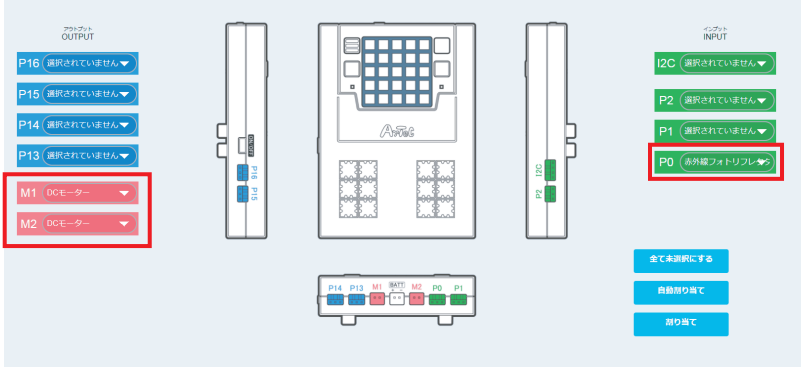






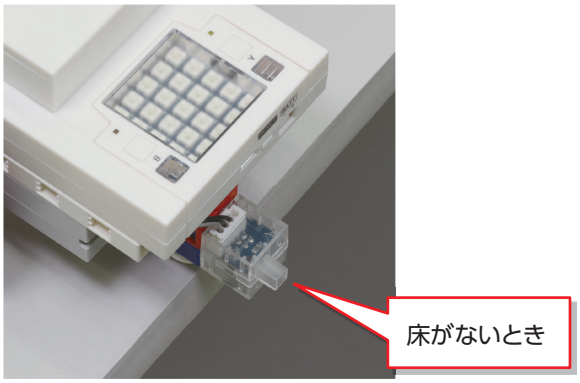
②入出力設定

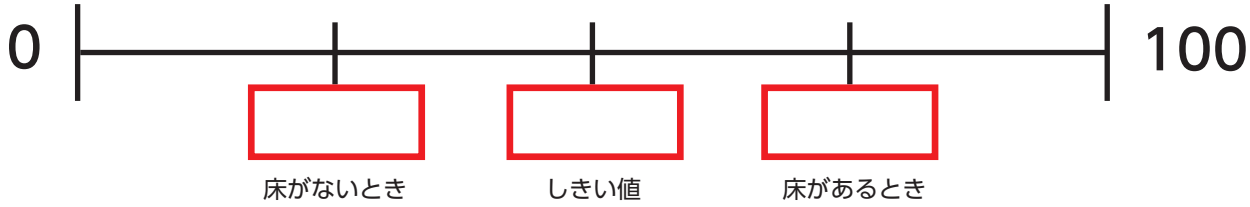
M1とM2にDCモーター、P0に赤外線フォトリフレクタを選択します。



③しきい値の決定

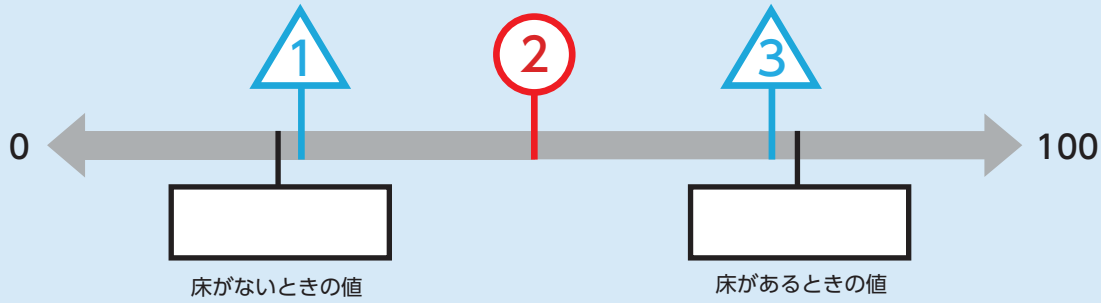
テストモードで床があるときとないときの赤外線フォトリフレクタの値を確認して、しきい値を決めましょう。





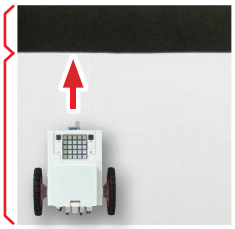
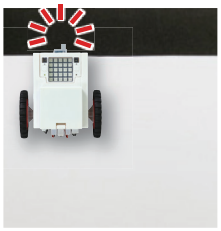
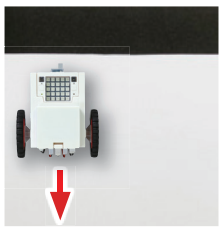
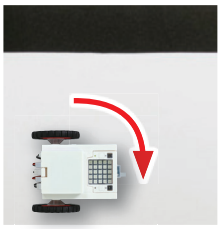
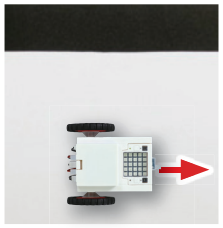
しきい値の決定

しきい値を ① や ③ のように調べたどちらかの値の近くに決めてしまうと、赤外線フォトリフレクタの値がそこから少し変わるだけで、ブロックの有無を区別してしまいます。センサの値は周りの環境によって変わりやすいため誤って判断しないように、② のように調べた２つの値の中央あたりをしきい値として決めましょう。



④落下回避カーの動作整理

動作とプログラムの関係を整理しましょう。

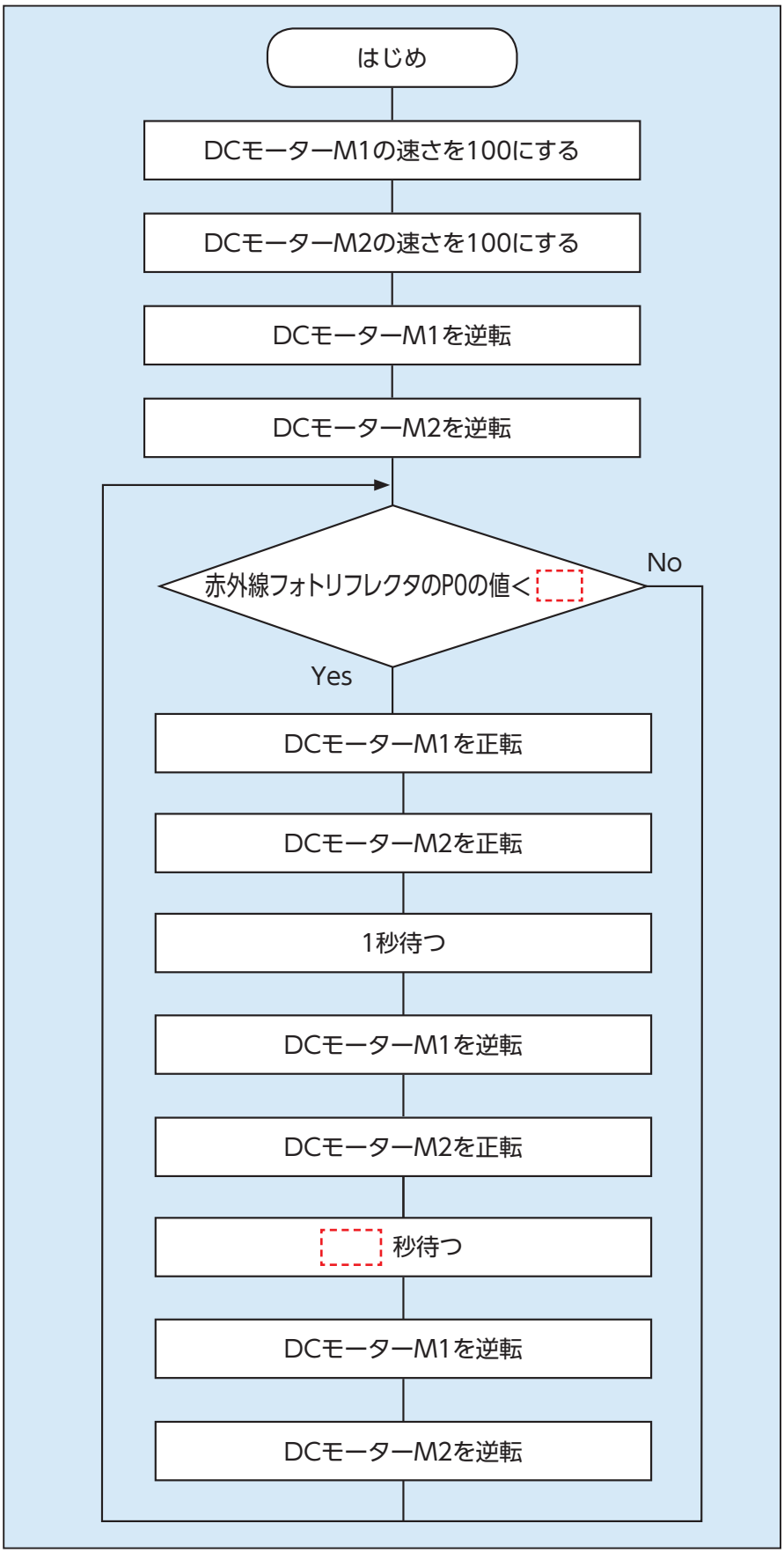
順番	動作	プログラム
①	前進する 床なし 床あり 	DCモーター M1 の速さを100にする DCモーター M2 の速さを100にする DCモーター M1 を逆転 DCモーター M2 を逆転
②	床がなくなったことを感知する 	赤外線フォトリフレクタ P0 の値 ＜ しきい値
③	1秒後退する 	DCモーター M1 を正転 DCモーター M2 を正転 1秒待つ
④	右に90度回転する 	DCモーター M1 を逆転 DCモーター M2 を正転 □ 秒待つ
⑤	前進する 	DCモーター M1 を逆転 DCモーター M2 を逆転

②～⑤を
ずっと繰り返す

※右に90度回転させる秒数は54ページで調べたものを参考にしてください。

⑤フローチャート

55ページで整理した動作をフローチャートにまとめましょう。

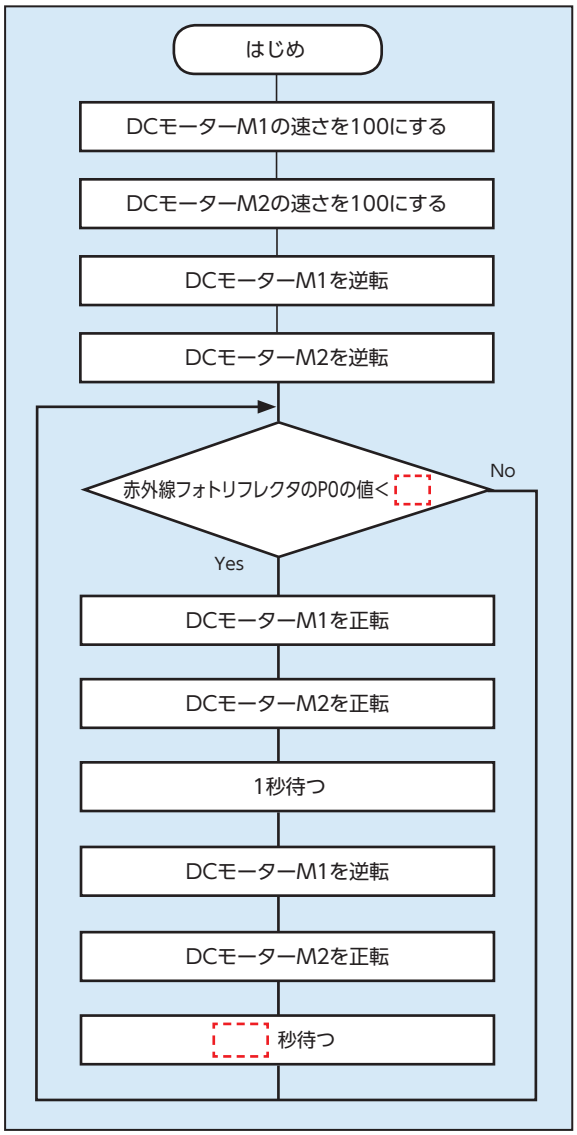


⑥プログラミング

56ページのフローチャートを参考にプログラムを完成させましょう。

落下回避カーの別解

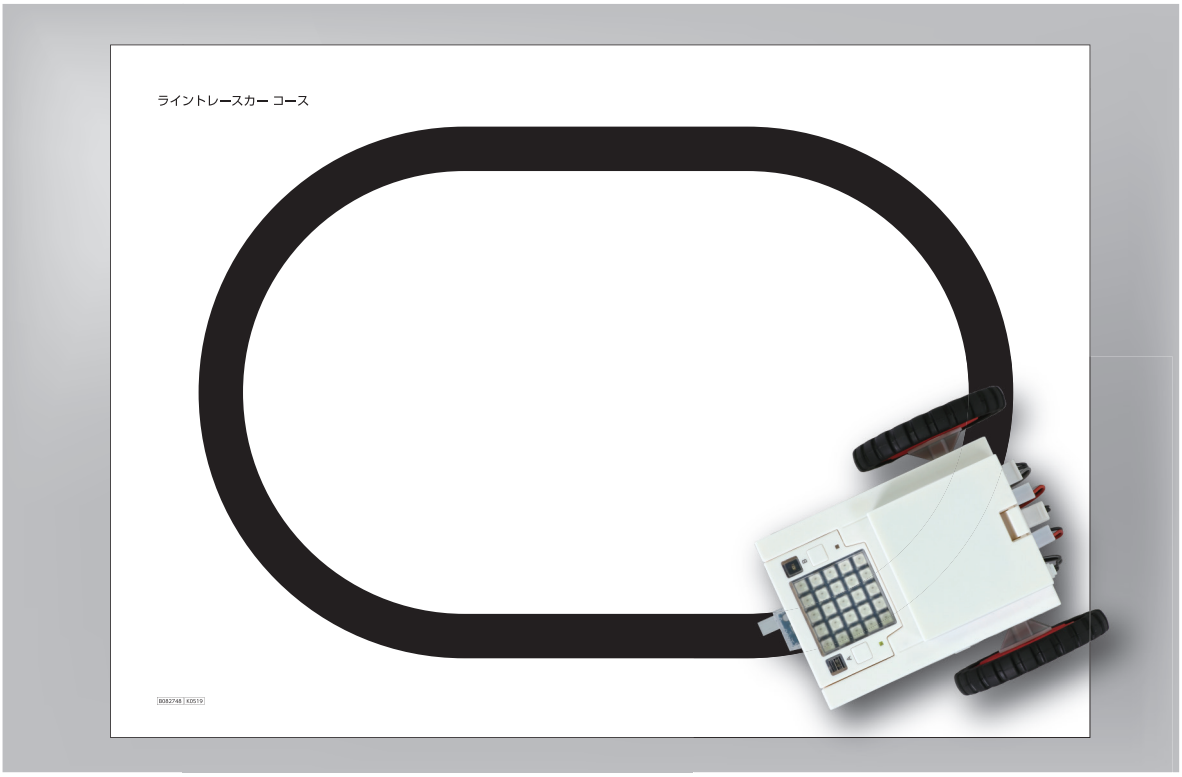
以下のプログラムでも落下回避カーの動作が実現できます。



プログラムのつくり方は一つとは限りません。想定する動作が実現できればどのようなプログラムでも構いません。自分なりに考えてプログラムをつくることが大切です。

5．ライントレースカーの製作

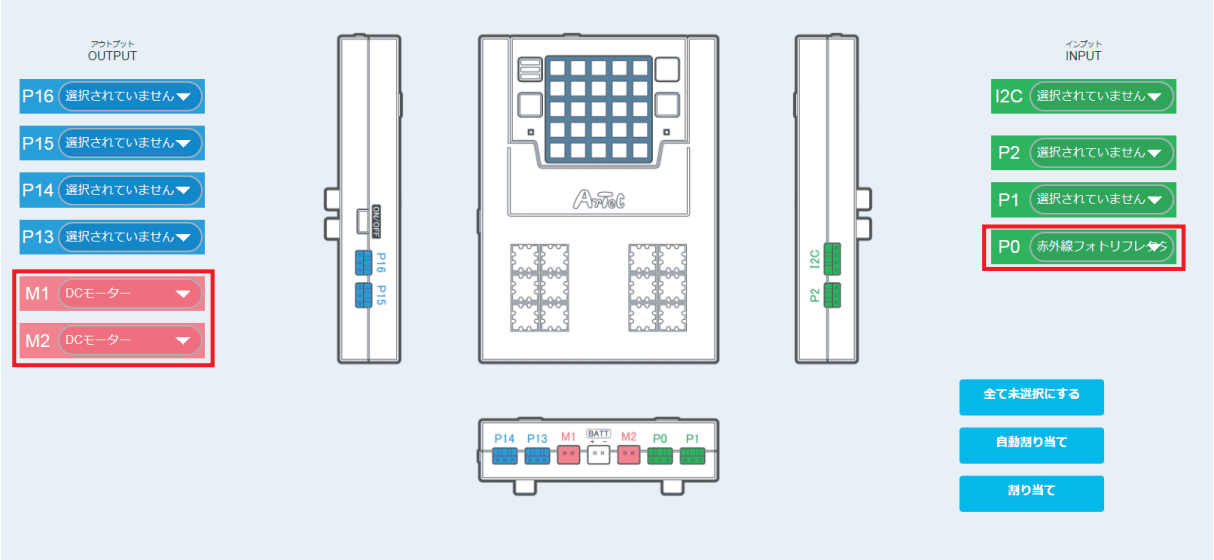
色によって赤外線の反射率が違うことを利用すると、赤外線フォトリフレクタで黒い線を認識することができます。
ライントレースとは、センサーに線を認識させて線に沿って車を走らせることを言います。



車は「4 落下回避カーの製作」で組み立てたものをそのまま使いましょう。

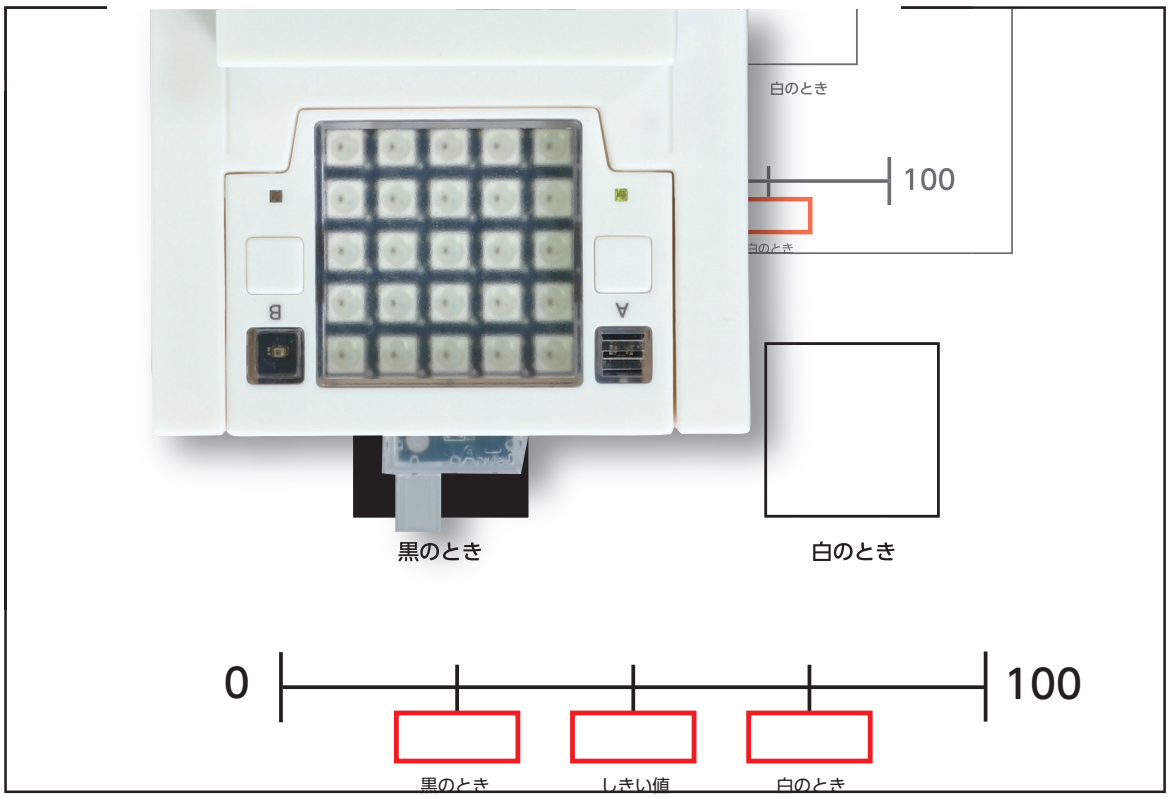
①入出力設定

M1とM2にDCモーター、P0に赤外線フォトリフレクタとなるように選択しましょう。

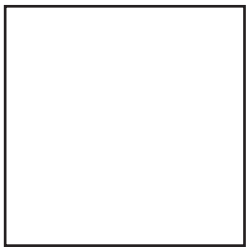


②しきい値の決定

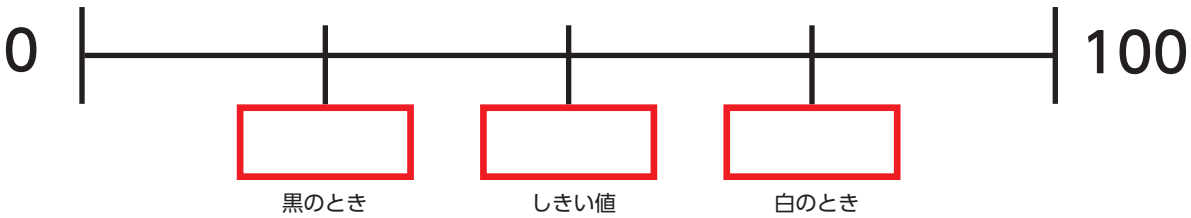
テストモードで黒い線があるときとないときの赤外線フォトリフレクタの値を下のイラストを使って確認して、しきい値を決めましょう。赤外線は黒い色に反射されにくいため、黒のときは値が小さく、白のときは値が大きくなります。



