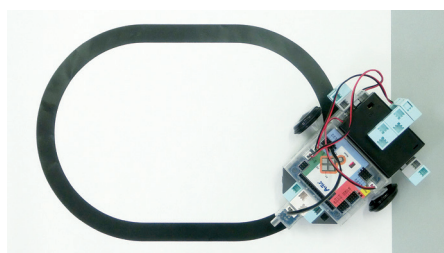
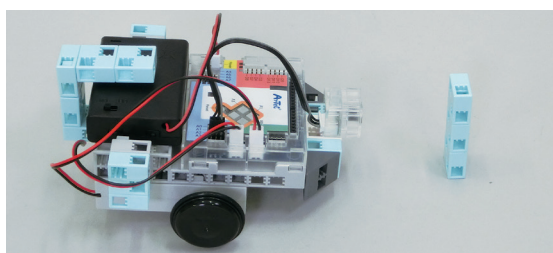
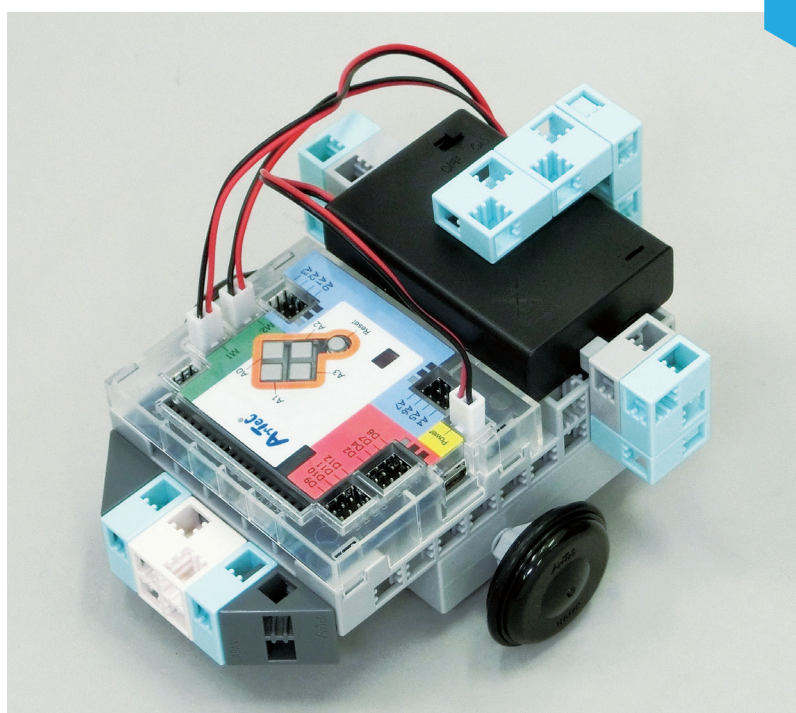


計測と制御キットD

プログラムによるセンサーカーの制御

教員用



目次

| | |
|------------------------|----|
| 第1章 身近なプログラミング | 3 |
| 第2章 順次処理とフローチャート | 10 |
| 第3章 車の制御 | 20 |
| 第4章 センサーによる車の操縦 | 28 |
| 第5章 衝突回避カーの製作 | 37 |
| 演習1 落下回避カーの製作 | 45 |
| 演習2 ライントレースカーの製作 | 51 |
| 組み立て説明..... | 57 |
| 付録..... | 63 |
| 確認問題集..... | 67 |

部品名などの記載について

教科書では「センサ」「モータ」と表記されていますが、本テキストではソフトウェアの表記に合わせるため「センサー」「モーター」※1と表記しています。

※1：JIS 規格ではどちらでも誤りではないとされています。

第1章

身近なプログラミング

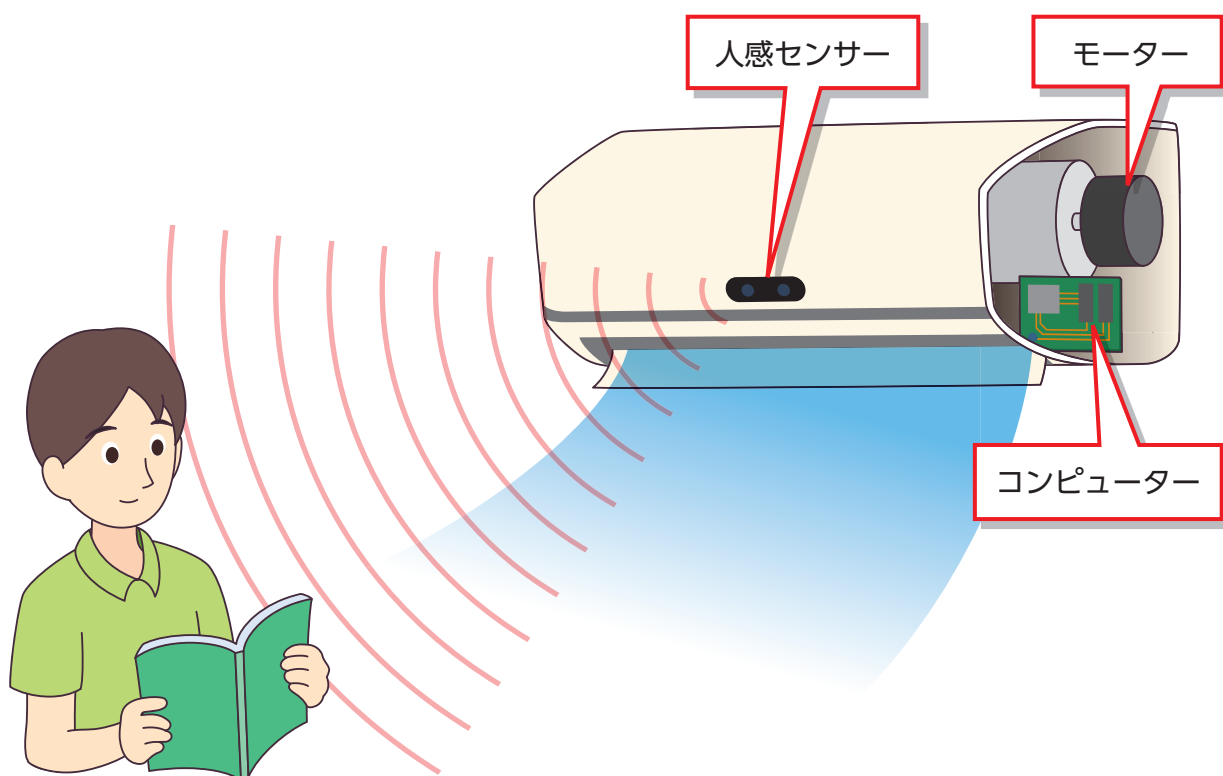
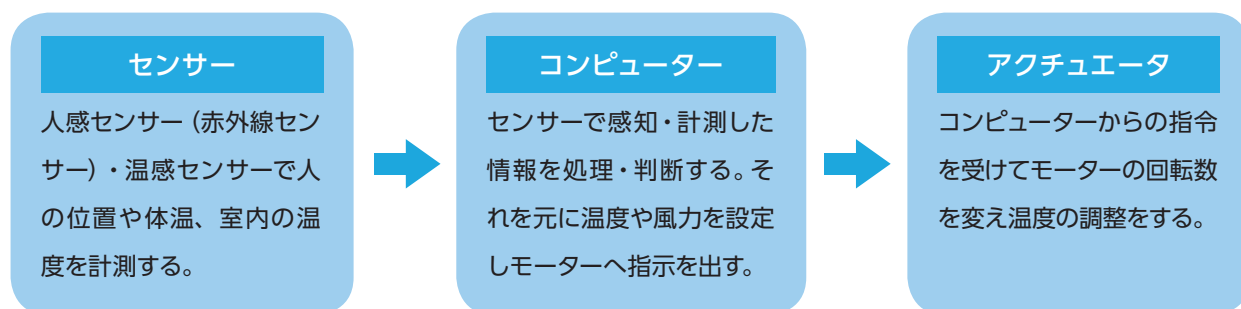
<学習内容>