黒のとき

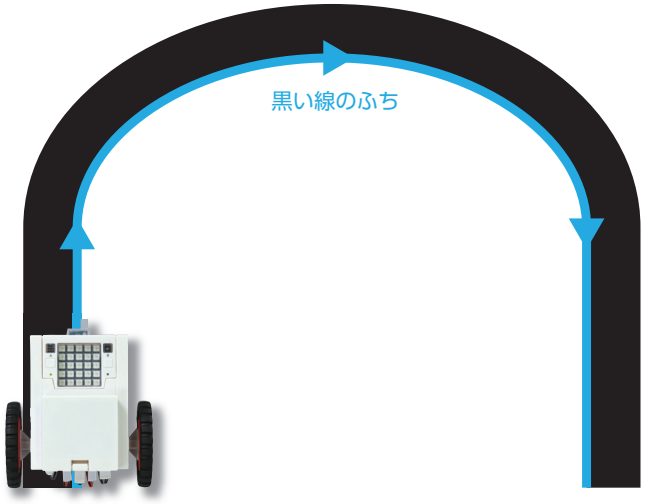


白のとき

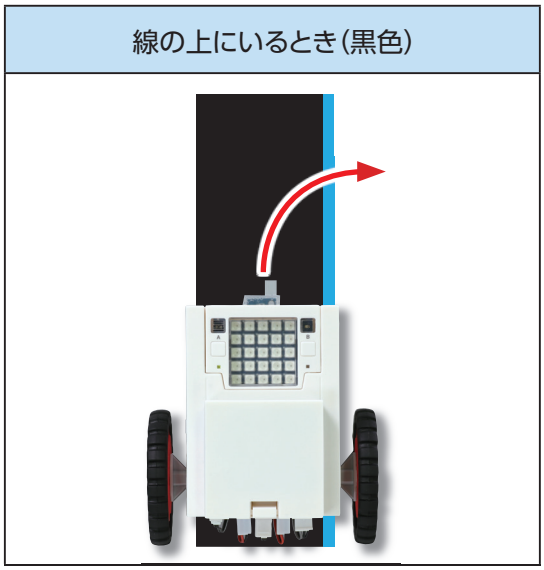


③ライントレースカーの動作整理

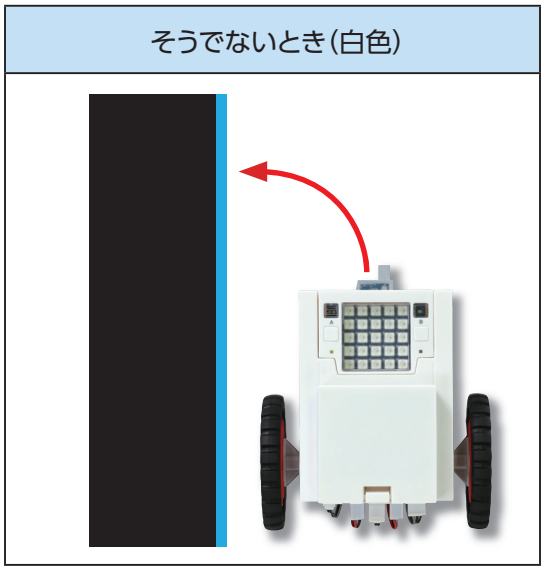
黒い線に沿って走らせる方法の一つとして、黒い線の上ではなく、黒い線のふち（黒い線とまわりとの境界線）に沿って走らせるものがあります。



黒い線のふちに沿って走らせるには、例えば線の上にいるとき（黒色）と、そうでないとき（白色）で次のように DC モーターを制御します。

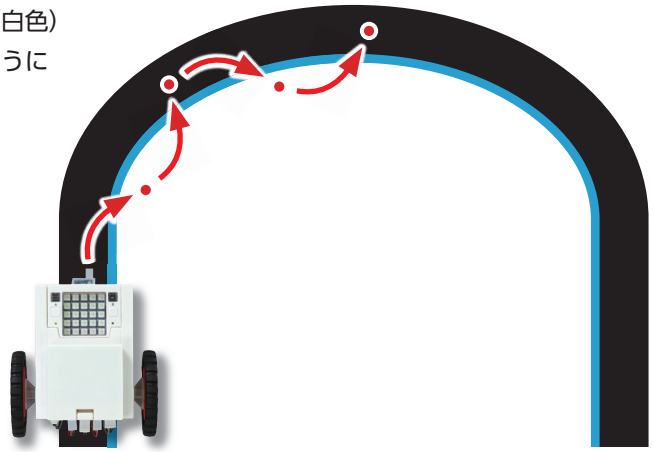


黒い線のふちに近づくように右折する

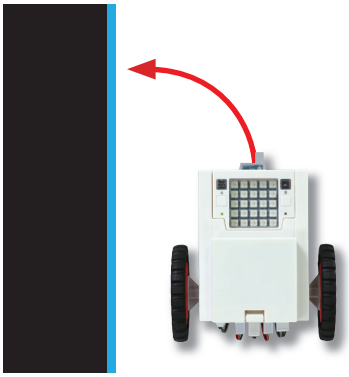


黒い線のふちに近づくように左折する

この動きを繰り返すことで、線の上（黒色）とその外（白色）を行ったり来たりしながら線のふちに沿って走るようになります。

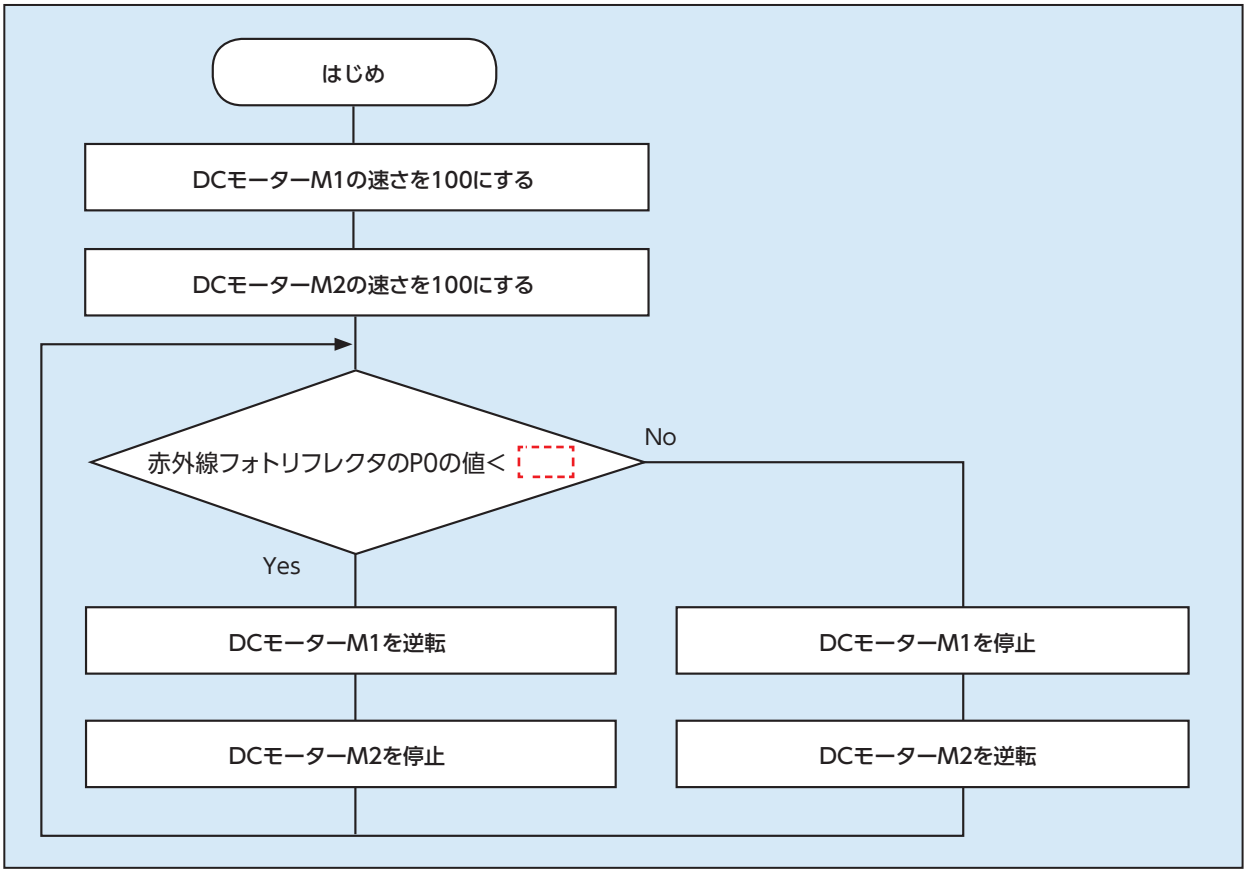


この動きをまとめると、以下のようになります。

	線の上にいるとき(黒色)	そうでないとき(白色)
状態		
条件	赤外線フォトリフレクタ P0の値 <しきい値	左の条件以外のとき
動作	右折する	左折する
プログラム	DCモーター M1 を逆転 DCモーター M2 を停止	DCモーター M1 を停止 DCモーター M2 を逆転

④フローチャート

整理した動作を実現できるように手順を考えてフローチャートをまとめましょう。今回はしきい値を超えるときと超えないときでそれぞれ違う動作をする必要があることに注意しましょう。



⑤プログラミング

まとめたフローチャートをもとにプログラムを作成しましょう。「制御」カテゴリーにある  を使うことに注意しましょう。  を使うと、指定した条件が成り立つときと成り立たないときで処理を分けることができます。

ライントレースカーの別解

以下のプログラムでもライントレースカーの動作が実現できます。  の代わりに、  を2つ並べているだけで、行っている処理はほとんど同じです。

応用学習 テーマ1

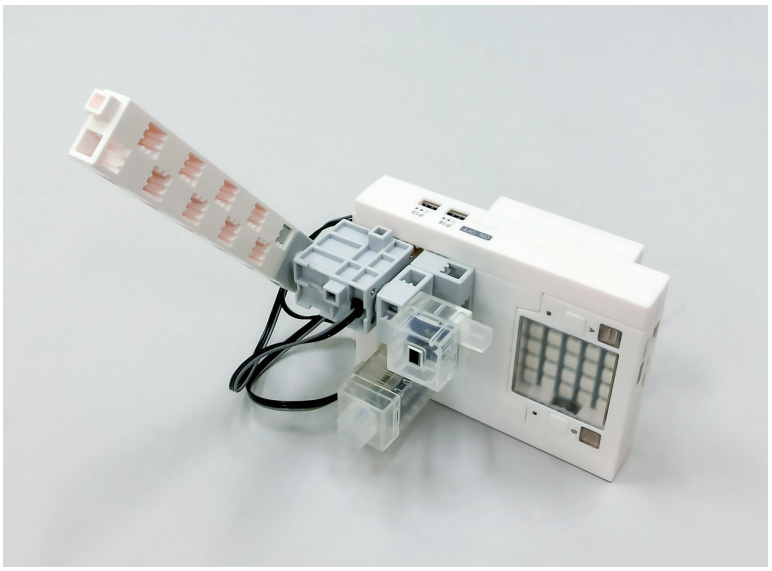
自動ゲートシステム

＜学習内容＞

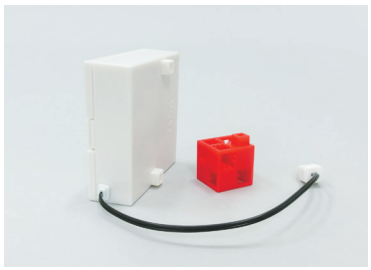
●サーボモーターの制御

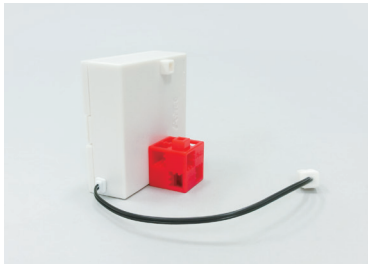
●複数のセンサーによる制御

①「自動ゲート」の組み立て

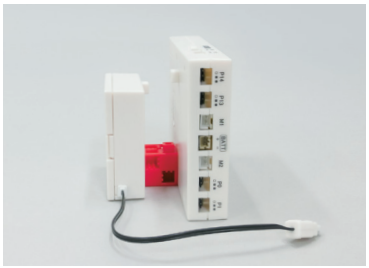


①

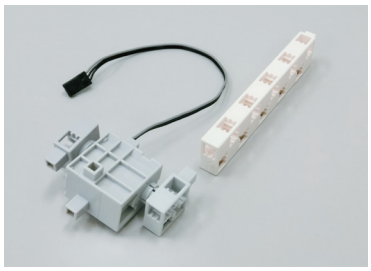


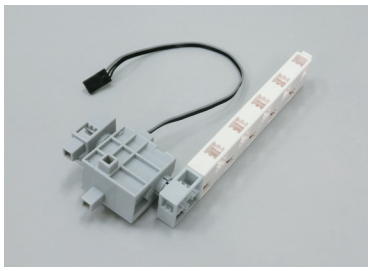


②

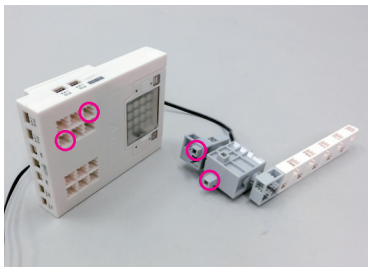


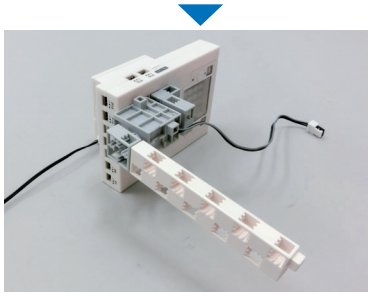
③

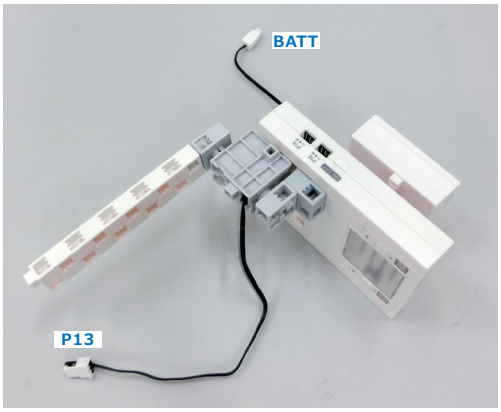


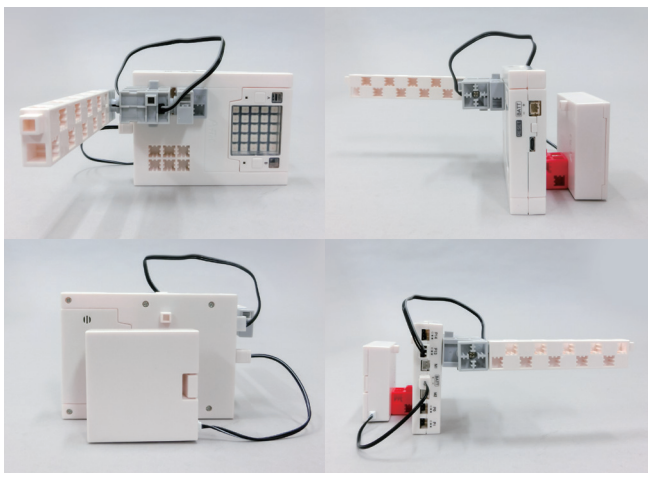


④







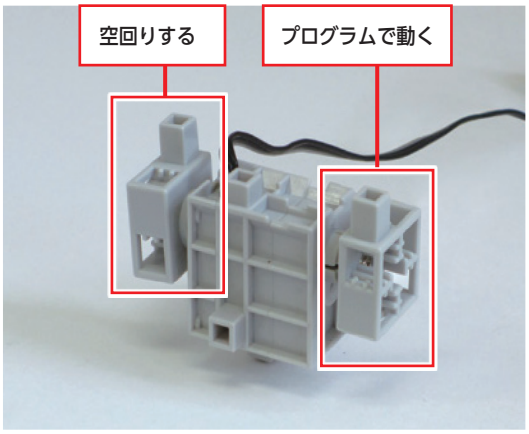


1. サーボモーターの動作確認

サーボモーターについて

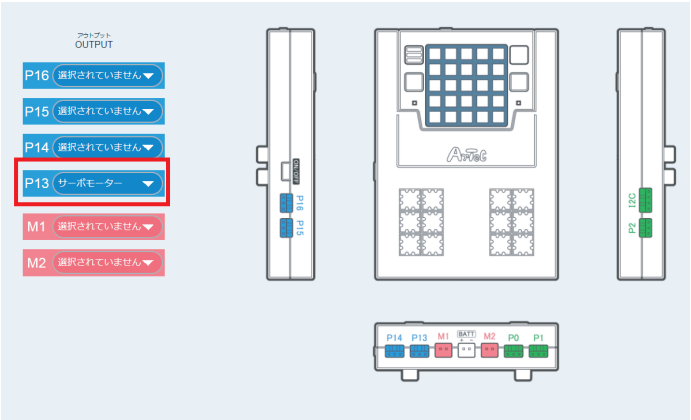
・プログラムで回る角度を決めて動かすモーターです。
・回る角度は0度から180度の間で指定します。
・手でゆっくり回したときに重たく感じる側がプログラムで動きます。

※無理やり回したり、はげしく動かしたりせずに、ゆっくりと手で回すようにしてください。



①入出力設定

「編集」より「入出力設定」を選択し、P13にサーボモーターを選択してください。



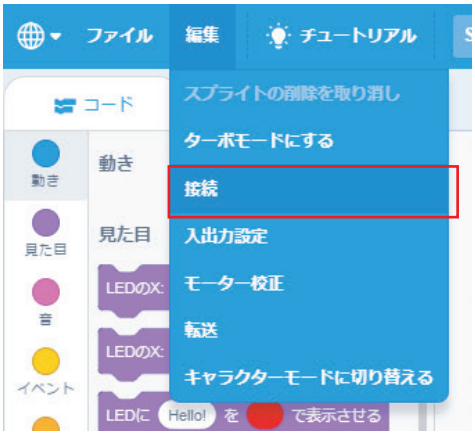
③コンピュータとの通信

Windows/Mac の場合

USBケーブルによる通信

①コンピューターとメインユニット(Studuino:bit)を USB ケーブルで接続します。

②「編集」より接続を選択します。

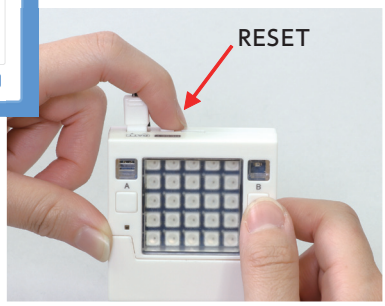


Android/iPad/Chromebookの場合

Bluetoothによる通信

①「編集」より接続を選択します。

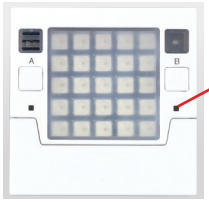
②表示されるメッセージに従い、メインユニット(Studuino:bit)のBボタンを押しながらリセットボタンを押してください。



③メインユニット(Studuino:bit)の LED に表示された点灯パターンと同じデバイスを選択してください。

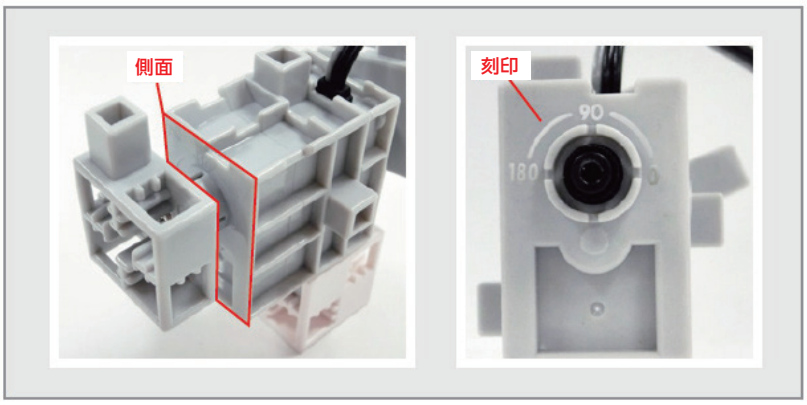
右図のようなセンサーボードが表示されると、通信が正常に行われています。

Bluetoothによる通信時は通信ランプが青に点灯します。

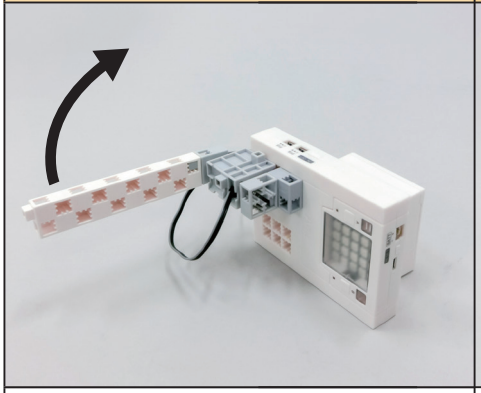
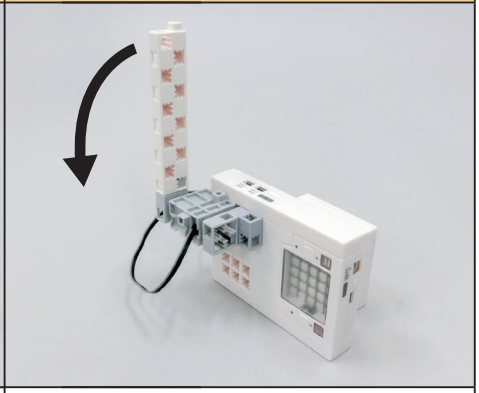


④サーボモーターの動作確認

サーボモーターを動かして、プログラムと動作の関係を調べましょう。サーボモーターの角度は、側面の刻印で確認することができます。

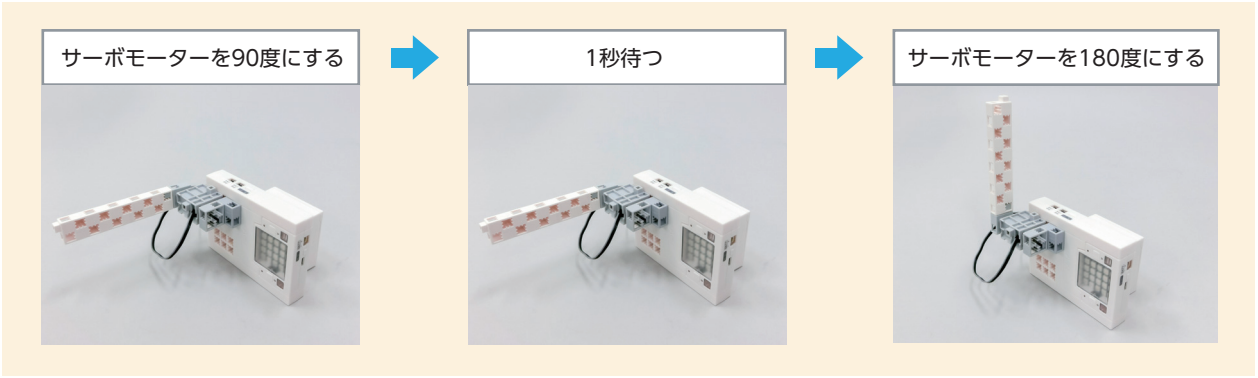


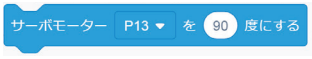
サーボモーターのブロックをスクリプトエリアに並べましょう。電源をオンにして、サーボモーターの角度で変え、ゲートの位置とサーボモーターの角度の関係を調べましょう。

ゲートが閉じているとき	ゲートが開いているとき
	
90 度	180 度

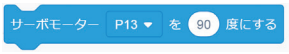
⑤プログラミング

サーボモーターが90度→180度と動くようにプログラムをつくりましょう。





のブロック２つを順番につなげ、下につなげたブロックの角度を180に変更します。

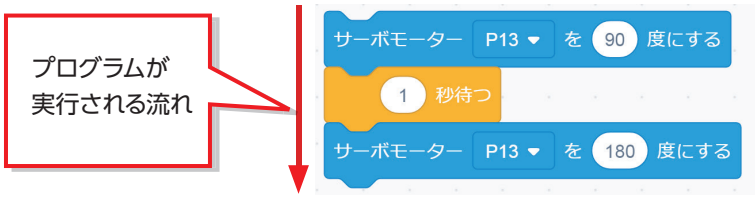
ブロックを


と


の間につなぎましょう。

つないだブロックをクリックしてプログラムを実行しましょう。

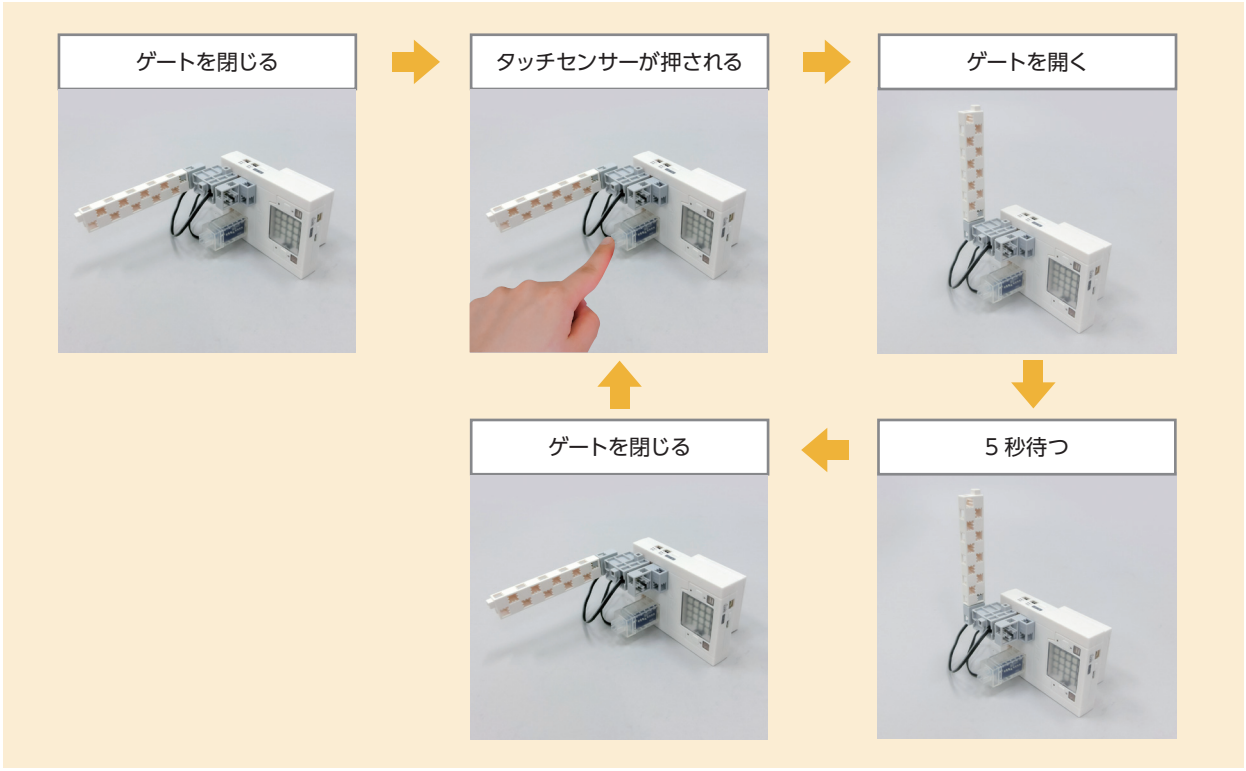
プログラムの実行中は実行されているブロックが白枠で囲われます。



このように上から順番にプログラムが処理されることを「順次処理」といいます。

2 押しボタン式ゲートの製作

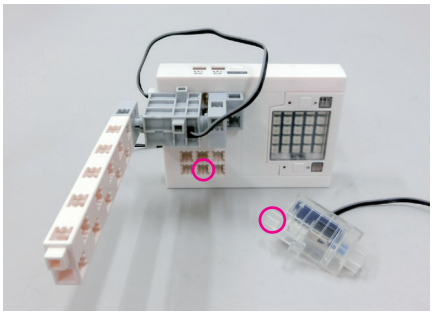
タッチセンサーを用いて、押しボタン式のゲートをつくりましょう。

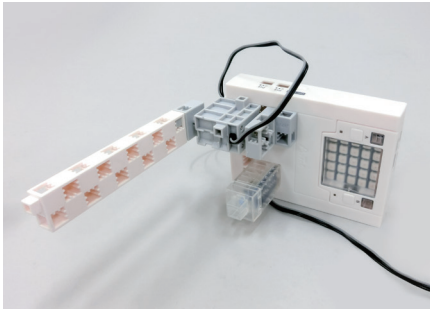


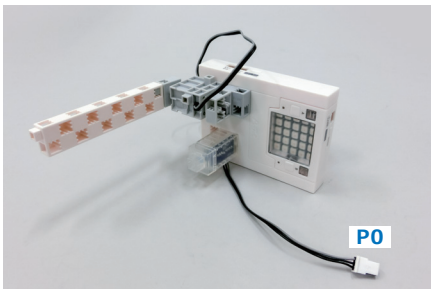
①「押しボタン式自動ゲート」の組み立て

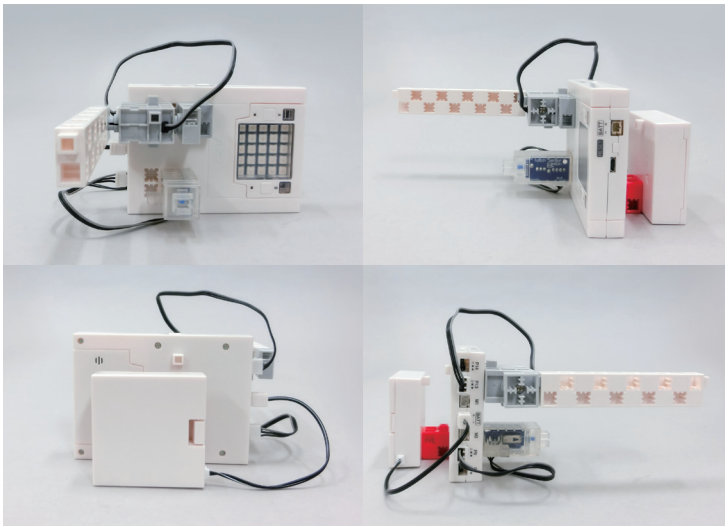
ゲートにタッチセンサーを取り付けた押しボタン式ゲートを組み立てましょう。

①



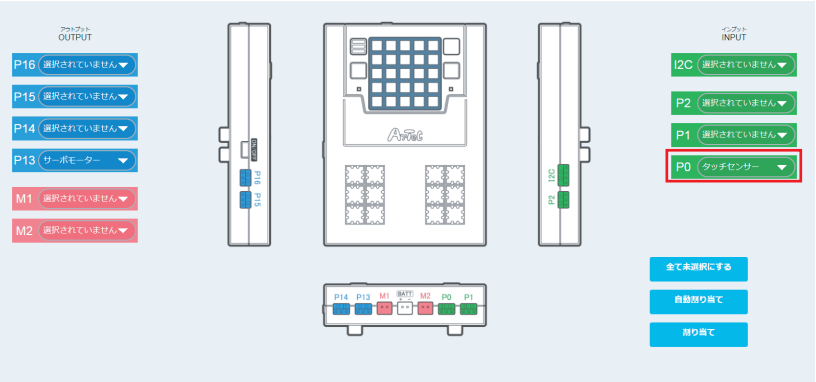






②入出力設定

P0にタッチセンサーが追加されたので、P0に「タッチセンサー」を選択しましょう。



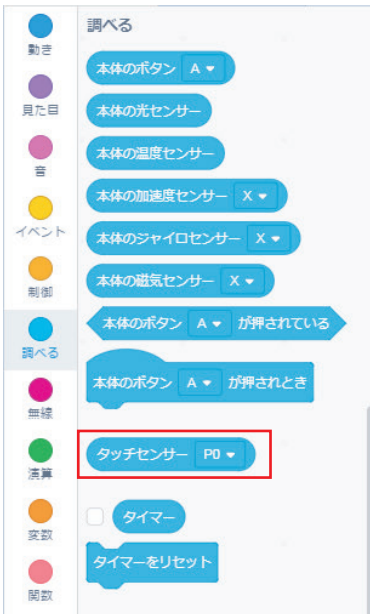
③タッチセンサーの数値確認

センサーの情報は数値で表されます。コンピューターと通信させ、タッチセンサーが押されているときと押されていないときの数値の変化を「センサーボード」で確認しましょう。「センサーボード」はコンピューターとの通信中に表示され、センサーの値がリアルタイムで確認できます。

タッチセンサーが 押されているときの数値	タッチセンサーが 押されていないときの数値
0	1

タッチセンサーの数値は「調べる」カテゴリーの

で知ることができます。

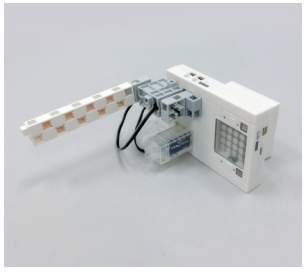
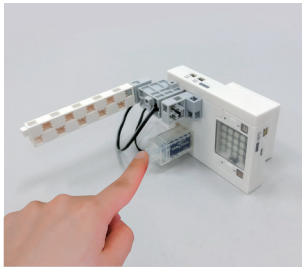
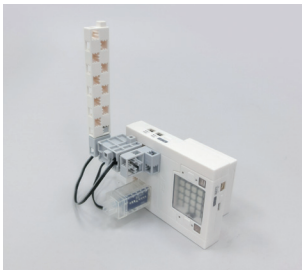
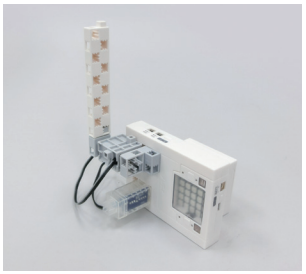
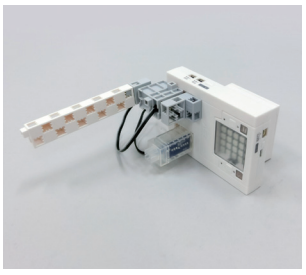


また、 

をクリックすると、右の画像のようなメッセージ形式で現在の数値を知ることができます。

④ 動作の整理

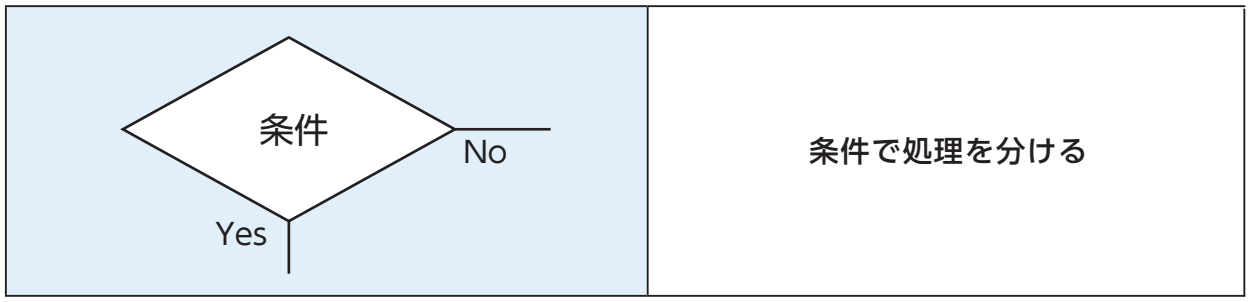
動作とプログラムの関係を整理しましょう。

順 番	動 作	プログラム
①	ゲートを閉じる 	サーボモーターP13を 90度にする
②	タッチセンサーが押される 	タッチセンサーP0の値 ＝ 0
③	ゲートを開く 	サーボモーターP13を 180度にする
④	5秒待つ 	5秒待つ
⑤	ゲートを閉じる 	サーボモーターP13を 90度にする

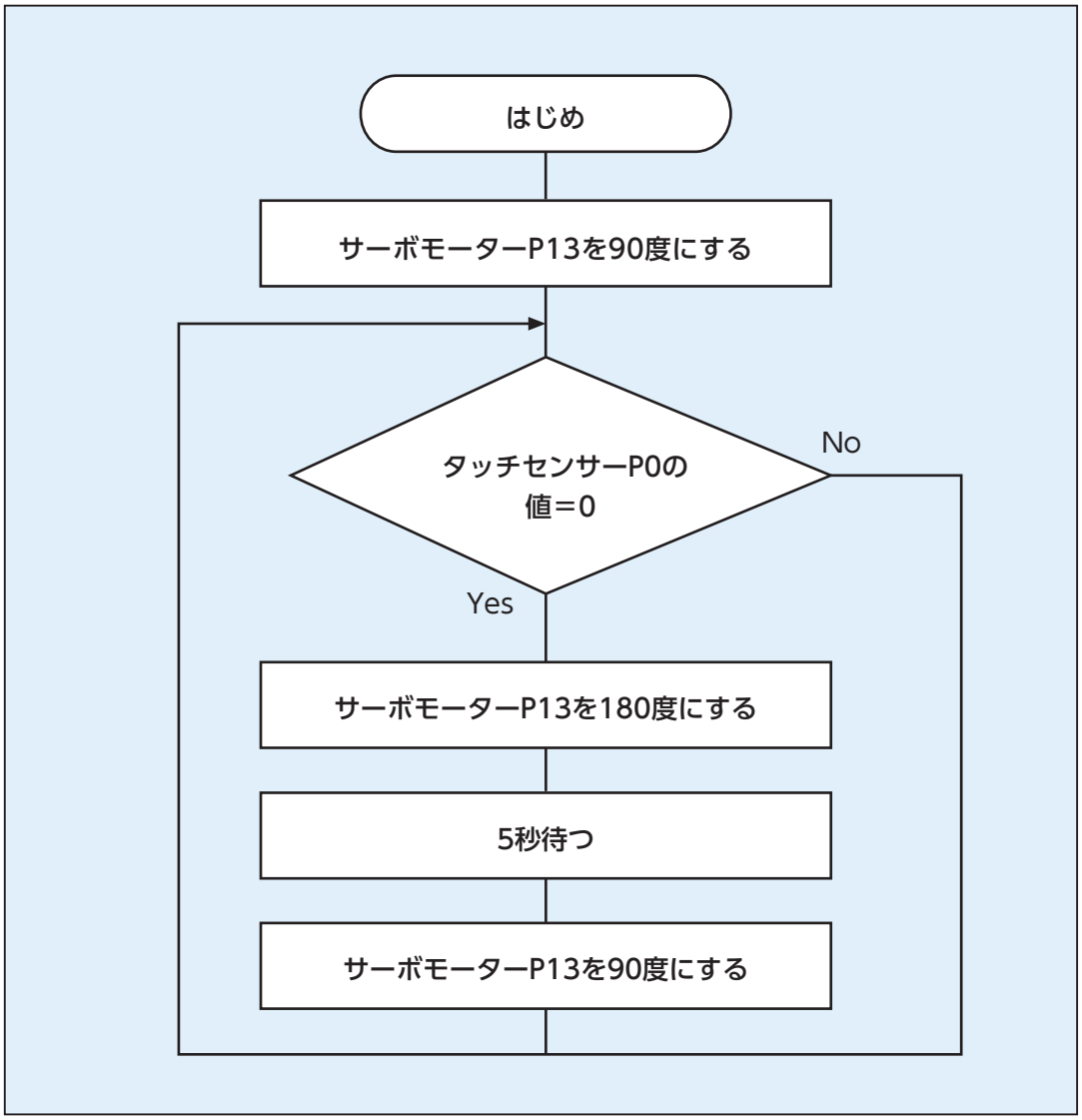
②～⑤を
ずっと繰り返す

⑤フローチャート

フローチャートで条件によって処理を分けることを表すときは以下の記号を使います。

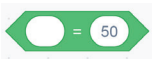


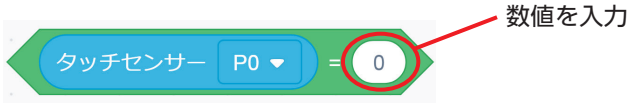
72ページで整理した動作を参考に、フローチャートをまとめましょう。



フローチャートに「終了」が存在せずに繰り返しているのは、タッチセンサーが押されているかを常に確認するためです。

⑥プログラム作成

条件は  と  を組み合わせてつくることができます。



作成した条件は右図の各種制御ブロックの空欄に入れて使うことができます。

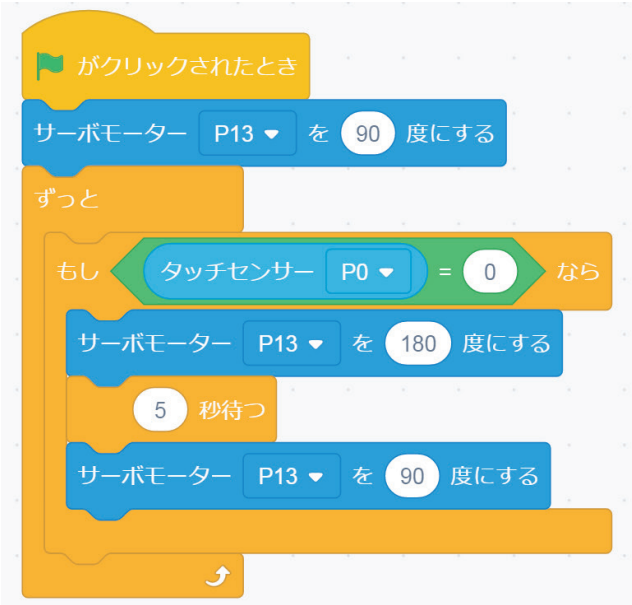
…… 条件が成り立つときだけ、囲まれたプログラムが処理されます。

…… 条件が成り立つときだけ、上段で囲まれたプログラムが処理され、
条件が成り立たないときは、下段で囲まれたプログラムが処理されます。

…… 条件が成り立つまで、次に続くプログラムを処理せずに待ちます。

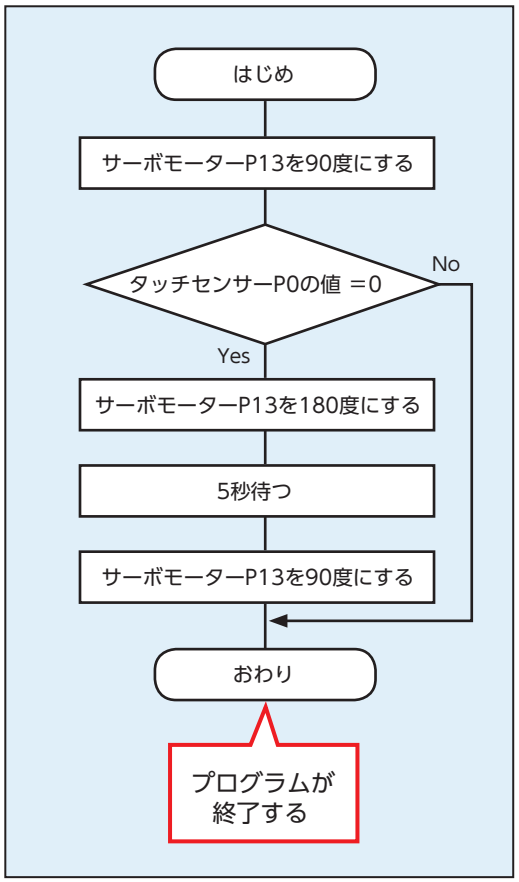
…… 条件が成り立つまで、囲まれたプログラムを繰り返し処理します。

押しボタンゲートのプログラム例

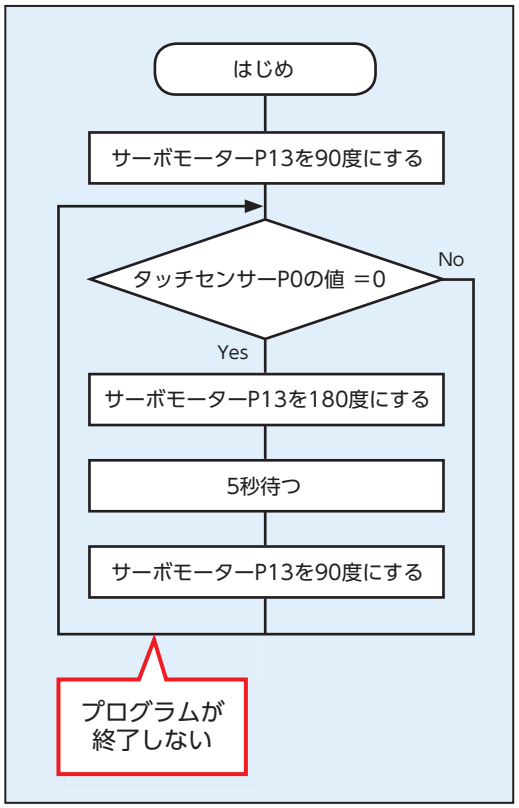


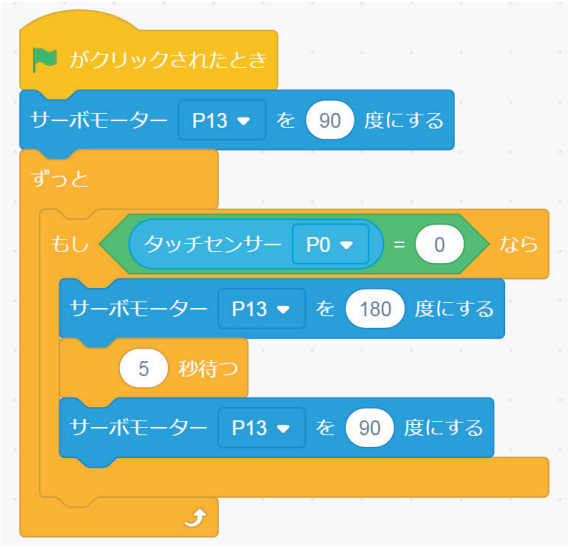
「ずっと」のブロックを入れないとどうなるのか？

「ずっと」を入れない場合のフローチャートとプログラムは以下のようになります。



このプログラムを動作させると、タッチセンサーを押しても全く反応しなくなります。これは、プログラムが非常に高速で処理されることで、プログラムを動作させた瞬間にプログラムが終了してしまうからです。「ずっと」のブロックを入れることで、タッチセンサーの値を常に確認するようになるため想定通りの動作をするようになります。