- 身近な計測・制御システム
- プログラミング

1. 身の回りの計測・制御

私たちの身の回りには電気製品が、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



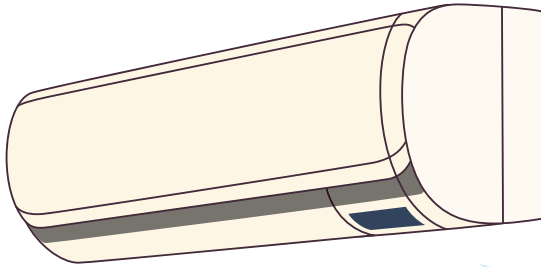
身近な製品でどういうものがコンピューターやセンサーを活用しているか探しましょう。

| 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 |

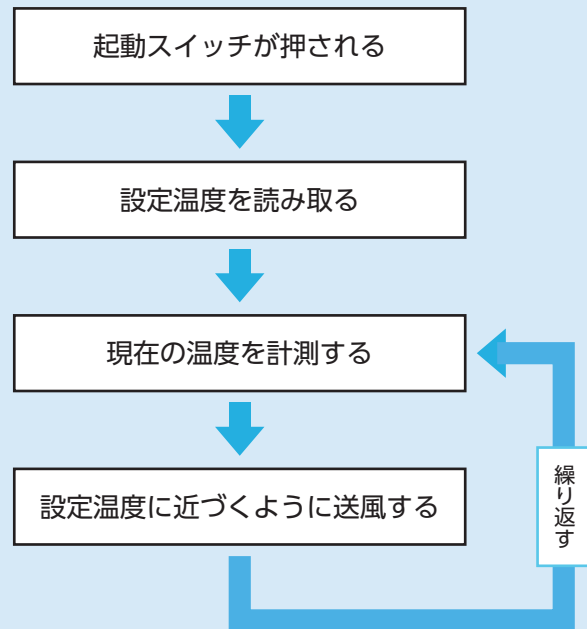
2. プログラミングとは

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もとに、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