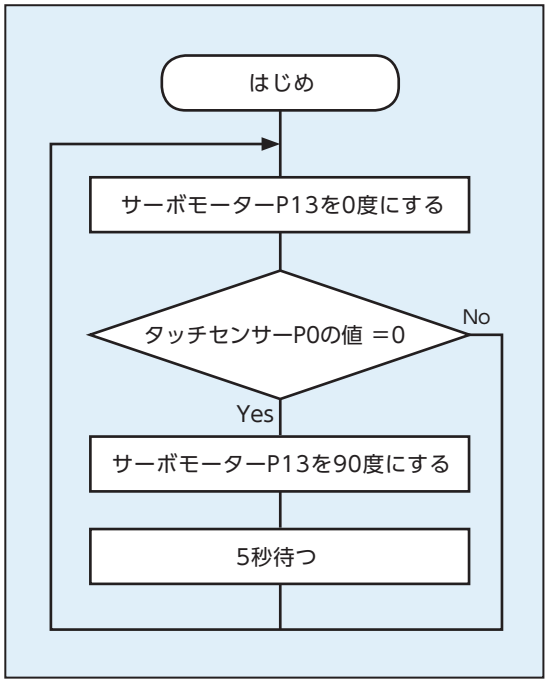


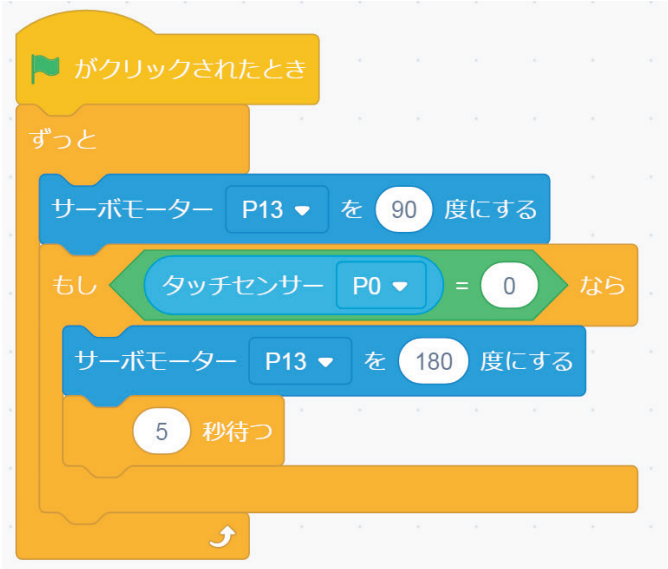


コンピューターはプログラム通りに動くので、思わぬ間違いで期待した動作と実際の動作が異なってしまうことがよくあります。フローチャートをまとめる段階で動作をよく考えることは間違いを避けることにつながります。

押しボタン式ゲートの別解

以下のようなプログラムでも押しボタン式ゲートの動作ができます。

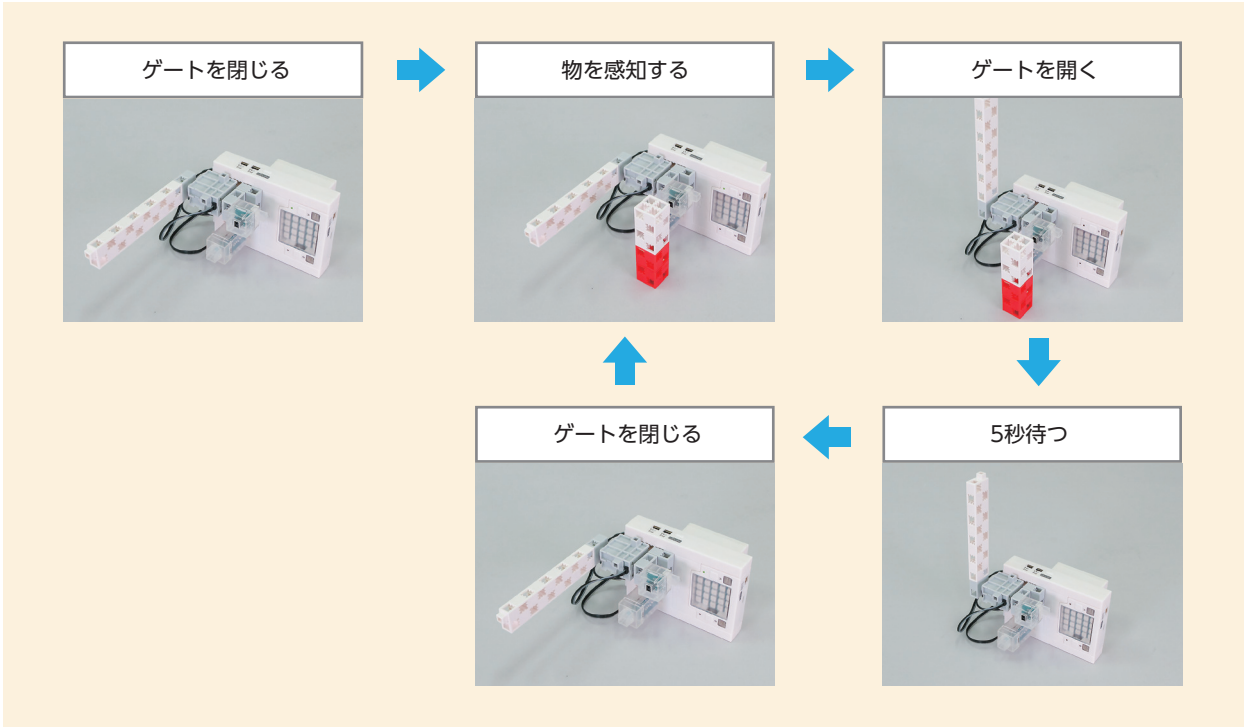




プログラムのつくり方は一つとは限りません。想定する動作が実現できればどのようなプログラムでも構いません。自分なりに考えてプログラムをつくることが大切です。

1．自動ゲートの製作

赤外線フォトリフレクタを利用して自動ゲートをつくりましょう。

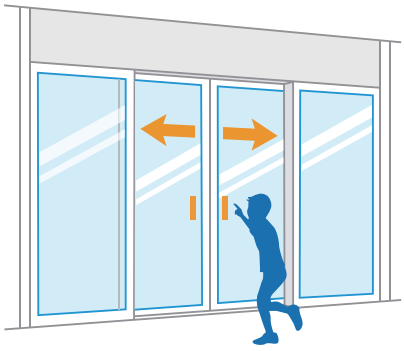


基本的な動作の流れは以前につくった押しボタン式ゲートと同じですが、タッチセンサーの代わりに赤外線フォトリフレクタを使用します。

自動ドアの種類

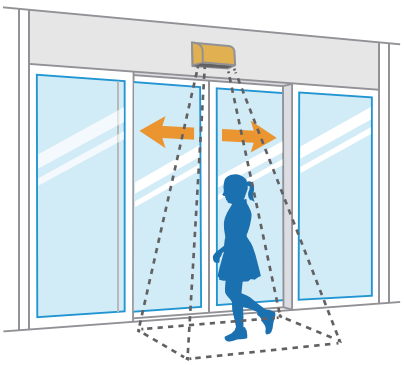
身近なところで使われている自動ドアのセンサーには様々な方式があり、代表的なものに、タッチ方式と光反射方式があります。タッチ方式はタッチセンサーで実現でき、光反射方式は赤外線フォトリフレクタで実現することができます。

タッチ方式



ドアに取り付けられた
タッチスイッチを押すと開く

光反射方式

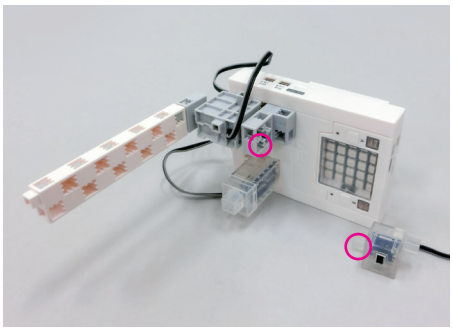


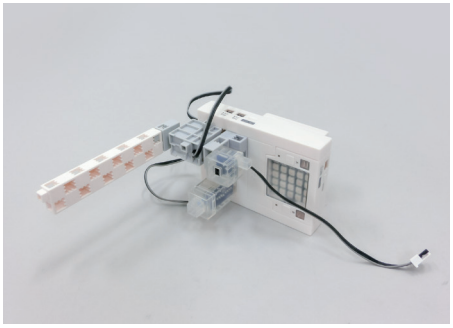
ドアの上に取り付けられたセンサーから出る
光を反射するものがあると開く

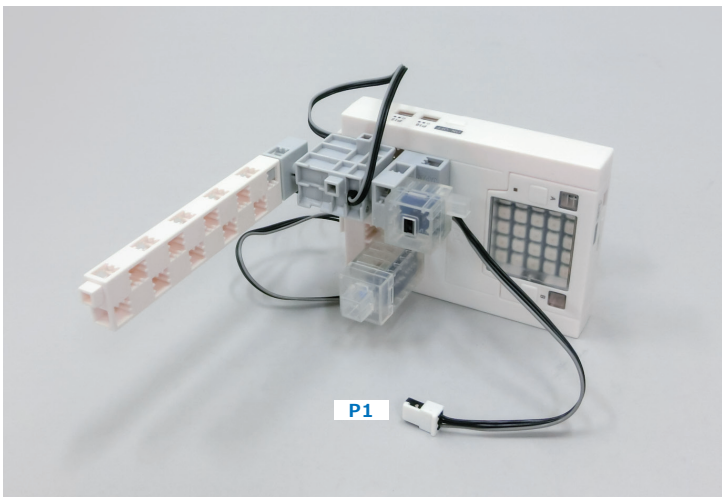
①「光反射方式自動ゲート」の組み立て

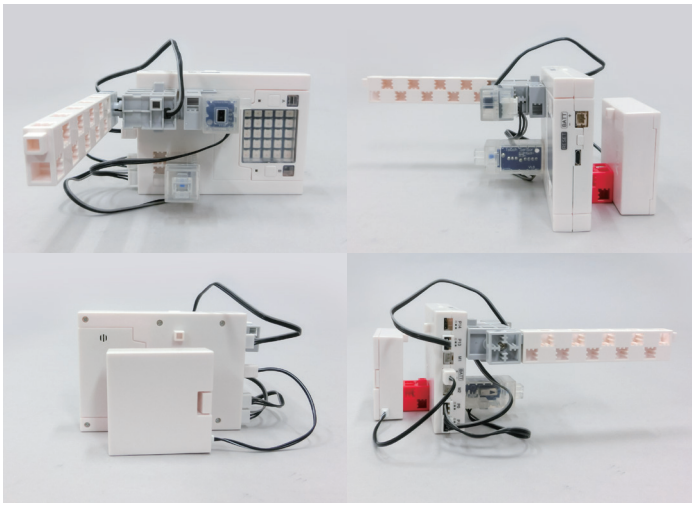
ゲートに赤外線フォトリフレクタを取り付けた自動ゲートを組み立てましょう。

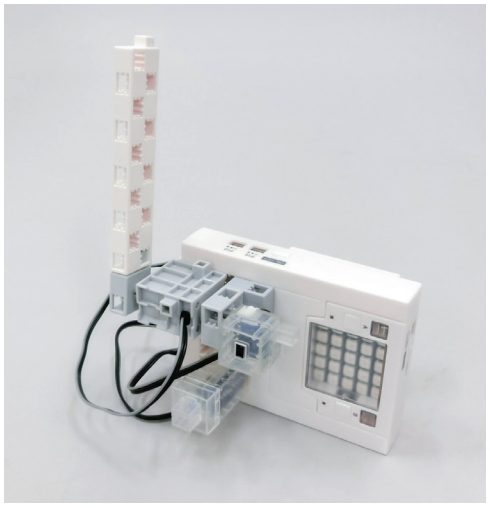
①





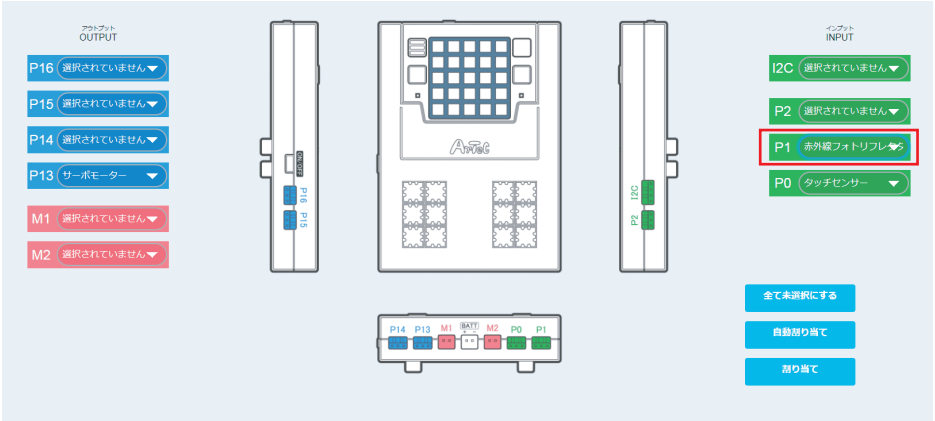






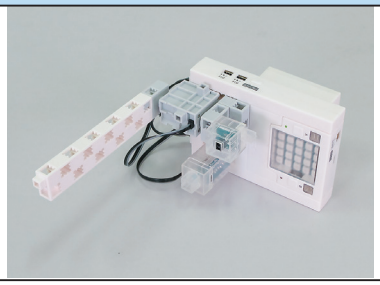
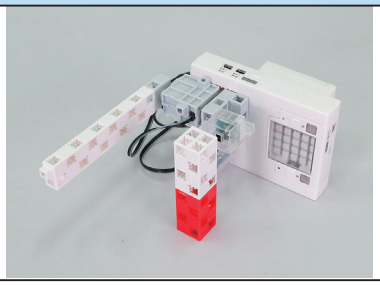
②入出力設定

P1 に赤外線フォトリフレクタが追加されたので、P1 に「赤外線フォトリフレクタ」を選択しましょう。



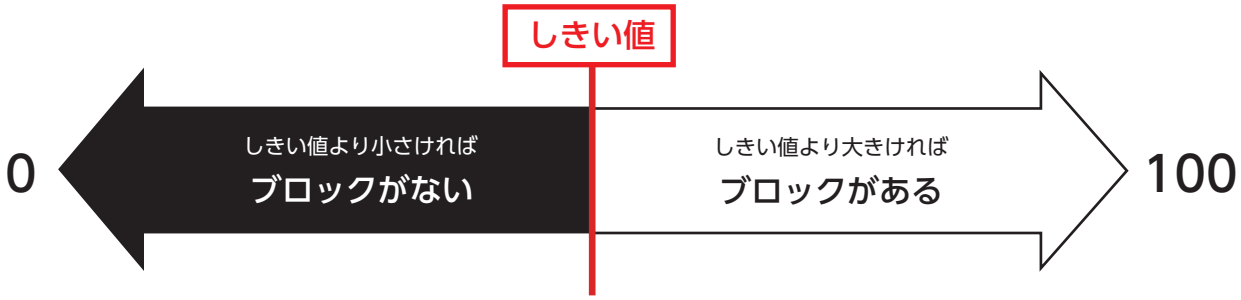
③赤外線フォトリフレクタの数値確認

自動ゲートはゲートの前に人が立ったときにセンサーで感知してゲートを開きます。
この「感知」の役割を赤外線フォトリフレクタで行います。
コンピュータと接続し、通信中に表示される「センサーボード」で赤外線フォトリフレクタの値を確認しましょう。
下の図のように赤外線フォトリフレクタの前にブロックのあるときとないときで数値の変化を確認しましょう。
※赤外線フォトリフレクタの値は太陽光の影響を受けるため、日光のあたる窓に向けないようにしてください。

白いブロックを 置いていないときの数値	白いブロックを 置いているときの数値
	

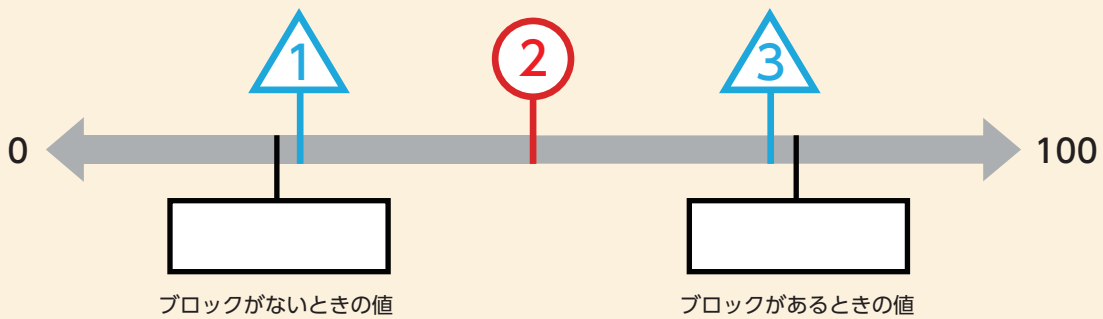
④しきい値の決定

赤外線フォトリフレクタの値で白いブロックがないときとあるときを区別する場合は、２つの状態の境目となる値を決めて判断します。このような境目となる値のことを「しきい値」と言います。



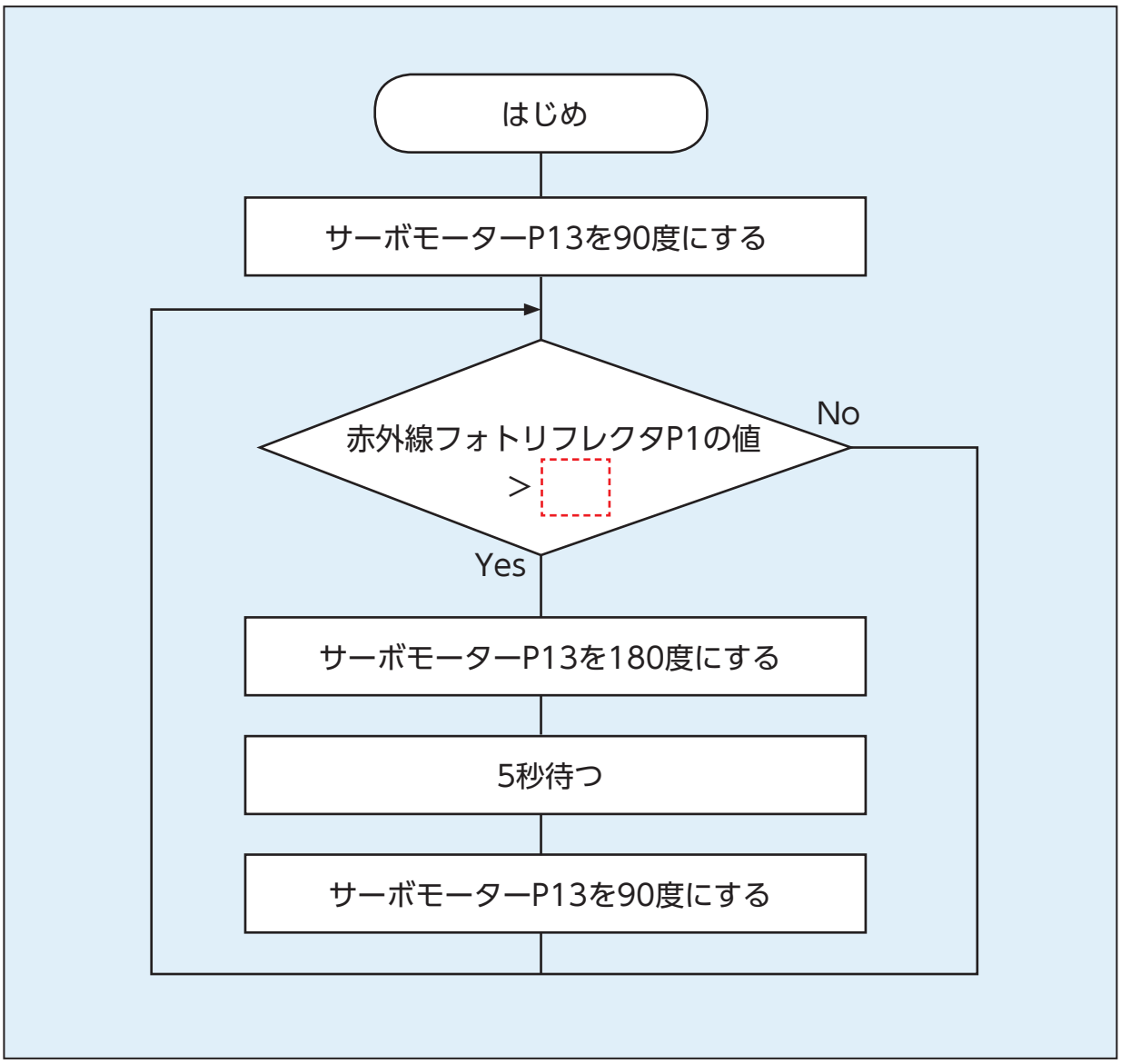
しきい値の決定

しきい値を 1 や 3 のように調べたどちらかの値の近くに決めてしまうと、赤外線フォトリフレクタの値がそこから少し変わるだけで、ブロックの有無を区別してしまいます。センサーの値は周りの環境によって変わりやすいため誤って判断しないように、2 のように調べた２つの値の中央あたりをしきい値として決めましょう。



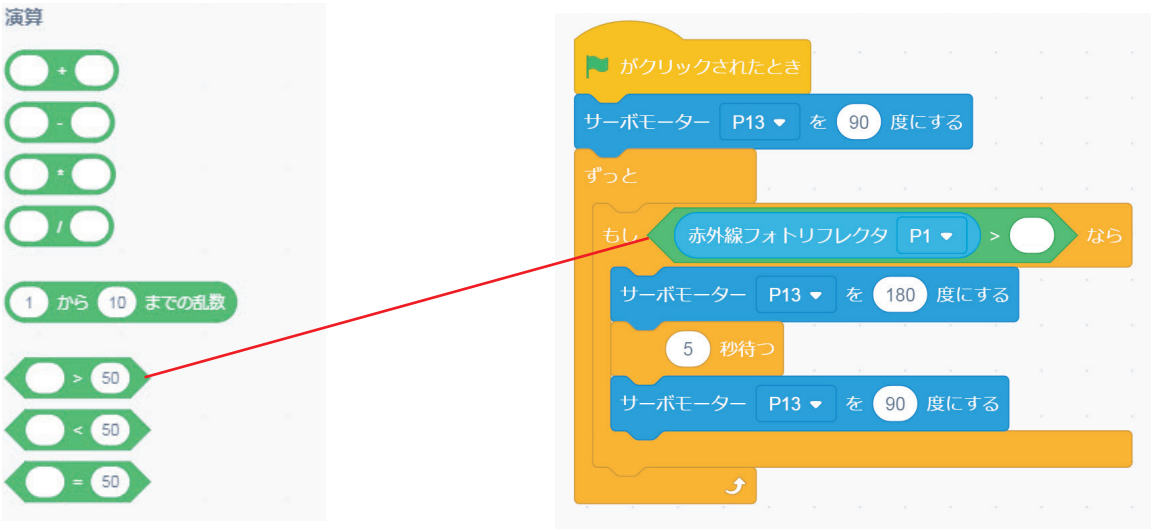
⑤フローチャート

76ページに示した動作を実現できるように手順を考えてフローチャートをまとめましょう。
基本的なフローチャートの構造は押しボタン式ゲートでつくったものと同じで、センサーをタッチセンサーから赤外線フォトリフレクタに変更しただけです。条件の部分を決めたしきい値を利用して書き換えましょう。



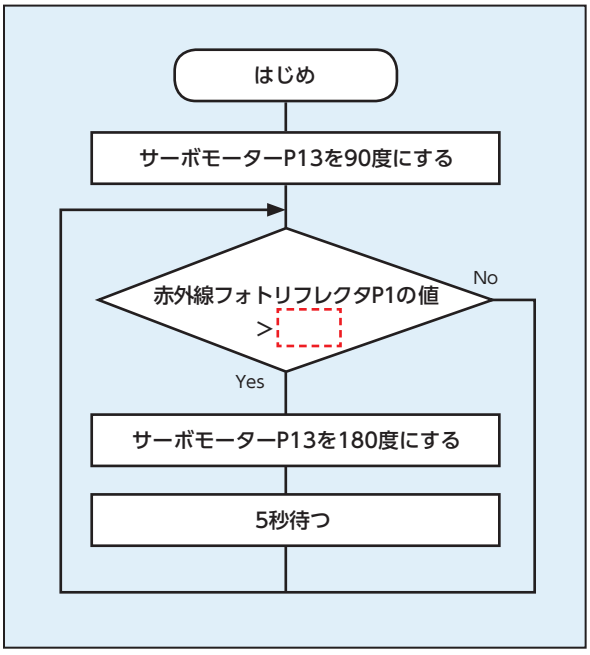
⑥プログラム作成

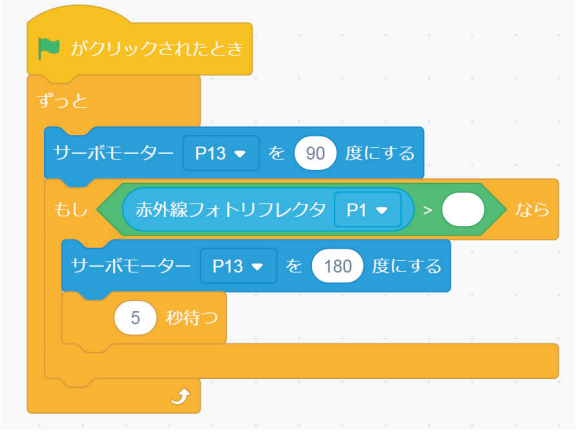
不等号のブロックを使うことに注意して、プログラムをつくりましょう。



自動ゲートの別解

以下のプログラムでも自動ゲートの動作が実現できます。





発展学習

ここでは、複数の条件による処理の分け方と、状態を保存する「変数」について学習します。
作成した自動ゲートの問題点を見つけ、どのようにしたら解決できるか考えましょう。

問題点の例

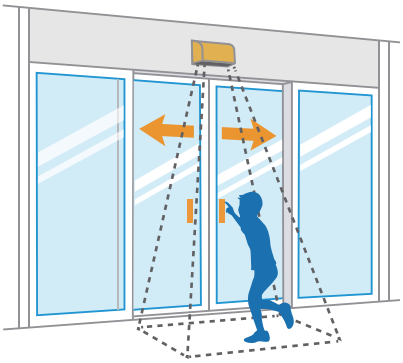
・人以外の物体や太陽光などにより赤外線フォトリフレクタが感知され、
人がいないのにゲートが開くといった誤作動が起こる。

・ゲートが閉まるときに人が挟まる危険性がある。

解決策

ここでは、実際の自動ドアのしくみを元に問題解決のプログラム例を示します。

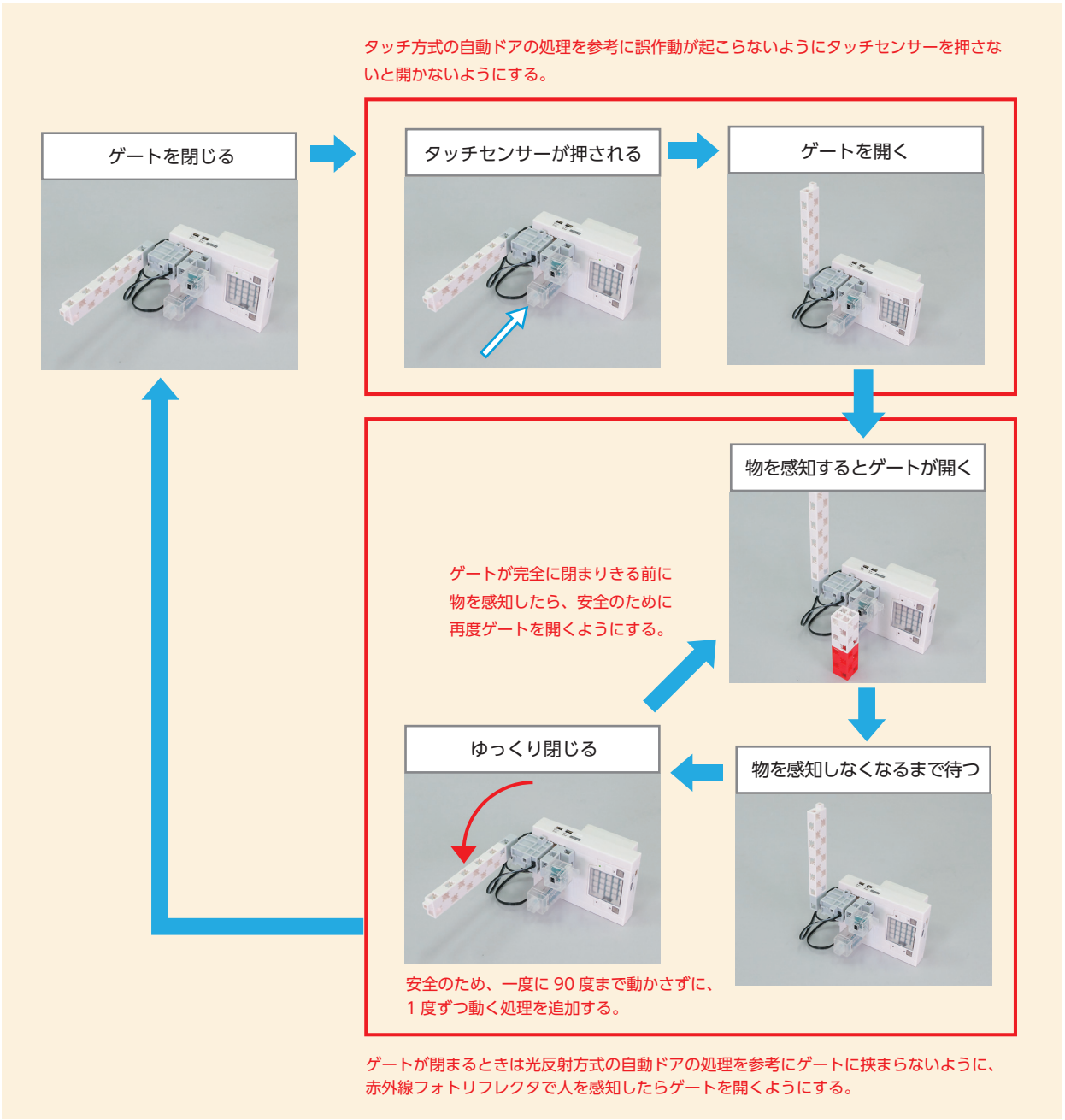
タッチ方式/光反射方式



[タッチ方式/光反射方式併用タイプの解決方法]

誤作動防止：人がタッチスイッチを押さないと開かない。

安 全 対 策 ： センサーで人の有無を感知し、人がいる間はドアが閉まらない。
ドアが閉まる途中でも人を感知すると自動で開く。



上記のように、ゲートの状態で処理を分ける必要があります。

ゲートが閉まっているとき（サーボモーターの角度が 90 度のとき）　→　物を感知しない。
ゲートが開いているとき（サーボモーターの角度が 90 度より大きいとき）　→　物を感知する。

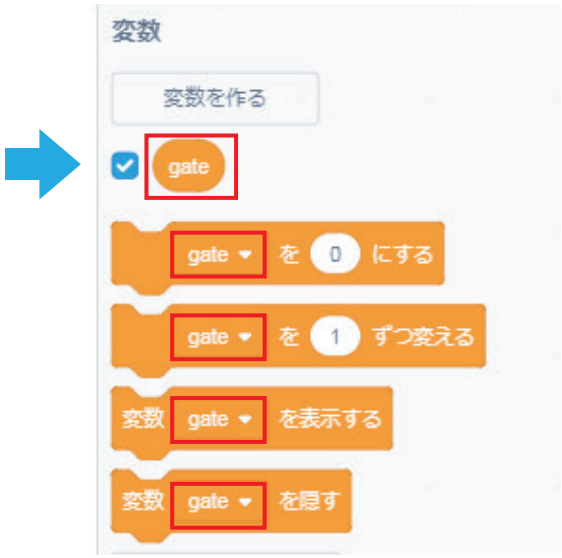
ゲートの状態（サーボモーターの角度）によって処理を分ける場合は、
「変数」をつかって状態を記録させる必要があります。

① 変数の作成

変数カテゴリの「変数を作る」をクリックして表示されたウインドウで新しい変数名を指定して作成します。

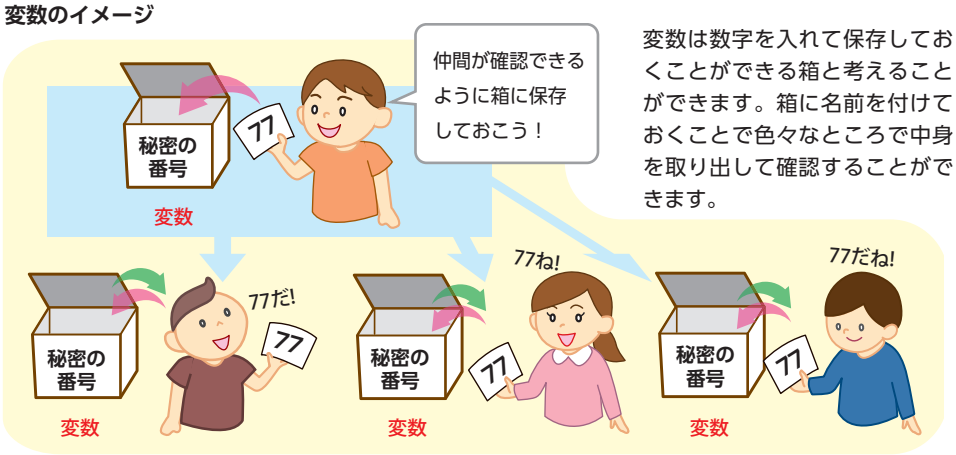
指定できる変数名は半角英数字です。

作成した名前の変数が使用できるようになります。



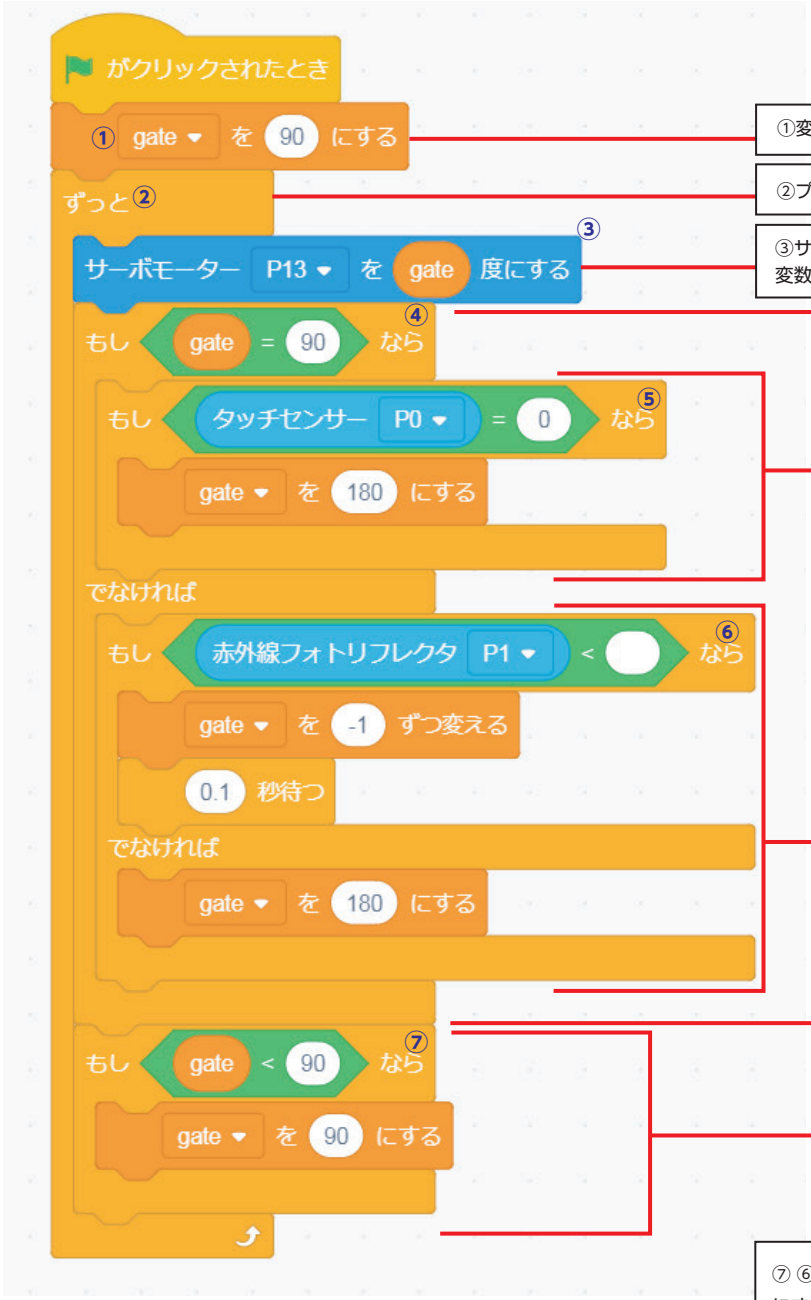
【変数とは…？】

変数のイメージ



変数は数字を入れて保存しておくことができる箱と考えることができます。箱に名前を付けておくことで色々なところで中身を取り出して確認することができます。

サンプルプログラム



①変数の初期値を 90 にします。

②プログラム全体をずっとで囲みます。

③サーボモーターの角度の指定に変数をいれることで、
変数で指定された値にサーボモーターの角度が変化します。

⑤ゲートが閉じているときの処理です。
タッチセンサーが押されると変数を
180 にすることで、ゲートが開きます。

④変数の値が 90 のときと、
90 でないときで条件を分けることで、ゲートが閉じているか開いているかの状態で処理を分けることができます。

⑥ゲートが開いているときの処理です。
赤外線フォトリフレクタで物の有無を感知して処理を分けます。
感知していないとき（物がないとき）は、
変数を -1 ずつ変えることで、ゆっくりゲートが閉じられます。
0.1 秒待つ の値を大きくするほどゆっくり閉じられます。
ゲートが完全に閉じる前に物を感知したときは、変数が 180 になり再びゲートが開きます。

⑦ ⑥の処理で変数が 90 より小さくならないようにする処理です。
変数が 90 より小さくなったら 90 にします。

応用学習　テーマ2

自動搬送システム

＜学習内容＞

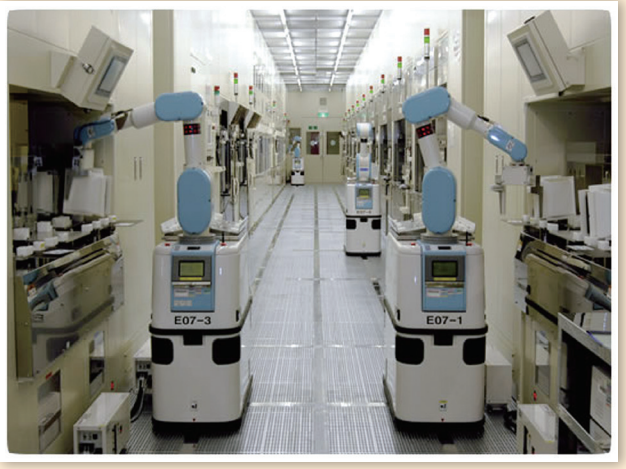
ロボットカーにサーボモーターをとりつけ、所定の位置までブロックを搬送する自動搬送システムを構想する。

※基本学習テーマ2と応用学習テーマ1を学習の上実施してください。

無人搬送ロボット

工場でつかわれているロボットは部品や商品を製造したり、検査するものだけではありません。製造につかわれる材料や製造された後の部品や商品を運ぶ「無人搬送ロボット」も活躍しています。
無人搬送ロボットは床に引かれた線を読み取って走行するものや、レールに沿って移動するものなど様々な種類があります。

ラピスセミコンダクタ宮城株式会社　工程内自動搬送システム



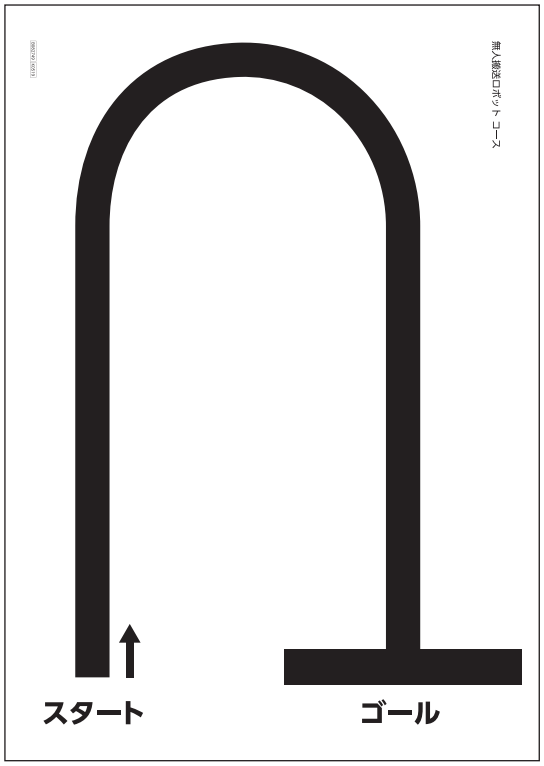
http://www.lapis-semi.com/miyagi/info/factory/tecno/auto.html

無人搬送ロボットをつかうメリット

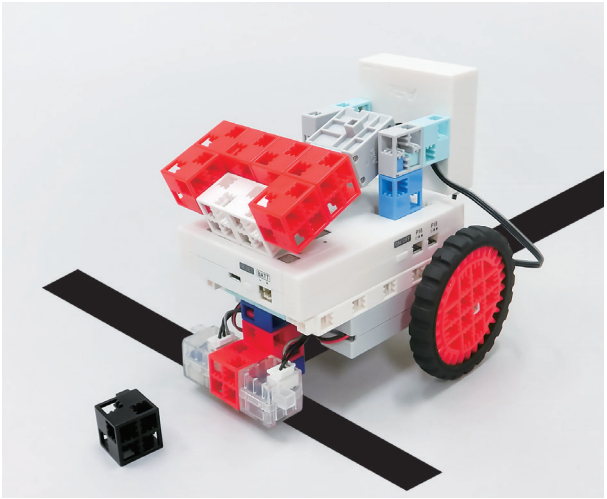
無人搬送ロボットを利用することで、作業の手間が省けるだけでなく、精密な電子部品を製造する工場では人を減らすことで、よりクリーンな環境を保ちやすくなります。

課 題

スタートラインからスタートしゴールの位置で停止してブロックを枠に落とす無人搬送ロボットを製作しましょう。



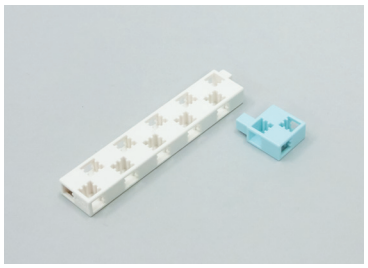
●仕様するパーツ
DCモーター×２…… ロボットカーの制御
赤外線フォトリフレクタ×２……ラインの検知
サーボモーター……搬送用アームの駆動