人間が決めたエアコンの動作手順



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。

```
f ifNil: [
    "Turn setting language"
    self setLanguage: ltmp.
    ↑ self].

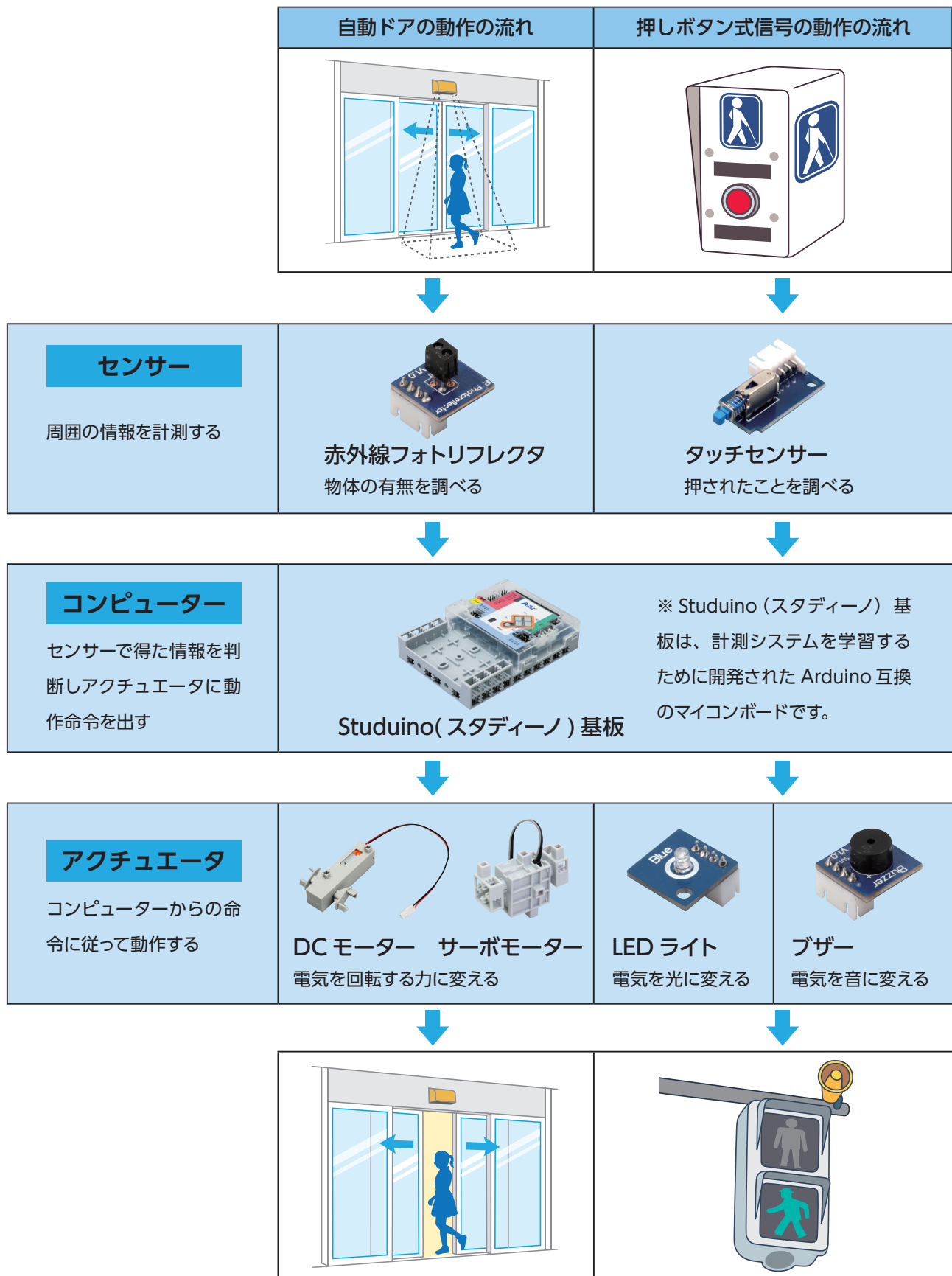
" Head and Footer file "
" Default English. "
headCode ← StandardFileStream new open: 'code\head+en.txt' forWrite: false.
footCode ← StandardFileStream new open: 'code\foot+en.txt' forWrite: false.

((ltmp = 'ja') | (ltmp = 'ja+HIRA')) ifTrue: [
    Transcript show: '----- Trans to Japan -----'; cr.
    headCode ← StandardFileStream new open: 'code\head.txt' forWrite: false.
    footCode ← StandardFileStream new open: 'code\foot.txt' forWrite: false.
]
```

▲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ArtecRoboパーツの対応

ArtecRobo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。



※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

The diagram illustrates the steps to launch the Studuino software and its main interface components. It starts with the Studuino Software icon, leading to the Studuino Programming Environment window. From there, it shows the 'ブロックプログラミング環境' (Block Programming Environment) selected, which leads to the 'ロボット' (Robot) category. The main interface is then shown with various components labeled:

- カテゴリ：命令の種類を選ぶことができます** (Category: You can select the type of command)
- 言語の選択** (Language Selection)
- プログラムの保存** (Save Program)
- メニュー** (Menu)
- 文字サイズの変更** (Change Text Size)
- ブロックパレット：センサーやアクチュエータへの命令が表示されます** (Block Palette: Commands for sensors and actuators are displayed)
- スクリプトエリア：命令をつないでプログラムをつくることができます** (Script Area: You can create a program by connecting commands)

The main interface components include:

- Studuino BLOCK Programming Environment** (Title Bar)
- メニューバー** (Menu Bar: ファイル 編集 実行 ヘルプ)
- ブロックパレット** (Block Palette: サーマーター, DCモーター, ブザー, LED, etc.)
- スクリプトエリア** (Script Area: 制御スタート)
- 文字サイズの変更** (Text Size Change: aA, aA, aA)

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」「制御」「調べる」「演算」「変数」の5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。



◆ 動き

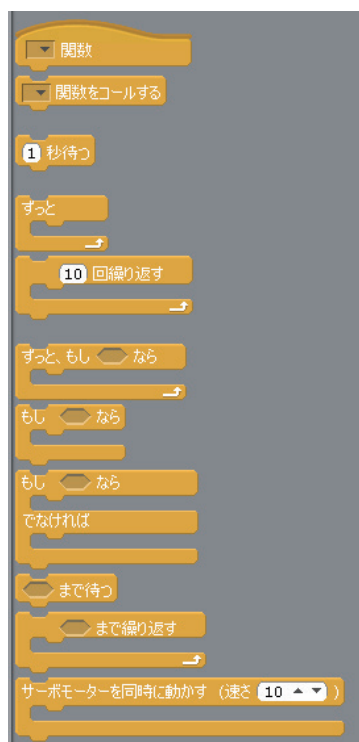
アクチュエータの動きを命令します。



※灰色のブロックは使用できません。
後述の設定変更で使用可能になります。

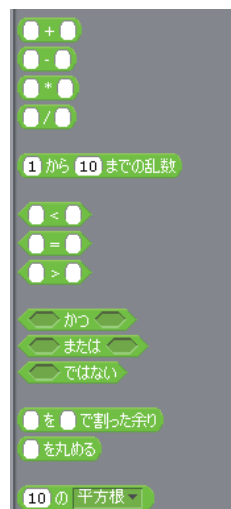
◆ 制御

命令の実行順を制御します。



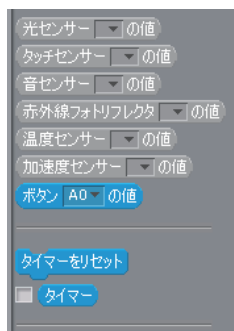
◆ 演算

計算の命令を出したり
条件式を作成します。



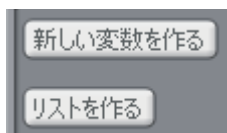
◆ 調べる

センサーを指定して
周りの情報を調べます。



◆ 変数

数値情報の記録に使います。
(今回は使いません)