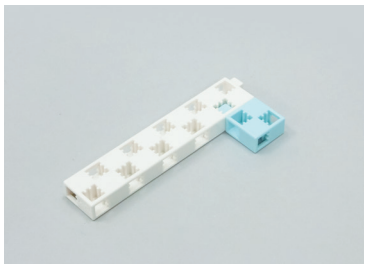


作例はあくまで参考です。
サーボモーターの位置やセンサーの位置、
プログラムも自由に考えて課題をクリアしましょう。

①組み立て

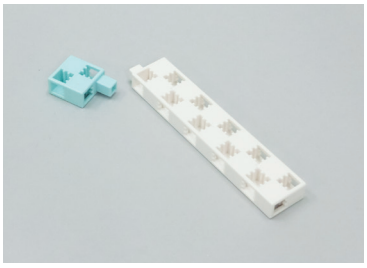
①

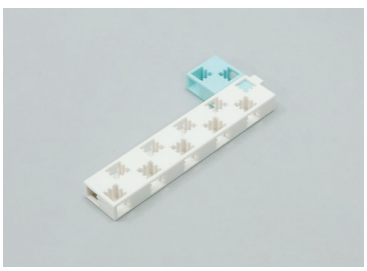




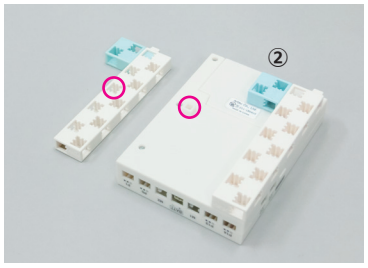
②

③

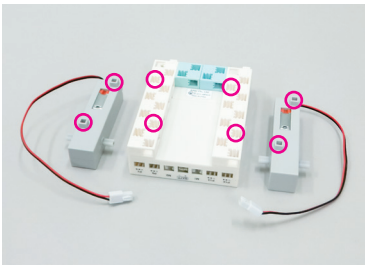


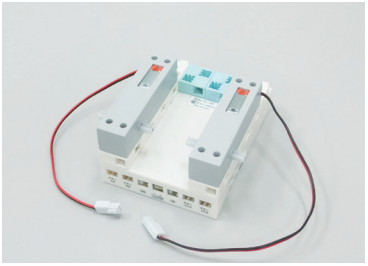


④

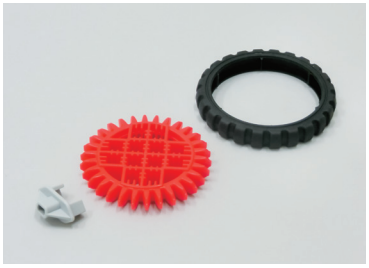


⑤

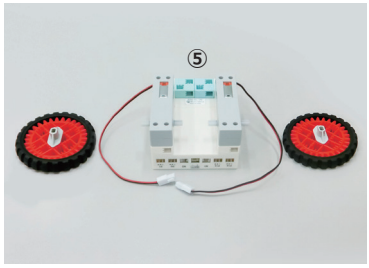


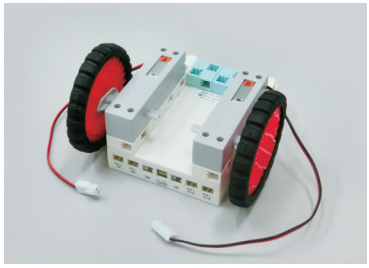


⑥ ×2

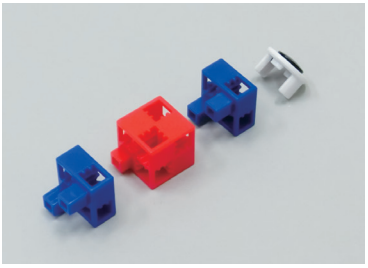


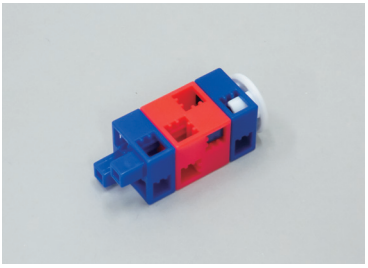
⑦



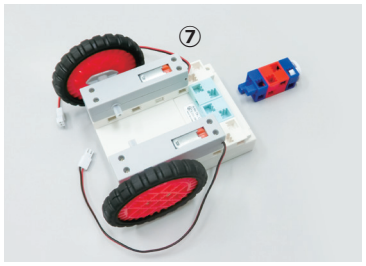


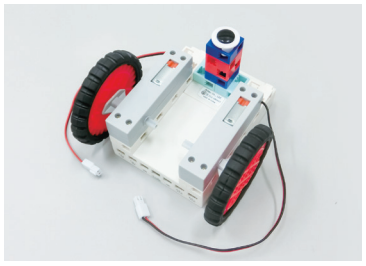
⑧



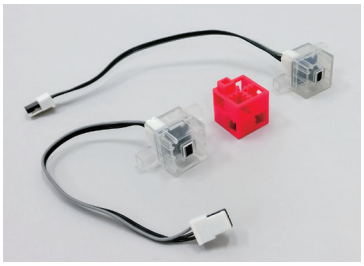


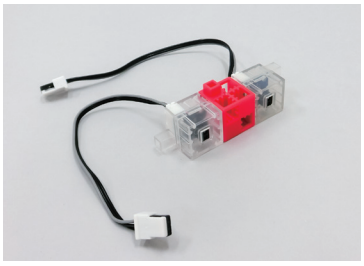
⑨



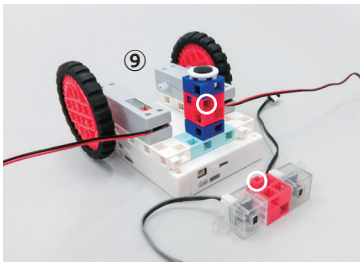


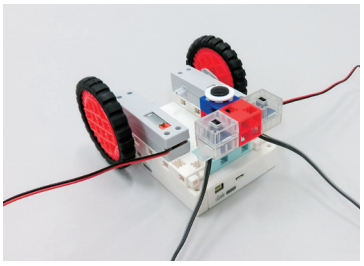
⑩



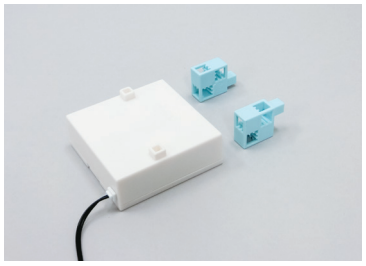


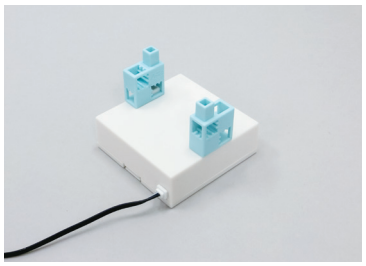
⑪



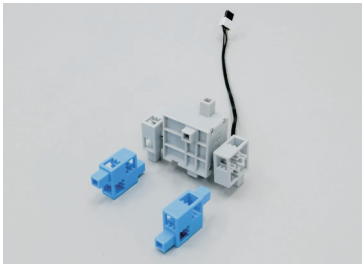


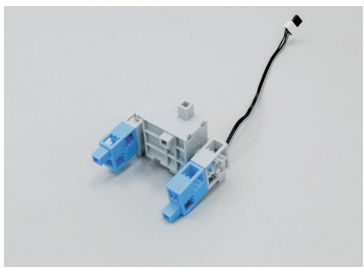
⑫



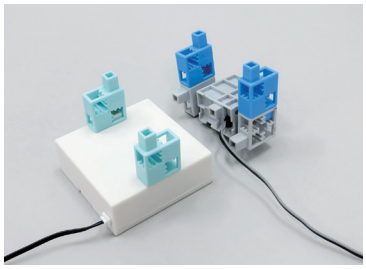


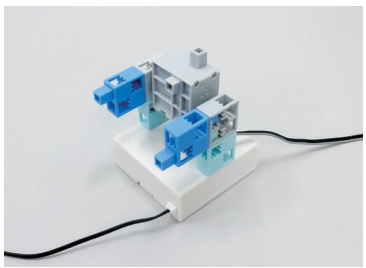
⑬



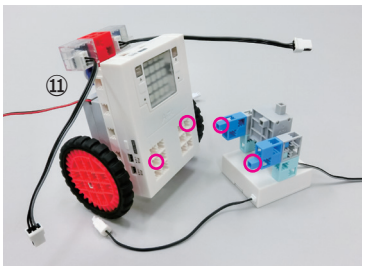


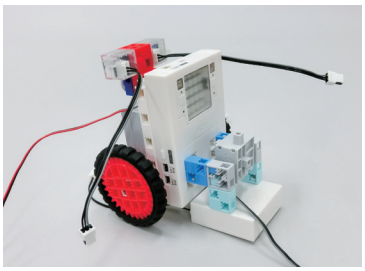
⑭



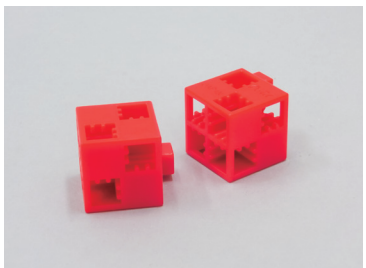


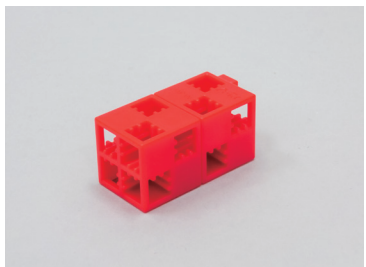
⑮



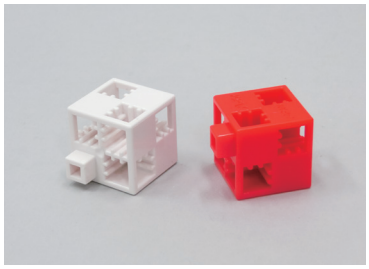


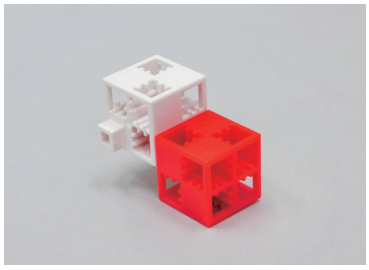
⑯



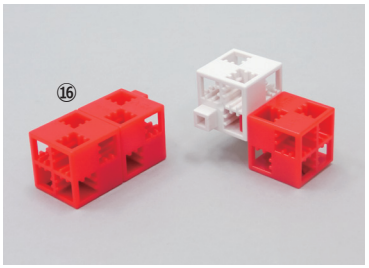


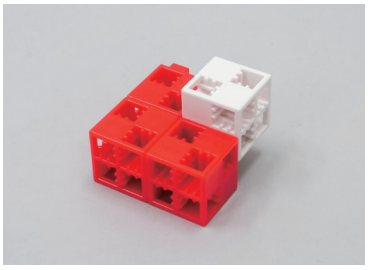
⑰



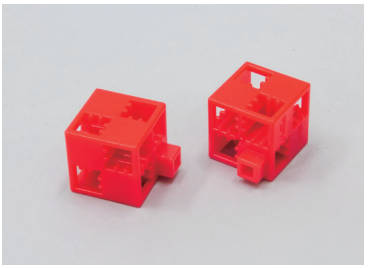


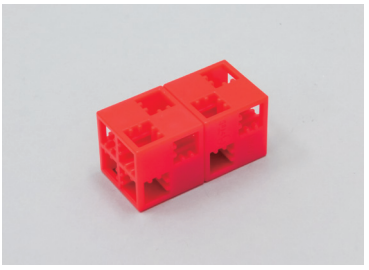
⑱



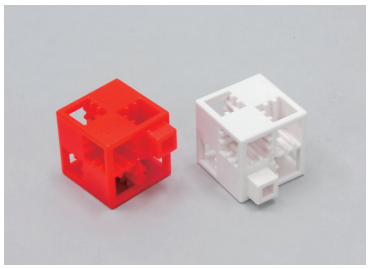


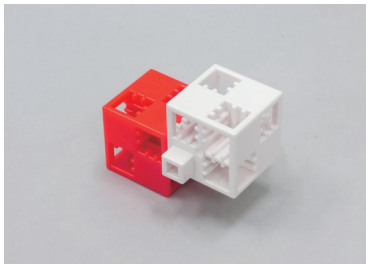
⑲



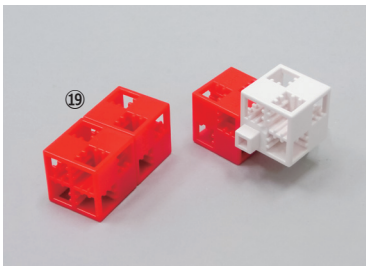


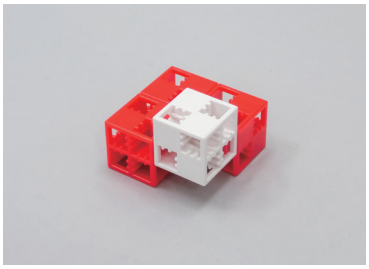
⑳



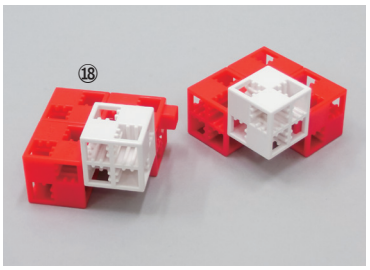


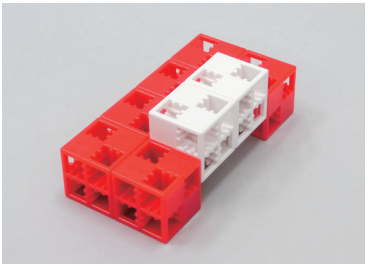
㉑



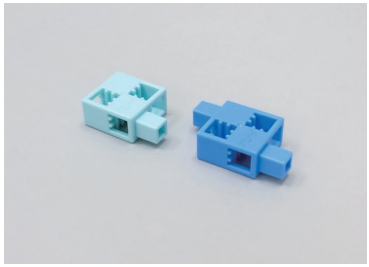


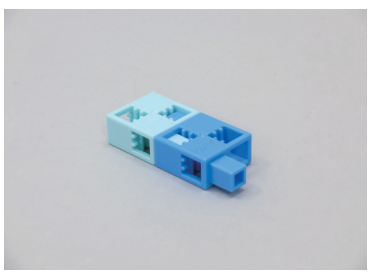
㉒



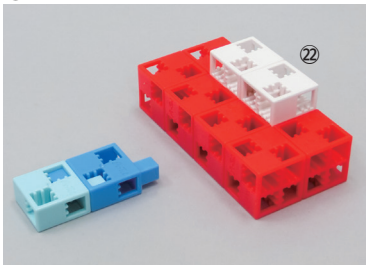


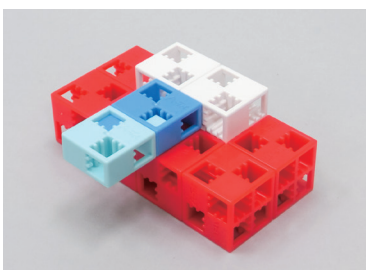
㉓



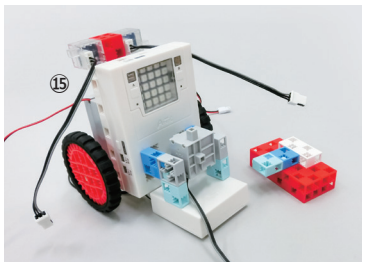


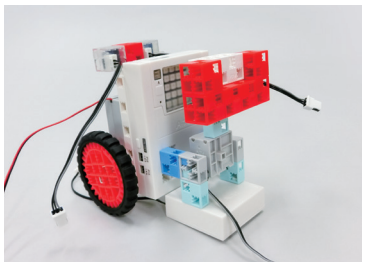
㉔

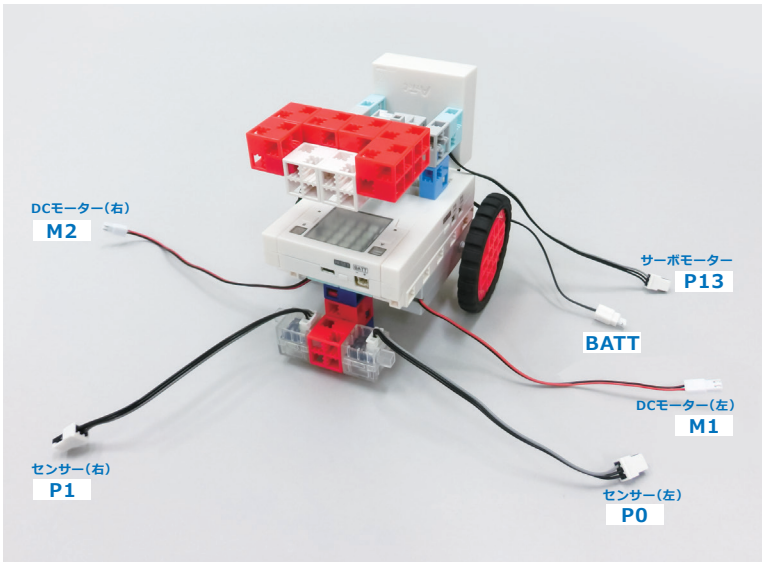


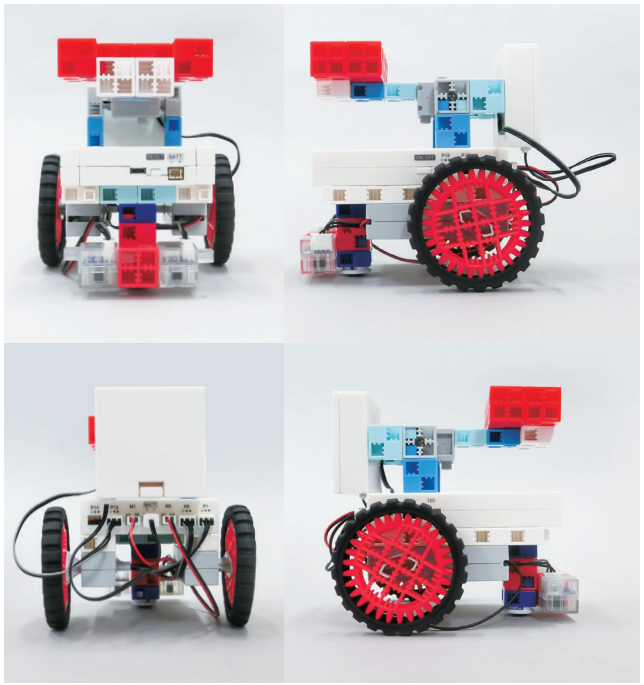


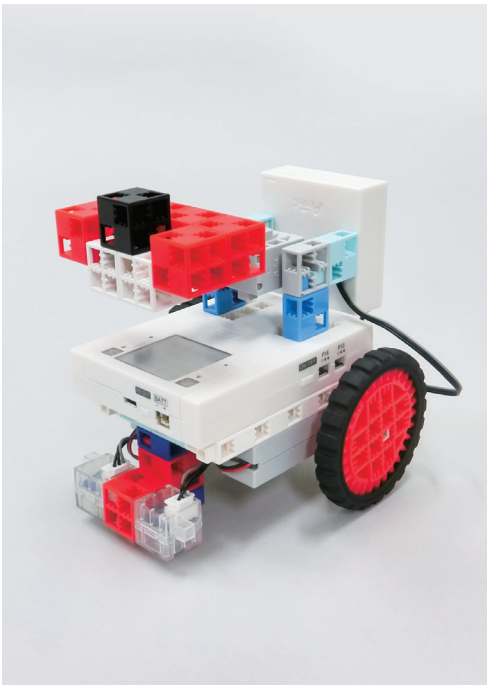
㉕





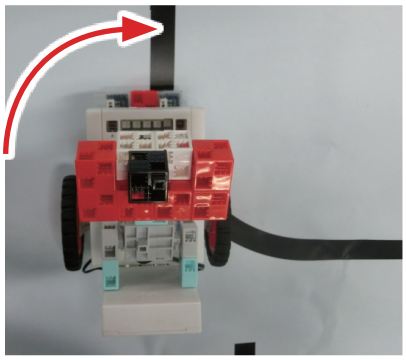
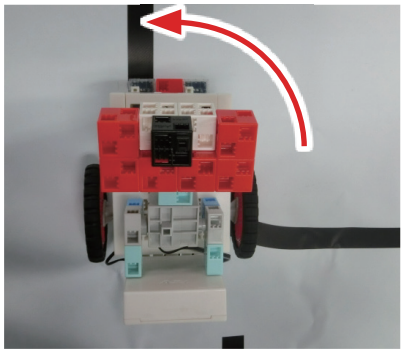


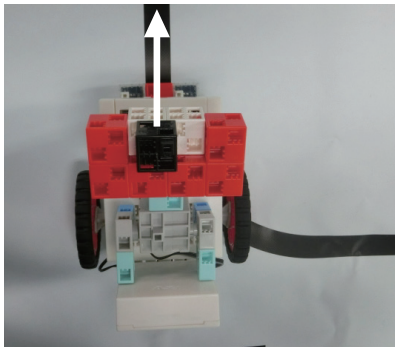
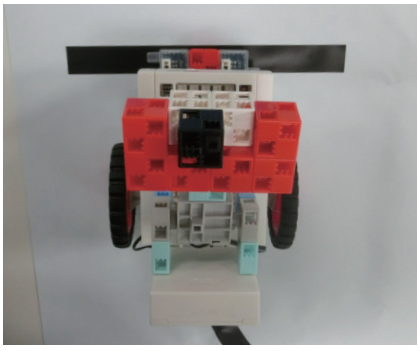




② 動作の整理

赤外線フォトリフレクタの条件とDCモーターの動作の関係を整理しましょう。

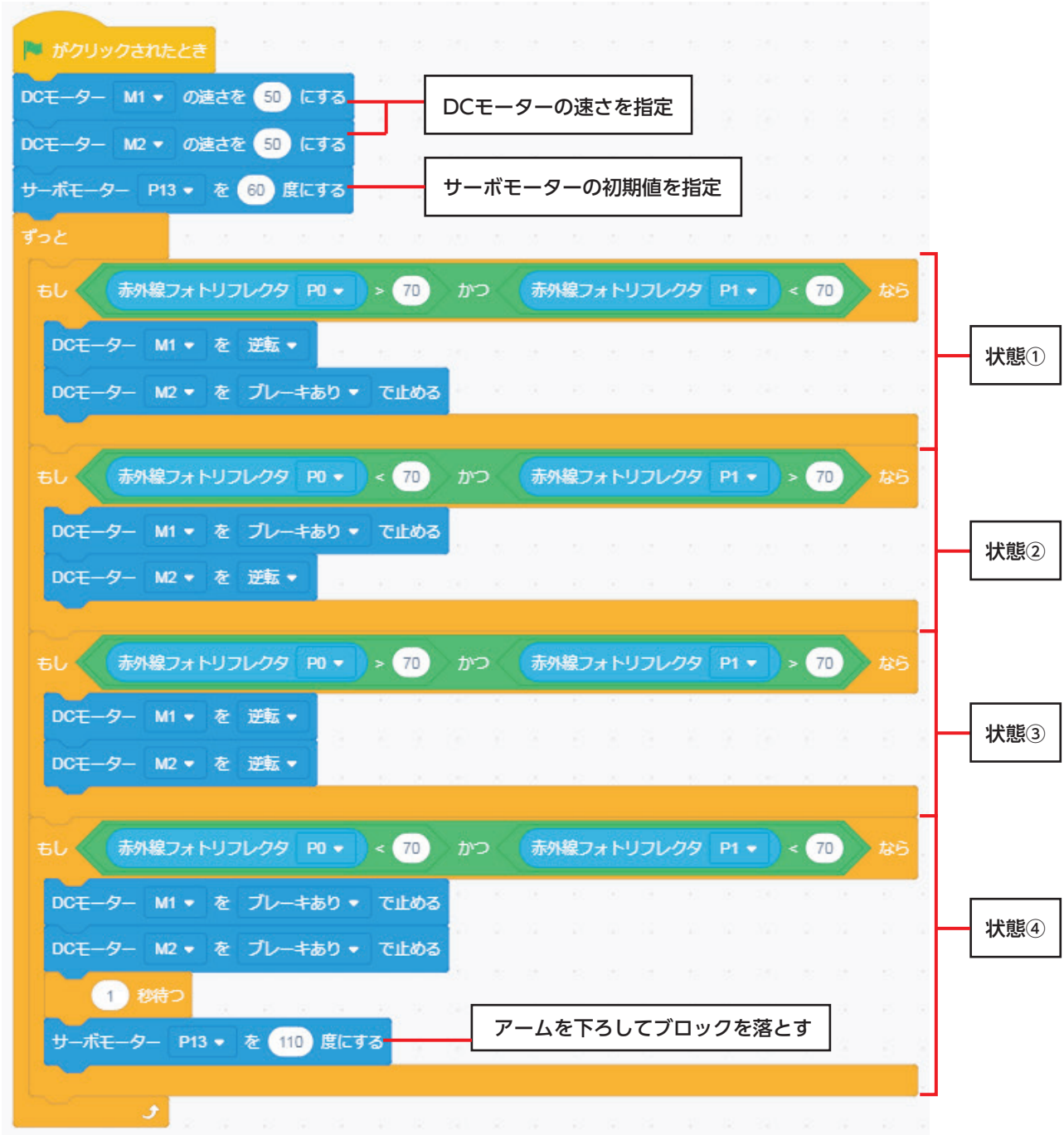
	右側の赤外線フォトリフレクタだけが黒線の上にあるとき	左側の赤外線フォトリフレクタだけが黒線の上にあるとき
状態	① 	② 
条件	赤外線フォトリフレクタ P0の値＞しきい値 かつ 赤外線フォトリフレクタ P1の値＜しきい値	赤外線フォトリフレクタ P0の値＜しきい値 かつ 赤外線フォトリフレクタ P1の値＞しきい値
動作	右折する	左折する
プログラム	DCモーター M1 を逆転 DCモーター M2 を停止	DCモーター M1 を停止 DCモーター M2 を逆転