第2章

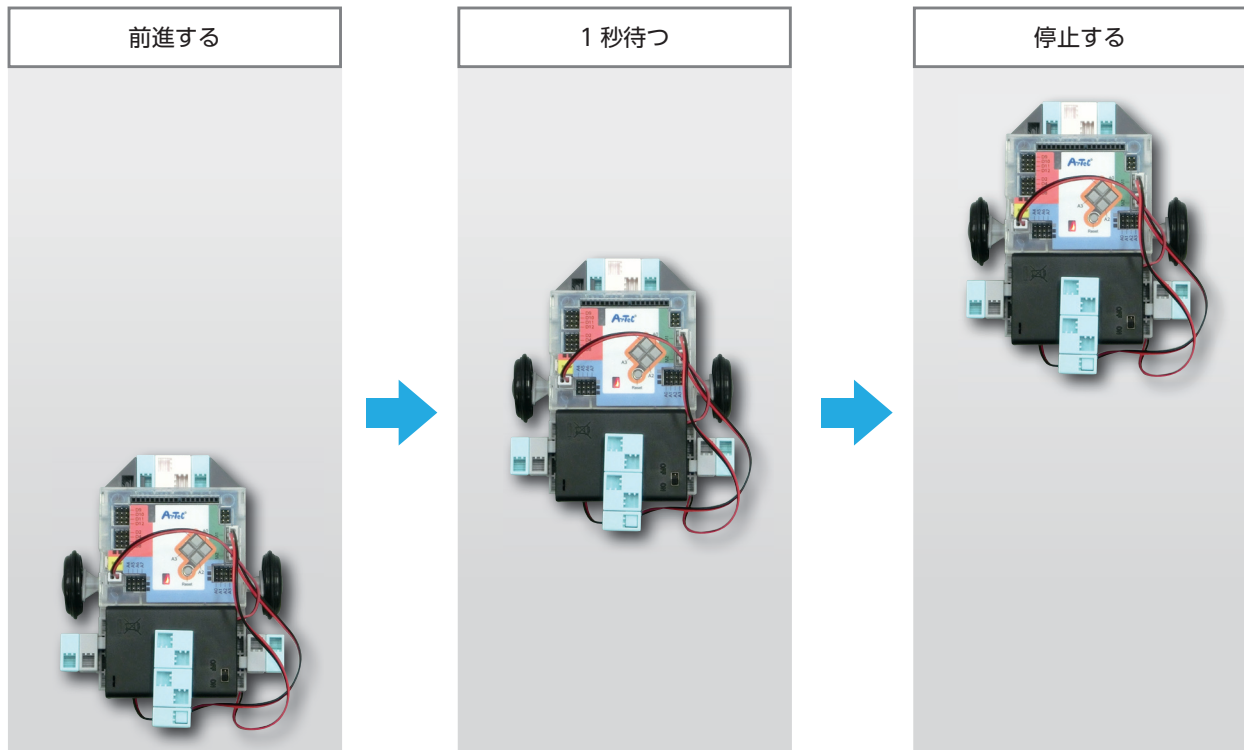
順次処理とフローチャート

<学習内容>

- 順次処理
- フローチャート

1. 順次処理のプログラム

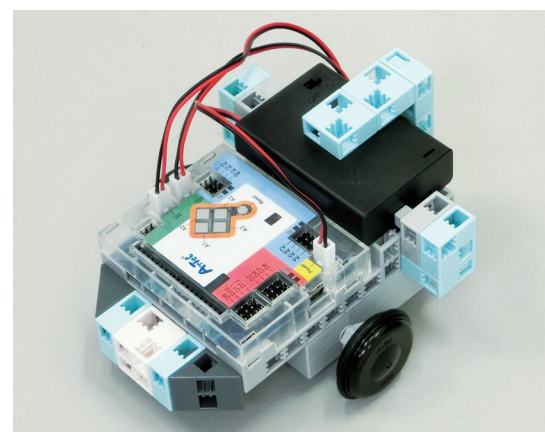
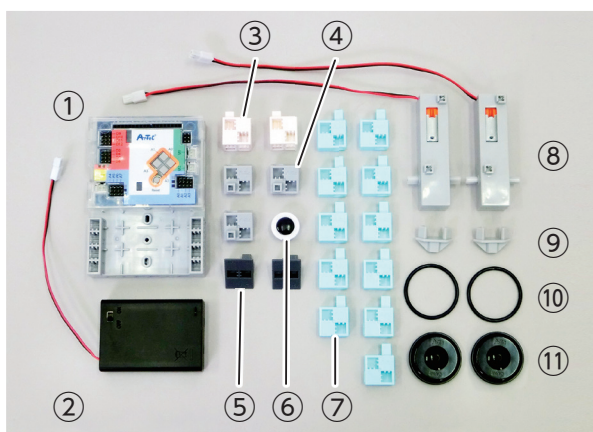
命令を順番通りに行う処理のことを「**順次処理**」といいます。DCモーターを使った車を作る中で、順次処理のプログラムを学びましょう。



①組み立て

58 ページの組み立て説明を見て、車を組み立てましょう。

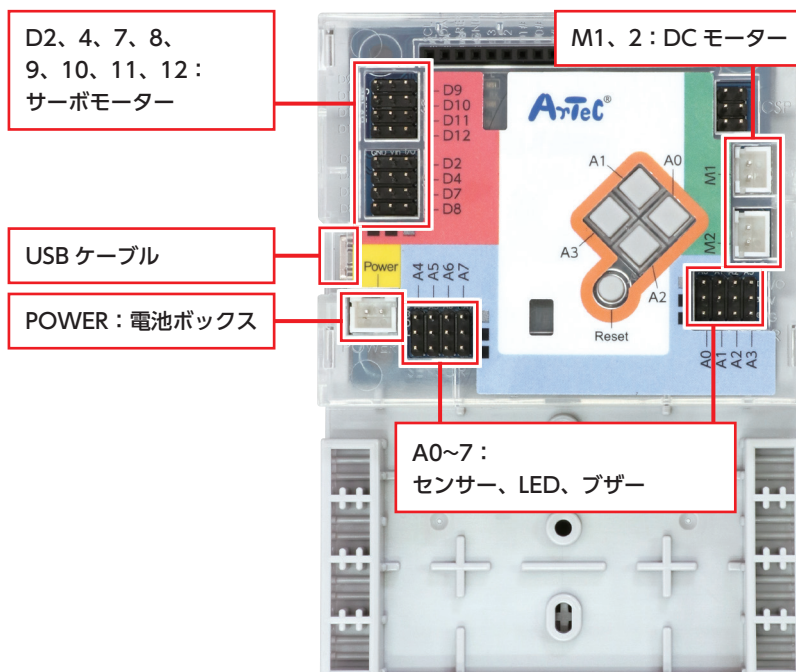
組み立て準備



- | | | |
|---------------------------|--------------------|--------------------|
| ① Studuino (スタディーノ) 基板… 1 | ⑤ 目玉パーツ …………… 1 | ⑨ DCモーター接続パーツ …… 2 |
| ② 電池ボックス(電池入) …… 1 | ⑥ 三角ブロック グレー …… 2 | ⑩ タイヤゴム …………… 2 |
| ③ 四角ブロック 白 …………… 2 | ⑦ ハーフブロック 薄水 …… 11 | ⑪ タイヤ …………… 2 |
| ④ ハーフブロック 薄グレー … 3 | ⑧ DCモーター …………… 2 | |

Studuino (スタディーノ) 基板に接続できるパーツ

Studuino (スタディーノ) 基板にセンサーやアクチュエータをつなぎ、プログラムを書き込むことで簡単に計測・制御システムを作ることができます。パーツによってコネクタをつなぐ場所が決まられているので注意しましょう。

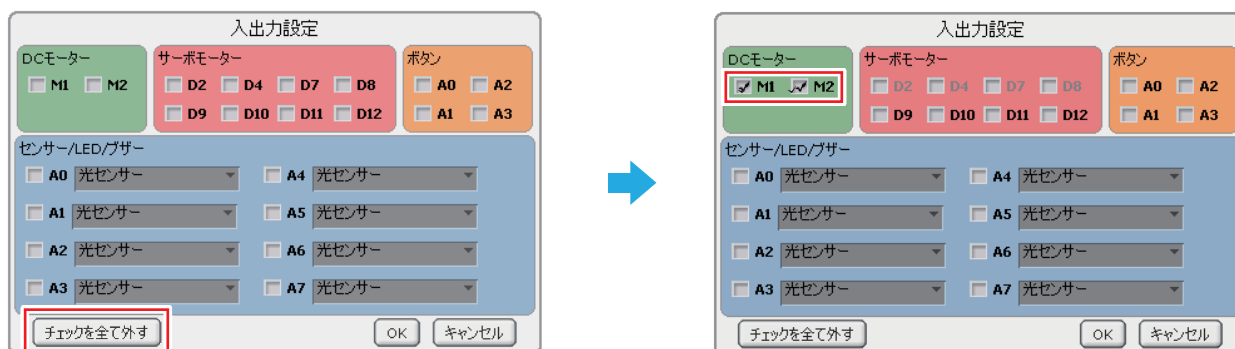


②入出力設定

入出力設定でStuduino (スタディーノ) 基板のどの場所にどのパーツをつないでいるかを登録します。



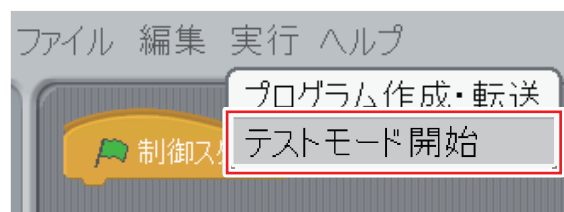
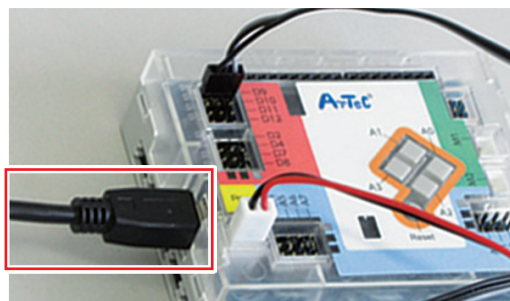
M1とM2につないだDCモーターを登録します。一度全てのチェックを外してから、DCモーターのM1とM2にチェックを入れて「OK」をクリックしましょう。



Studuino (スタディーノ) 基板はつないだパーツを自動で認識できないので、新しくパーツをつないだり、パーツをつなぎかえたりしたときは必ず入出力設定を行うようにしましょう。

③テストモード

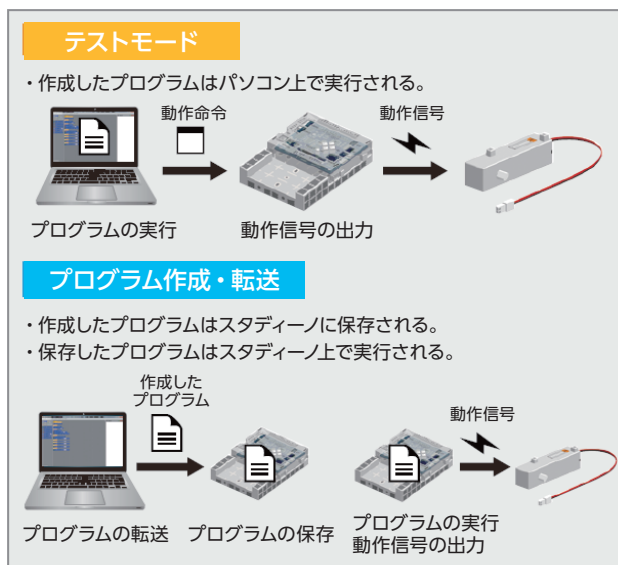
テストモードはパソコンとStuduino（スタディーノ）基板が常に通信している状態にする機能で、**動作を確認しながらプログラムを作る**ことができます。USBケーブルでコンピューターとStuduino（スタディーノ）基板をつないで、テストモードにしましょう。