	両方の赤外線フォトリフレクタが黒線の上にないとき	両方の赤外線フォトリフレクタが黒線の上にあるとき
状態	③ 	④ 
条件	赤外線フォトリフレクタ P0の値＞しきい値 かつ 赤外線フォトリフレクタ P1の値＞しきい値	赤外線フォトリフレクタ P0の値＜しきい値 かつ 赤外線フォトリフレクタ P1の値＜しきい値
動作	直進する	停止する
プログラム	DCモーター M1 を逆転 DCモーター M2 を逆転	DCモーター M1 を停止 DCモーター M2 を停止

④の動作の後にサーボモーターを動かし、アームからブロックを落とす動作を追加します。

複数の条件を同時に満たす条件（論理積）を判定するには以下のブロックを使用します。

サンプルプログラム



発展学習　テーマ1

交通管制センター

＜学習内容＞

信号機の制御が単独でおこなわれているわけではなく、様々な交通情報をもとにプログラムや通信技術を用いて渋滞などの問題解決に利用されていることを理解する。

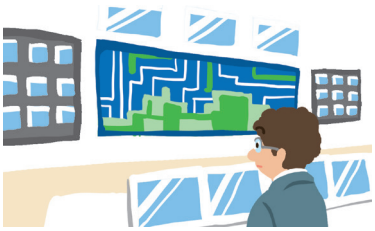
※基本学習テーマ１を学習の上実施してください。

交通管制センター

交通管制センターは、感知器など電子的に集めた交通情報（交通流、交通量など）をコンピューターで分析し、自動で信号機の制御を行ったり、監視カメラなどの他、警察官等により人的に集められた交通情報に基づき手動で信号機の制御を行なったりすることにより時間とともに変化する交通を適正に制御しています。他に、交通管制センターでは、集めた情報に基づき交通情報を交通情報板、ラジオ、カーナビゲーション等を通じてドライバー等に提供しています。

センサー

道路を走る車の台数などを計測



コンピューター

交通管制センターで交通情報を分析

アクチュエータ

信号機の点灯時間を変えるなどして渋滞をなくす

交通管制センターの目的

交通渋滞の緩和：交通状況に応じて信号機や交通情報板を制御し、交通渋滞を緩和する。
交通事故の防止：交通の流れを円滑にし、追突やイライラ運転による事故を減少させる。
交通公害の防止：車の停止回数を少なくし、排気ガス、騒音などの交通公害を緩和する。
経済効果：　車の停止回数の減少によって車の走行時間が短縮され、燃料が節約でき輸送コストを抑える。

デバイス間通信

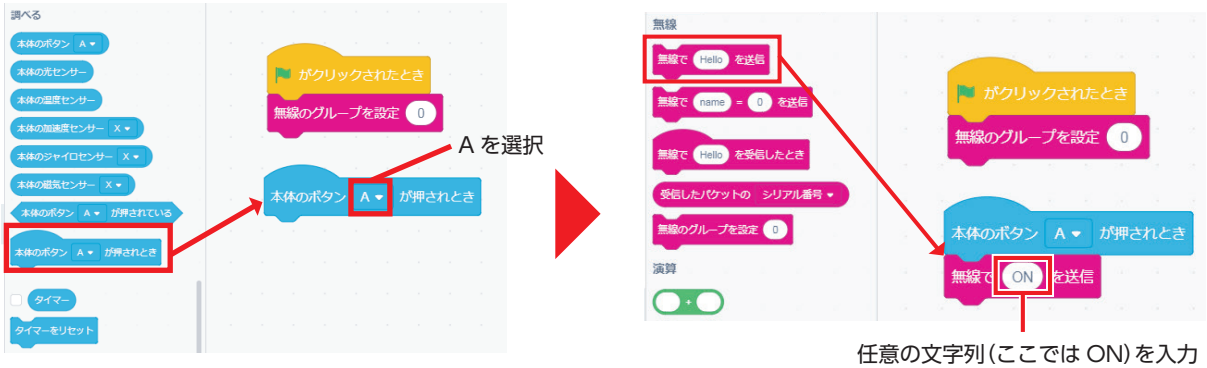
メインユニット（Studuino:bit）同士を無線（Wi-Fi）で通信させることができます。
以下のプログラムを作成して通信させてみましょう。

<A/B ボタンを押して、他のメインユニットの LED の点灯・消灯を制御するプログラム>

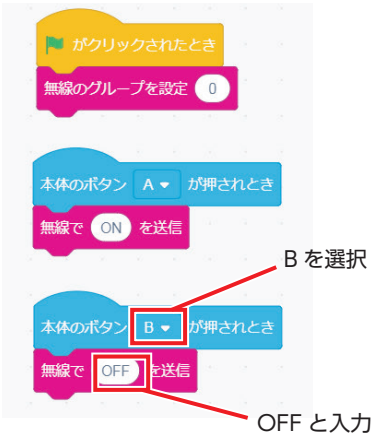
送信側のプログラム

①無線のグループを設定します。

②A ボタンを押したら「ON」という信号が送信されるプログラムを作成します。

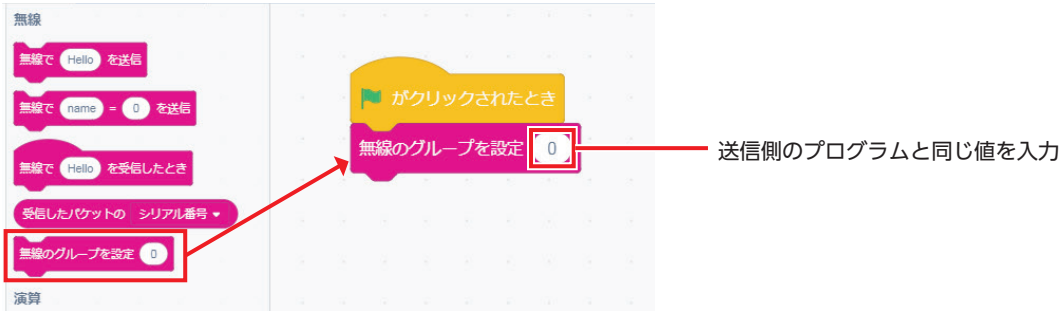


③同様に B ボタンを押したら「OFF」という信号が送信されるプログラムを作成します。



受信側のプログラム

①無線のグループを設定します。



②無線で「ON」を受信したら LED を点灯させるプログラムを作成します。

③同様に、無線で「OFF」を受信したら LED を消灯させるプログラムを作成します。

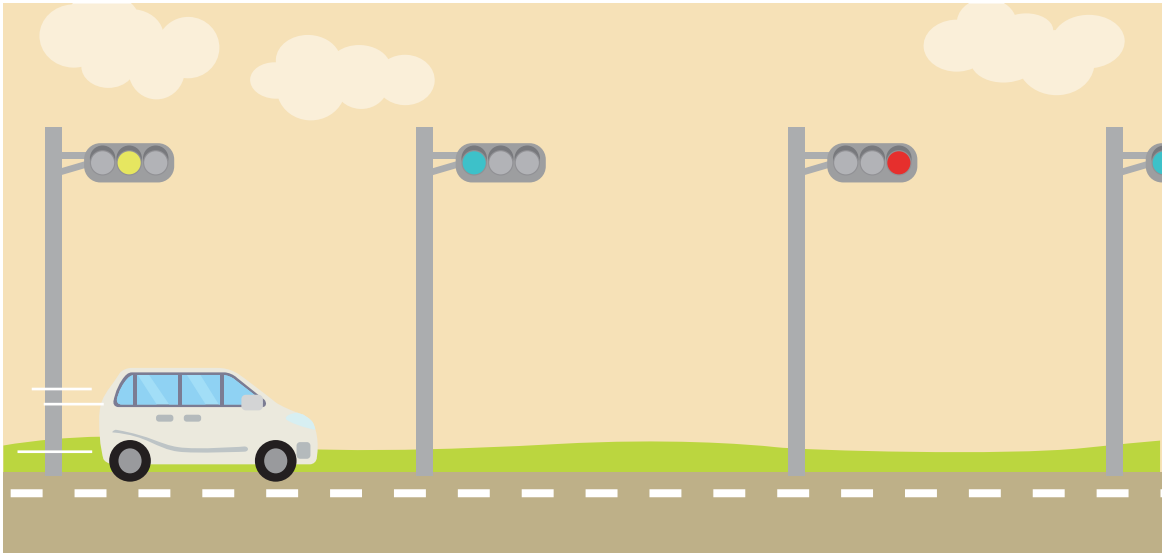
それぞれのプログラムを転送し、動きを確認しましょう。

プログラムが起動すると、
通信ランプが青色に点灯します。

送信側のボタンで受信側の LED の
点灯を制御できます。

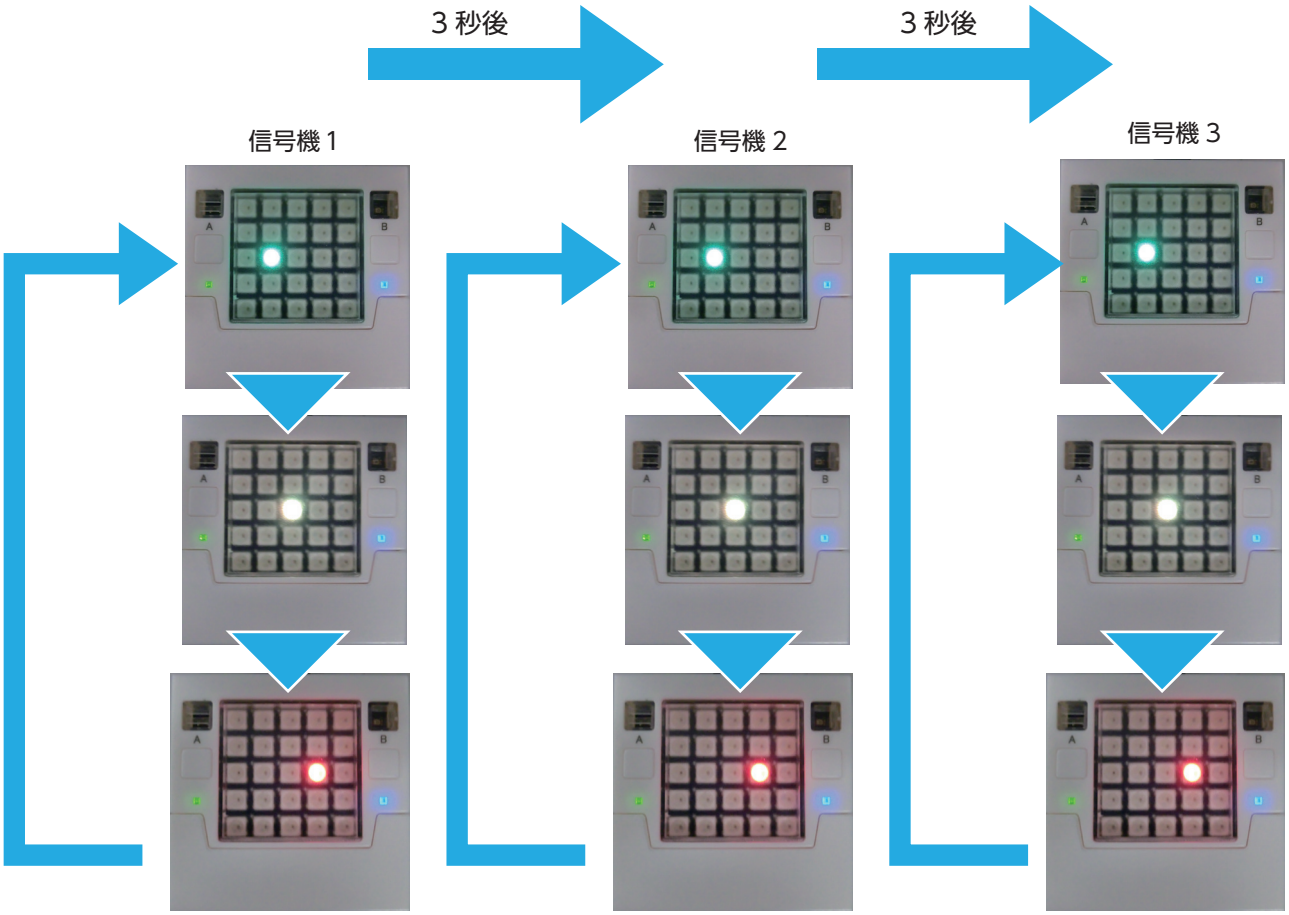
課題例

誰しも「車が全然来ないのに赤信号で止められている」という経験があるのではないでしょうか。多くの信号は一定間隔で赤と青が切り替わるようになっていますが、それによって無駄が生まれていることも確かです。
そんな問題点を解消するためのテクノロジーが、技術者たちによって開発されています。例えば、都心部の車通りの多い道路の信号は、運転者がまったく停車する必要のないよう信号が次から次へと青に変わります。



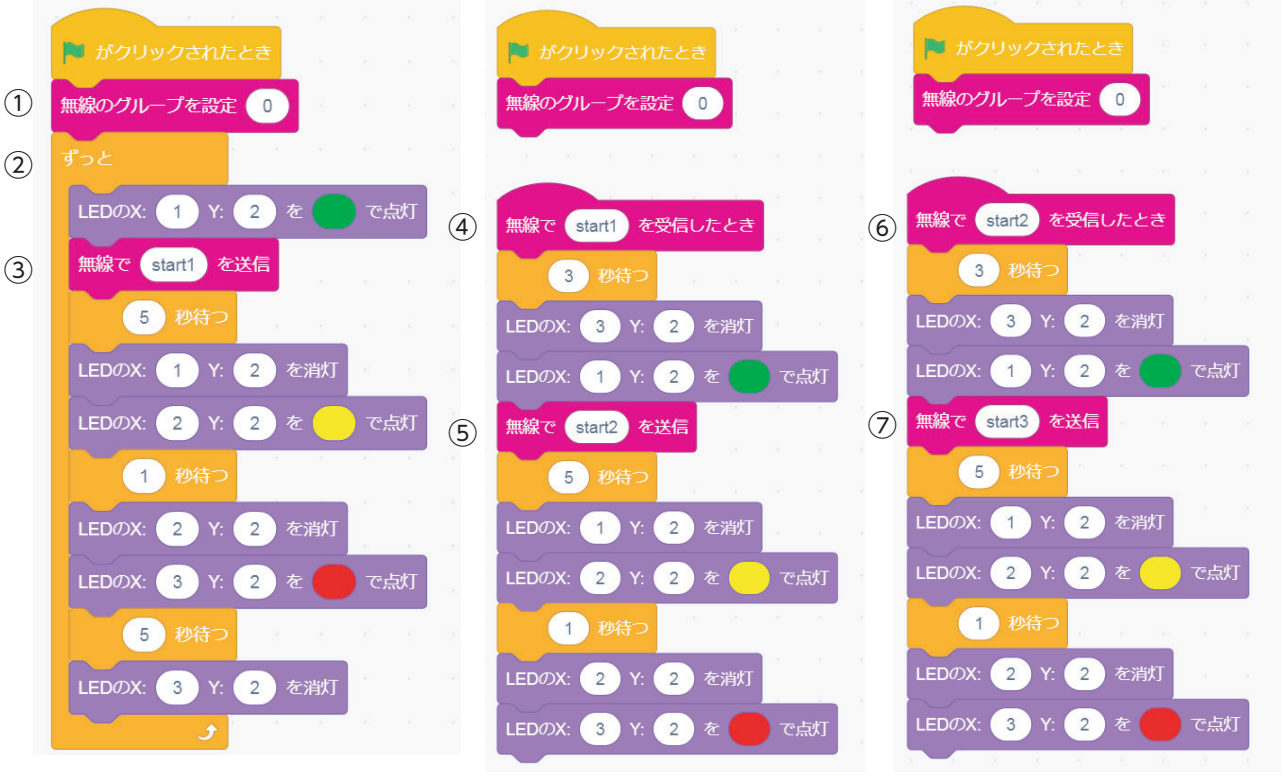
＜課題＞

3 台の信号機が変わるタイミングを 3 秒ずつずらして変わるように
無線通信を利用したプログラムを考えましょう。



プログラム作成例

信号機１　　信号機２　　信号機３



①信号機１～３の無線グループを同じ値で設定します。

②信号機１は緑→黄→赤と変化する自動車用信号機のプログラムを繰り返し実行します。

③信号機１の緑が点灯したタイミングで「start1」という信号を送信します。

④信号機２は「start1」という信号を受信したのち、３秒経過後に信号機１と同じ緑→黄→赤と変化する自動車用信号機のプログラムを実行します。

⑤信号機２の緑が点灯したタイミングで「start2」という信号を送信します。

⑥信号機３は「start2」という信号を受信したのち、３秒経過後に信号機１と同じ緑→黄→赤と変化する自動車用信号機のプログラムを実行します。

⑦さらに多くの信号を連携させる場合は、③～⑥と同様に「start3」「start4」と送受信ブロックを追加して無線で連携させることができます。

発展学習　テーマ2

高度道路交通システム

＜学習内容＞

信号機の変化の情報を無線で自動車に伝え、停止させる運転支援システムを構想する。

※基本学習テーマ１と基本学習テーマ２および、発展学習テーマ１のデバイス間通信を学習の上実施してください。

高度道路交通システム

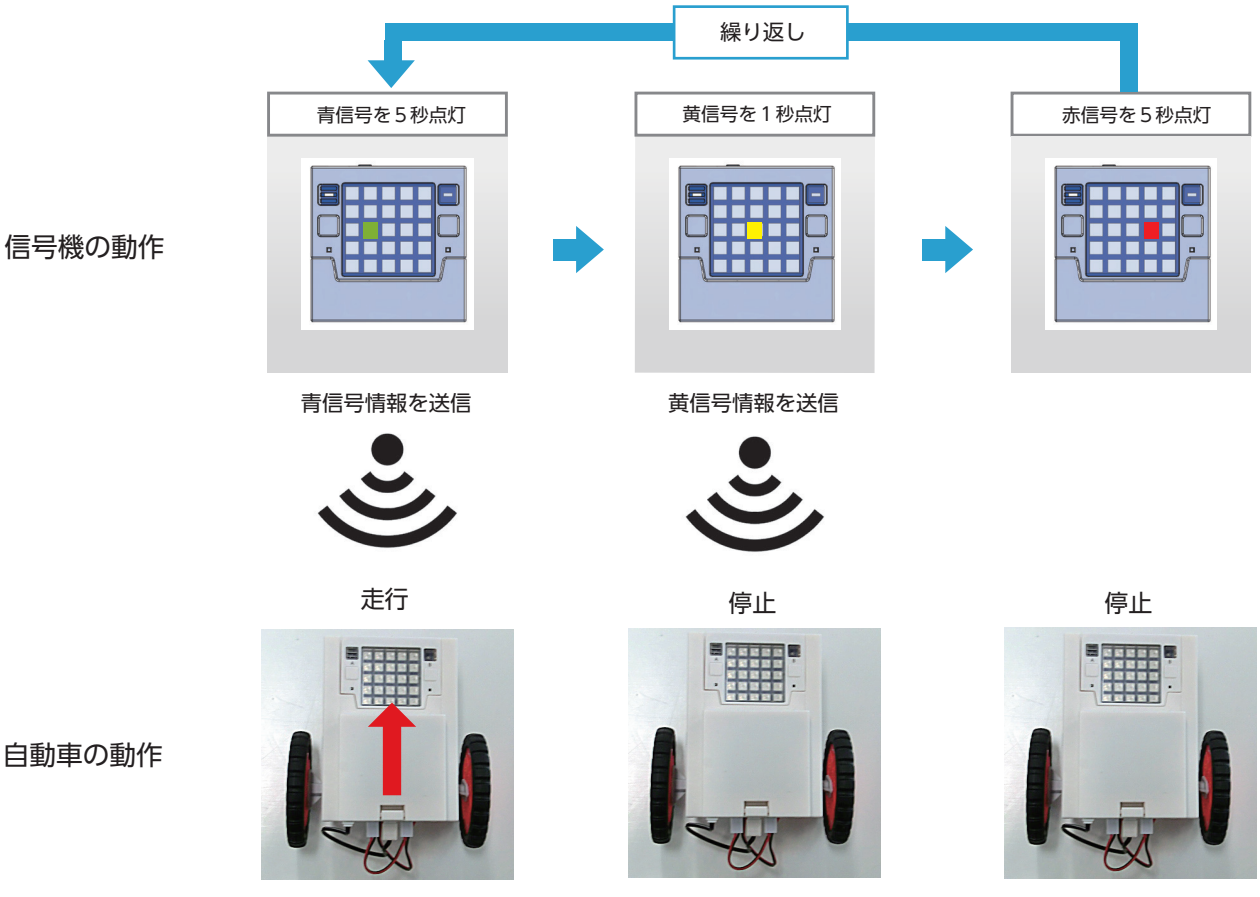
高度道路交通システムとは、最先端の情報通信技術や制御技術を利用して、人と道路と自動車の間で情報の受発信を行い、運転操作の支援等を行うシステムです。

その1つとして「信号情報活用運転支援システム」が上げられます。
「信号情報活用運転支援システム」とは、簡単に説明すると「信号が変わるタイミングを教えてくれるシステムのことです。