USBケーブルの接続時は**パソコンから電力が供給されるので、電池ボックスのスイッチをオンにしなくてもブザーやLEDを動かすことができます**。ただし、**DCモーターやサーボモーターはパソコンからの電力が使用できないため、必ず電池ボックスを使う**ことに注意しましょう。

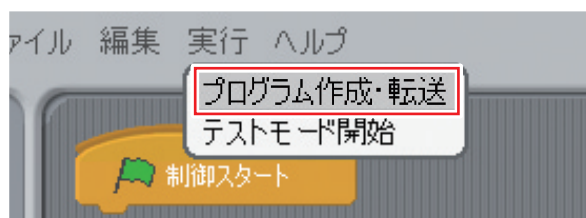
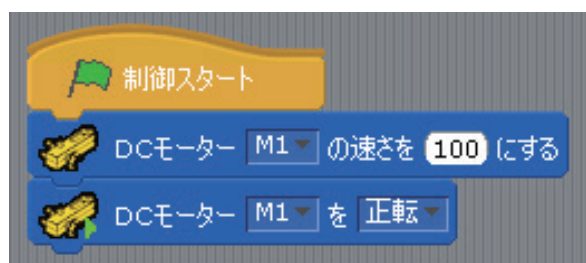
コンピューターとStuduino（スタディーノ）基板を離して使いたいとき

プログラムを実行させる方法には「テストモード」のほかに、「プログラム作成・転送」があります。「プログラム作成・転送」は発表などでコンピューターとStuduino（スタディーノ）基板を離して使いたいときに行います。



◆ プログラムの転送方法

作成したプログラムを「制御スタート」につないでから「プログラム作成・転送」をクリックします。

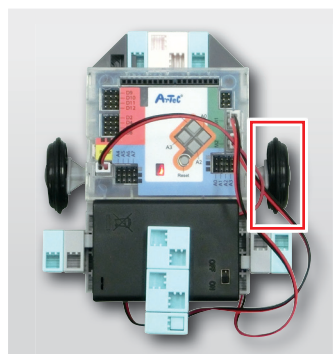


プログラム作成・転送は、プログラムを作るたびに転送を行う必要がありますが、**USBケーブルを外してもプログラムを実行できたり、プログラムの実行や読み込みが早くなる**というメリットもあります。

④DC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につないだ右のDCモーターのタイヤに注目しましょう。



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。



電池ボックスのスイッチをオンにして、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、

DCモーター M1 の速さを 100 にする 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。

速さを変えられる

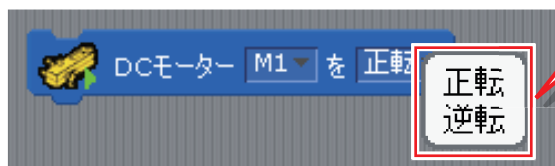


※数値を変更した場合、再度 DCモーター M1 の速さを 100 にする をクリックしないと、変更した数値が反映されません。

DCモーターの回転

DCモーター M1 の速さを 100 にする をクリックした後、DCモーター M1 を 正転 をクリックすると DC モーターが正転します。

「正転」と「逆転」を変えることで、DCモーターの回転する向きを変えることができます。



※速さを設定した後でないと、DCモーター M1 を 正転 をクリックしても DC モーターは動きません。

DCモーターの停止

DCモーター M1 を 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「停止」と「解放」を選択することができます。

DCモーターを短絡させて負荷をかけ、ぴったりと止める

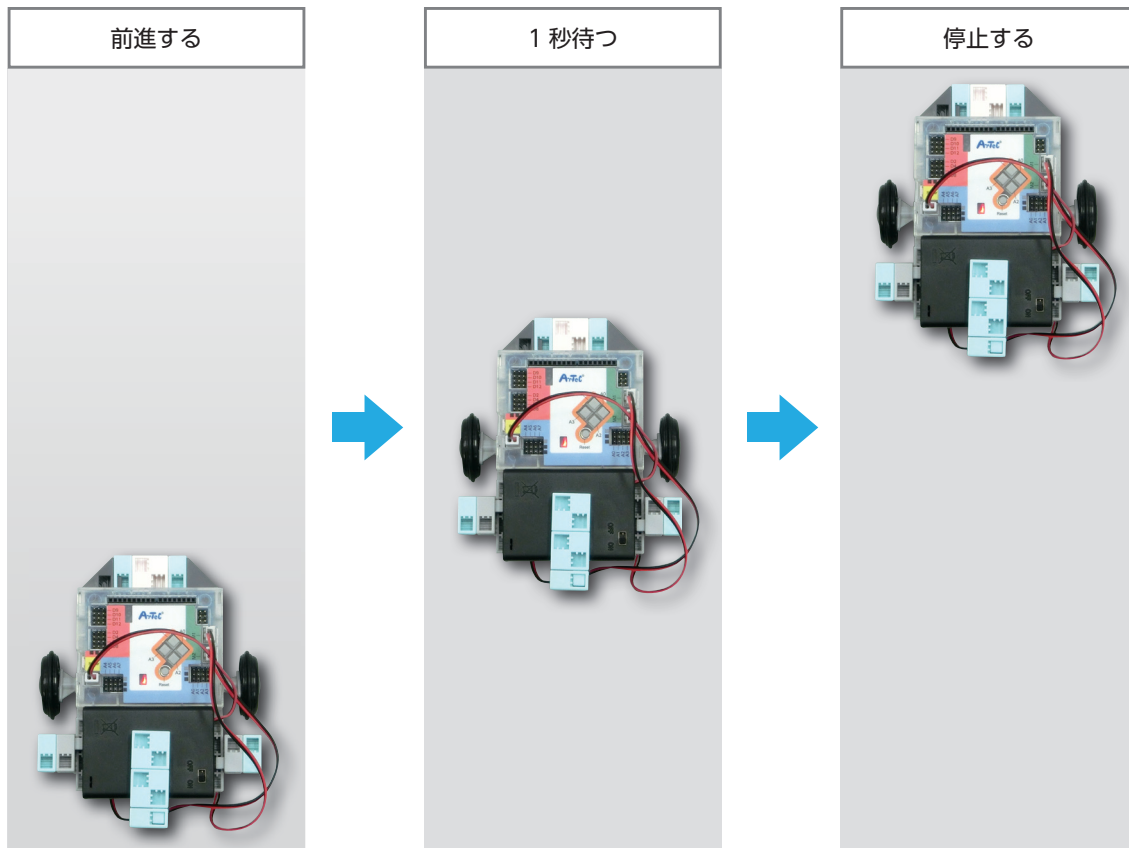


DCモーターにかかる電圧を0にして、惰性でゆっくりと止める

※本テキストでは「解放」は使用しません。

⑤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。



両輪のDCモーターが同じ速さで動けば、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、正転させましょう。



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モーターを停止させるブロックをつなぎます。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。

実行されているブロック全体が白枠で囲われる

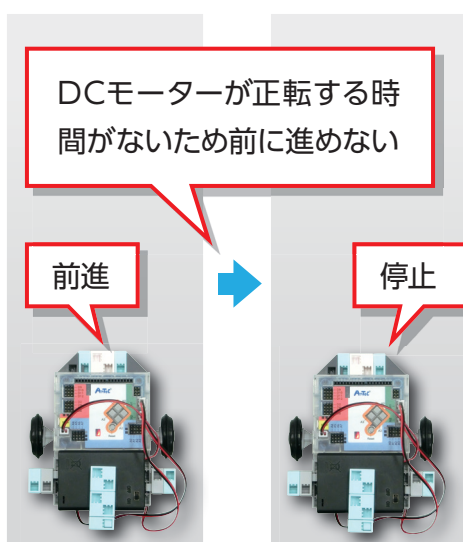
このように上から順番に命令が行われるプログラムの処理を「**順次処理**」といいます。

「1秒待つ」のブロックを入れないとどうなるのか？

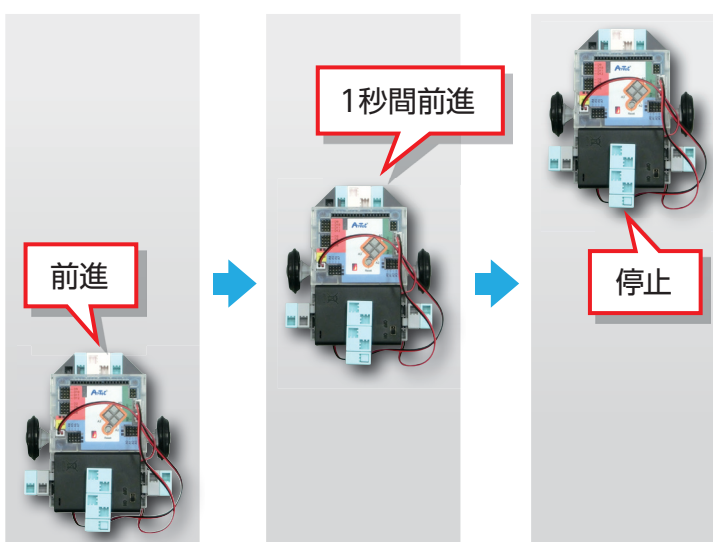


左のプログラムで動作をさせた場合、車はほとんど動きません。これはコンピュータが**プログラムを非常に高速で処理していることが原因**です。つまり、「DCモーターを正転」の命令の直後に、「DCモーターを停止」の命令が行われてしまうため、車が前に進む時間がありません。

「1秒待つ」を入れない場合




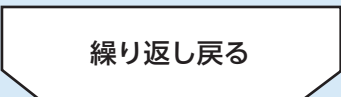
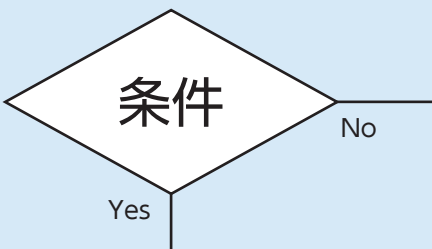