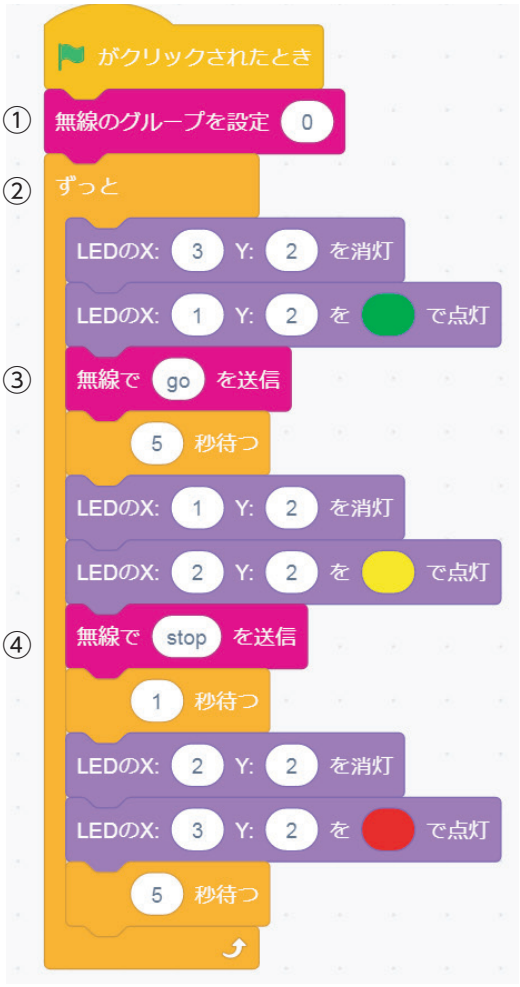
信号機が変わるタイミングを自動車が取得することで、直近の信号を青信号で通過できることをドライバーに伝達する（信号通過支援）、直近の信号が赤信号で停車する必要がある場合にドライバーに早めのアクセルオフを促す（赤信号減速支援）、ドライバーに信号が赤から青に変わる残り時間の情報を提示する（発進遅れ防止支援）
といった活用がされ、安全性や燃費向上、渋滞防止の効果がもたらされています。

課題

信号機の変化に応じて自動車を制御する自動運転システムを作成しましょう。



信号機



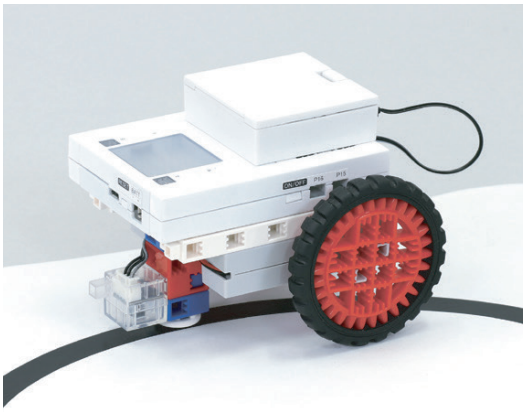
①自動車の無線グループを同じ番号で設定します。

②緑→黄→赤と変化する自動車用信号機のプログラムを繰り返し実行します。

③緑が点灯したタイミングで「go」という信号を送信します。

④黄が点灯したタイミングで「stop」という信号を送信します。

自動車



例①ロボットカーの応用

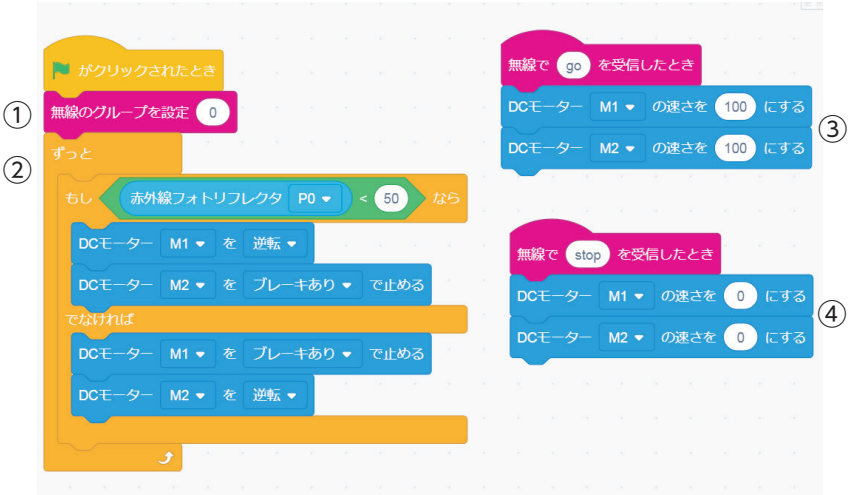
①信号機の無線グループを同じ番号で設定します。

②DC モーターの速さを設定します。

③「go」の信号を受信したとき、前進するプログラムを実行します。

④「stop」の信号を受信したとき、停止するプログラムを実行します。

例②ライントレースカーの応用



①信号機の無線グループを同じ番号で設定します。

②ライントレースカーのプログラムを実行します。（※DC モーターの速度指定はしません。）

③「go」の信号を受信したとき、DC モーターの速さを 100 に設定します。

④「stop」の信号を受信したとき、DC モーターの速さを 0 に設定します。

発展学習　テーマ3

ETCシステム

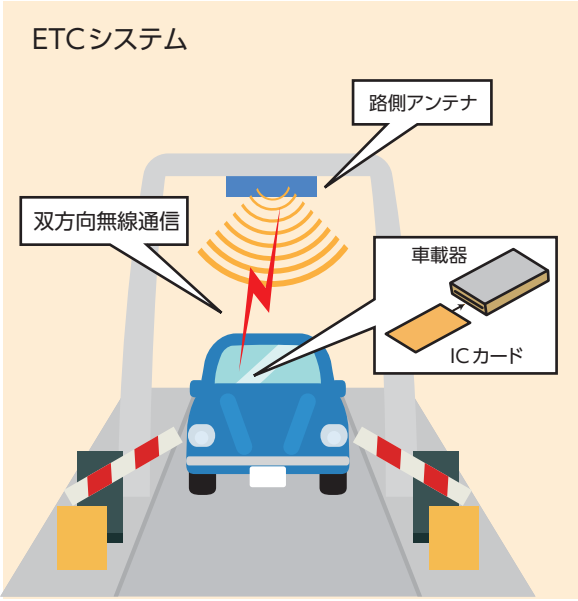
＜学習内容＞

自動ゲートが自動車の無線情報を検知して開閉を制御するETCシステムを構築する。

※基本学習テーマ２と応用学習テーマ１および、発展学習テーマ１のデバイス間通信を学習の上、実施してください。

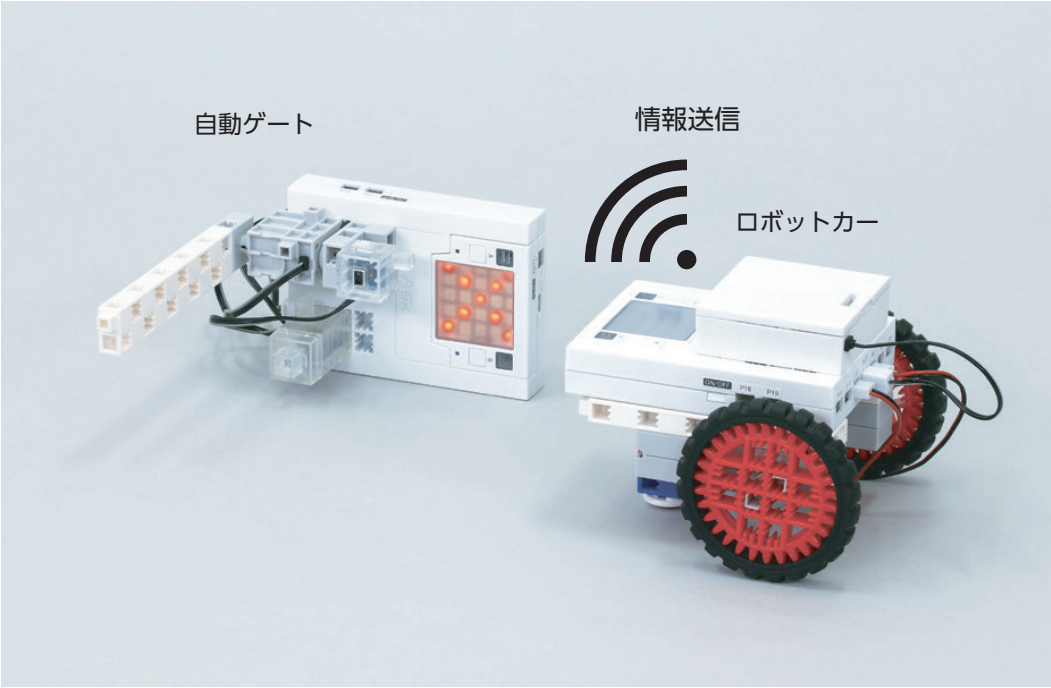
ETC システム

ETC システムとは電子料金収受システム（ Electronic Toll Collection System ）の略称で、高度道路交通システムのひとつです。
有料道路を利用する際に料金所で停止することなく通過できるため、渋滞緩和に役立っています。
ETC は、料金所のアンテナとクルマに装着した「車載器」との間で、通行料金に関する情報などを無線で交信します。



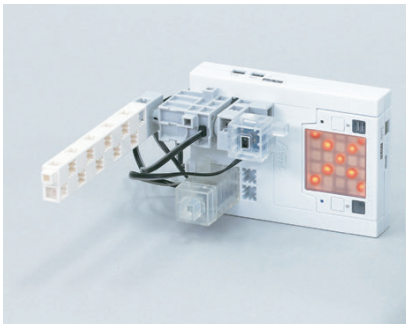
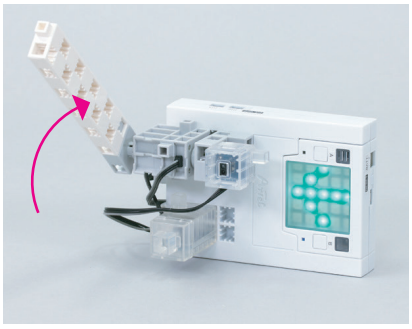
課 題

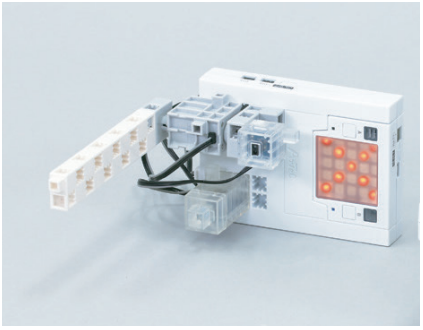
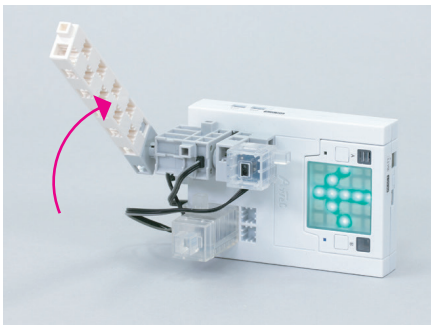
自動ゲートとロボットカーを改良して ETC システムを再現しましょう。



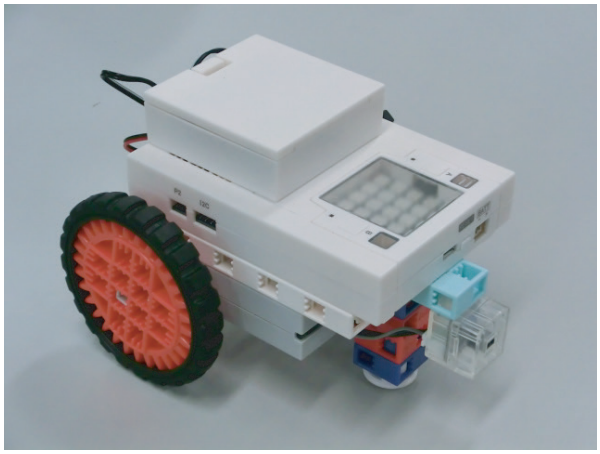
① 動作の整理

ロボットカーの条件と ETC システムの動作を整理しましょう。

	通常時（待機状態）	ETC 搭載車通過時
状態	① 	② 
条件	赤外線フォトリフレクタ P1 の値＜しきい値 かつ ロボットカーからの無線信号なし	赤外線フォトリフレクタ P1 の値＞しきい値 かつ ロボットカーからの無線信号あり
動作	LED に × を表示・ゲートを閉じる	LED に←を表示・ゲートを開く

	ETC 非搭載車通過時	手動で料金支払い時
状態	③ 	④ 
条件	赤外線フォトリフレクタ P1 の値＞しきい値 かつ ロボットカーからの無線信号なし	タッチセンサー P0 の値＝0
動作	LED に × を表示・ゲートを閉じる	LED に←を表示・ゲートを開く

◆ ロボットカー（衝突回避カー）



衝突回避カーの応用

①自動ゲートの無線グループを同じ番号で設定します。

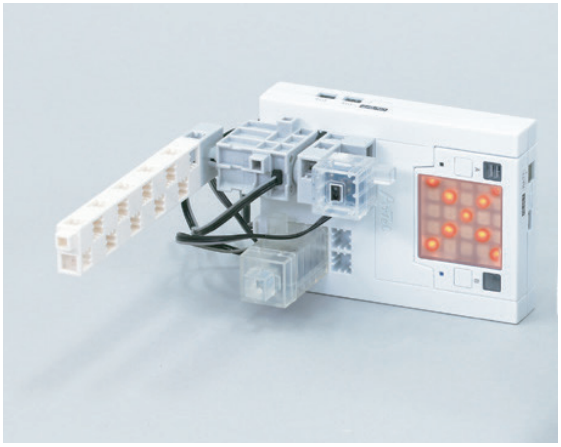
②DC モーターの速度を少し遅めで設定します。

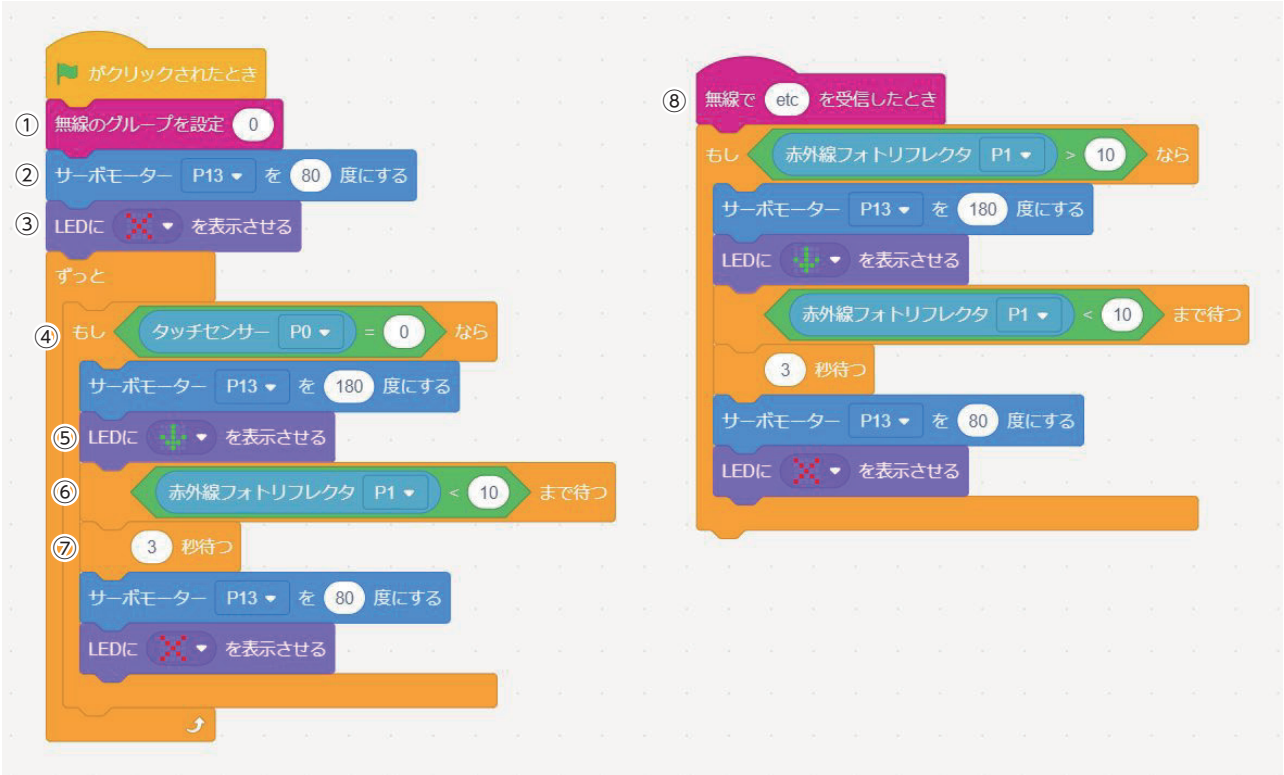
③衝突回避カーのプログラムを実行します。

④ゲートを感知して止るタイミングで「etc」という信号を送信します。

⑤ゲートにぶつからないようにゲートを感知しなくなった後、少し時間を待ってから
再度前進するように「○秒待つ」ブロックを追加します。

◆ 自動ゲート





①ロボットカーの無線グループを同じ番号で設定します。

②サーボモーターの角度を 80 度に設定します。
(ロボットカーの赤外線フォトリフレクタが感知できるように、ゲートを水平より少し下げます。)

③LED で × 表示をします。

④タッチセンサーが押されたらゲートを開けるプログラムを実行します。
(手動で料金を支払ったときの動作。)

⑤LED で←表示をします。

⑥赤外線フォトリフレクタで感知してロボットカーが通過するまで待ちます。

⑦ロボットカーがゲートに接触しないように、通過してから少し時間を待ってから
ゲートを閉じます。

⑧「etc」の信号を受信したときは、赤外線フォトリフレクタでロボットカーを感知した際に、
タッチセンサーを押したときと同様のプログラムを自動で実行します。

付 録

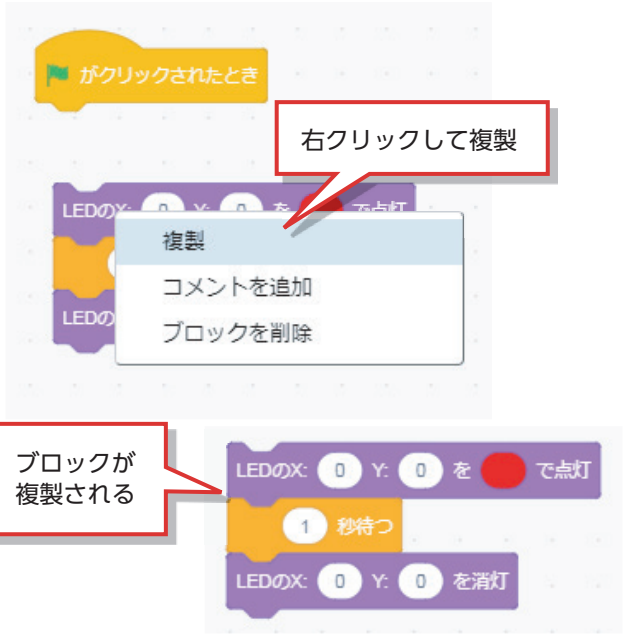
＜掲載内容＞

●操作方法
●ブロック一覧表

付録1　その他の操作方法

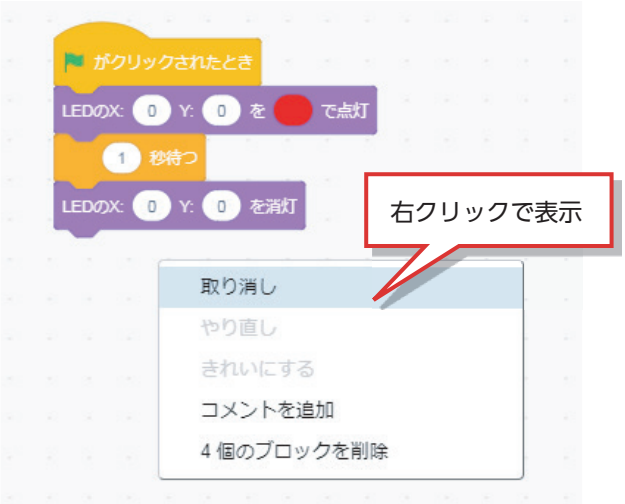
◆ ブロックの複製

つながっているブロックを複製できます。つなげたブロックの一部分だけ複製したい場合は、その部分を抜き出して同じ操作を行います。



◆ 取り消し

スクリプトエリアで右クリックをして表示されるメニューから取り消しを押すことで、一つ前の状態に戻すことができます。

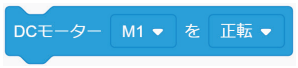
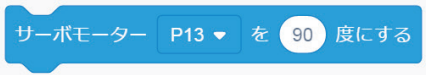
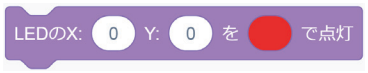
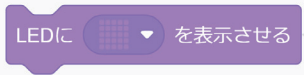
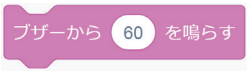


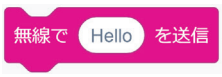
◆ プログラムの保存方法

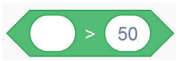
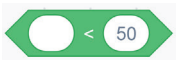
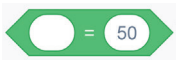
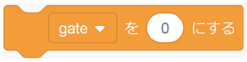
①「ファイル」から「コンピューターに保存する」を選択します。

付録 2　ブロック一覧表

テキストで使うブロックに関する使い方の一覧表です。

カテゴリ	ブロック	使い方
動き		DCモーターの回転方向を指定する
		DCモーターを停止させる
		DCモーターの速さを指定する
		サーボモーターを指定した角度にする
見た目		指定した座標のLEDを点灯させる
		指定した座標のLEDを消灯させる
		指定した文字をスクロールで表示させる
		全てのLEDの点灯色を個別に指定する
		全てのLEDの点灯色を同時に指定する
		全てのLEDを消灯させる
音		指定した高さの音を鳴らす
		指定した高さの音を指定時間鳴らす
		ブザーからの音を止める
イベント		起動したときに下に組み込まれたプログラムを実行する

カテゴリ	ブロック	使い方
制御		指定した秒数待つ
		囲われたプログラム指定回数くり返し実行する
		囲われたプログラムをずっとくり返し実行する
		囲われたプログラムを 条件が満たされたとき実行する
		条件が満たされたとき上段のプログラムを実行し、満たされないとき下段のプログラムを実行する
		条件が満たされるまで待つ
		条件が満たされるまで 囲われたプログラムをずっと実行する
調べる		本体のボタンの値を調べる
		本体の光センサーの値を調べる
		タッチセンサーの値を調べる
		赤外線フォトリフレクタの値を調べる
		本体のボタンが押されたときに下に組み込まれたプログラムを実行する
無線		無線で文字列を送信する
		無線で変数の値を送信する
		無線で文字列を受信したときに下に組み込まれたプログラムを実行する
		無線のグループを設定する

カテゴリ	ブロック	使い方
演算		「大なり」で値を比較する
		「小なり」で値を比較する
		「等しい」で値を比較する
		複数の条件を同時に満たす条件 (論理積)を判定する
変数		変数の値を調べる
		変数の値を指定した値に変える
		変数に指定した値を加算する
関数		下に組み込まれたプログラムを 関数として定義する
		関数を呼び出す