「1秒待つ」を入れる場合

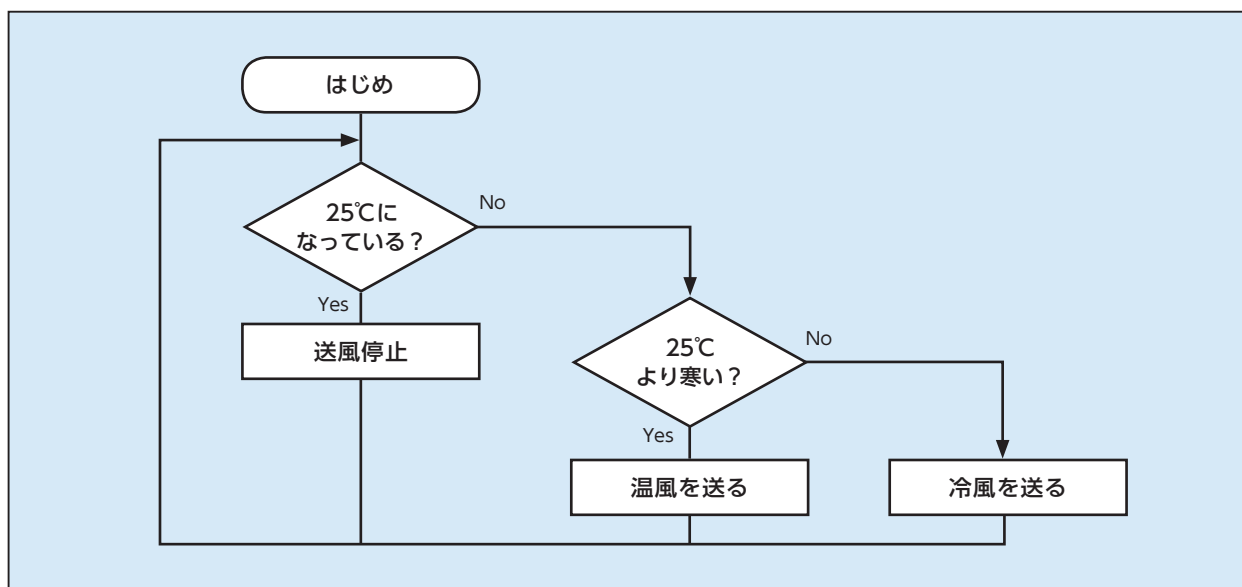


2. フローチャート

処理の手順をあらかじめ整理しておくことでプログラムが作りやすくなります。手順を整理したり考えをまとめる方法の一つとして、「**フローチャート**」という図がよく用いられます。

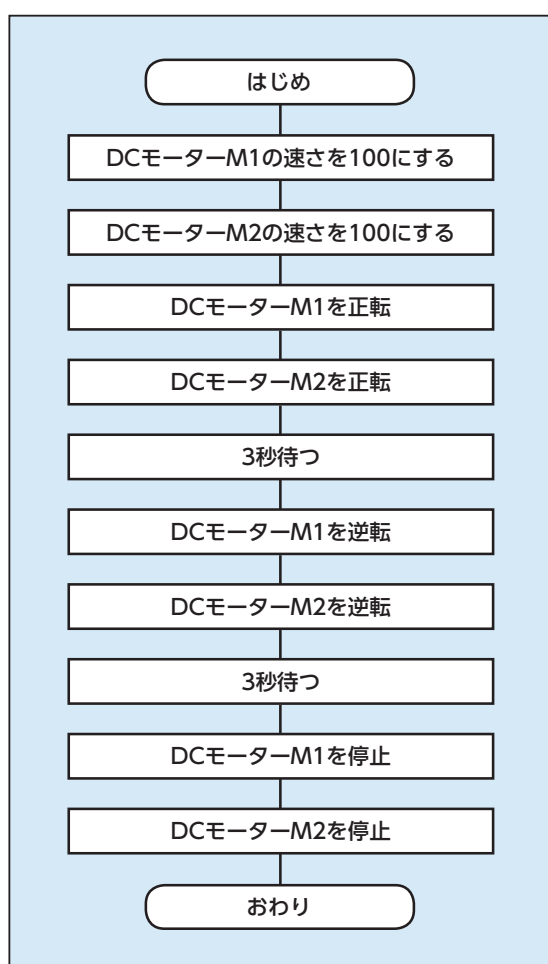
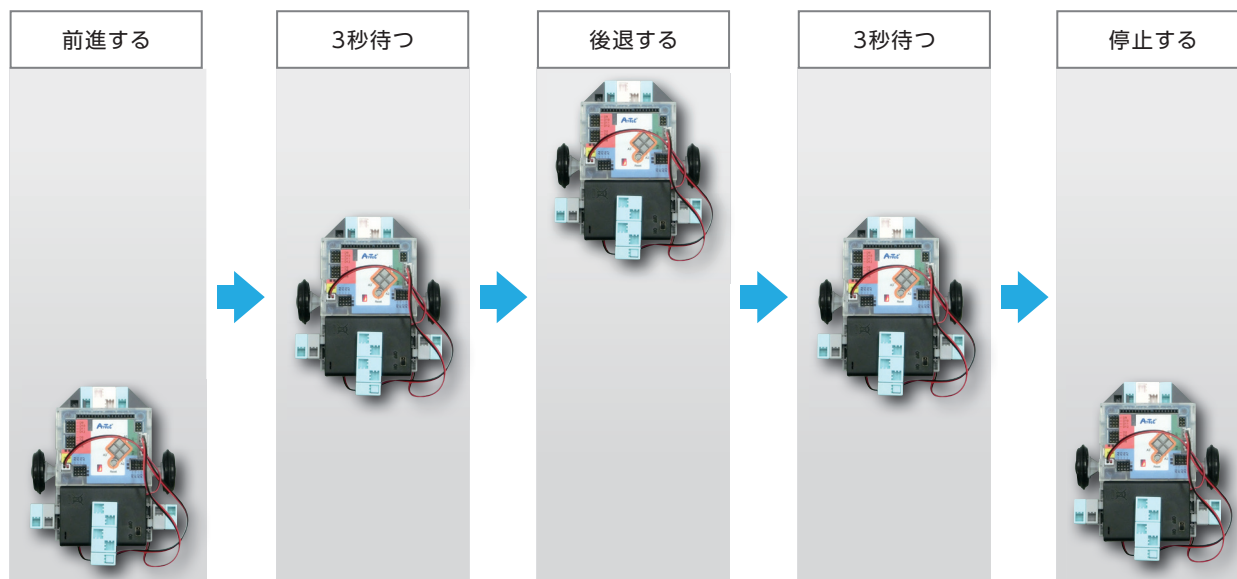
| 代表的なフローチャート記号 | |
|---|-----------|
|  | 処理の開始・終了 |
|  | 一般的な処理 |
|  | 繰り返し処理の開始 |
|  | 繰り返し処理の終了 |
|  | 条件で処理を分ける |

例 室温を25℃に保つエアコンの動作の手順を表したフローチャート



練習課題

次の動作の手順をフローチャートでまとめたあと、プログラムをつくりましょう。



DCモーターの回転する速さを途中で変更する必要がない場合は、最初に1度だけ設定します。



※テストモードで動作確認を行うとつないでいるUSBケーブルが動作の妨げになるので、プログラムを作成・転送し、USBケーブルをはずしてから動作を確認してください。

第3章

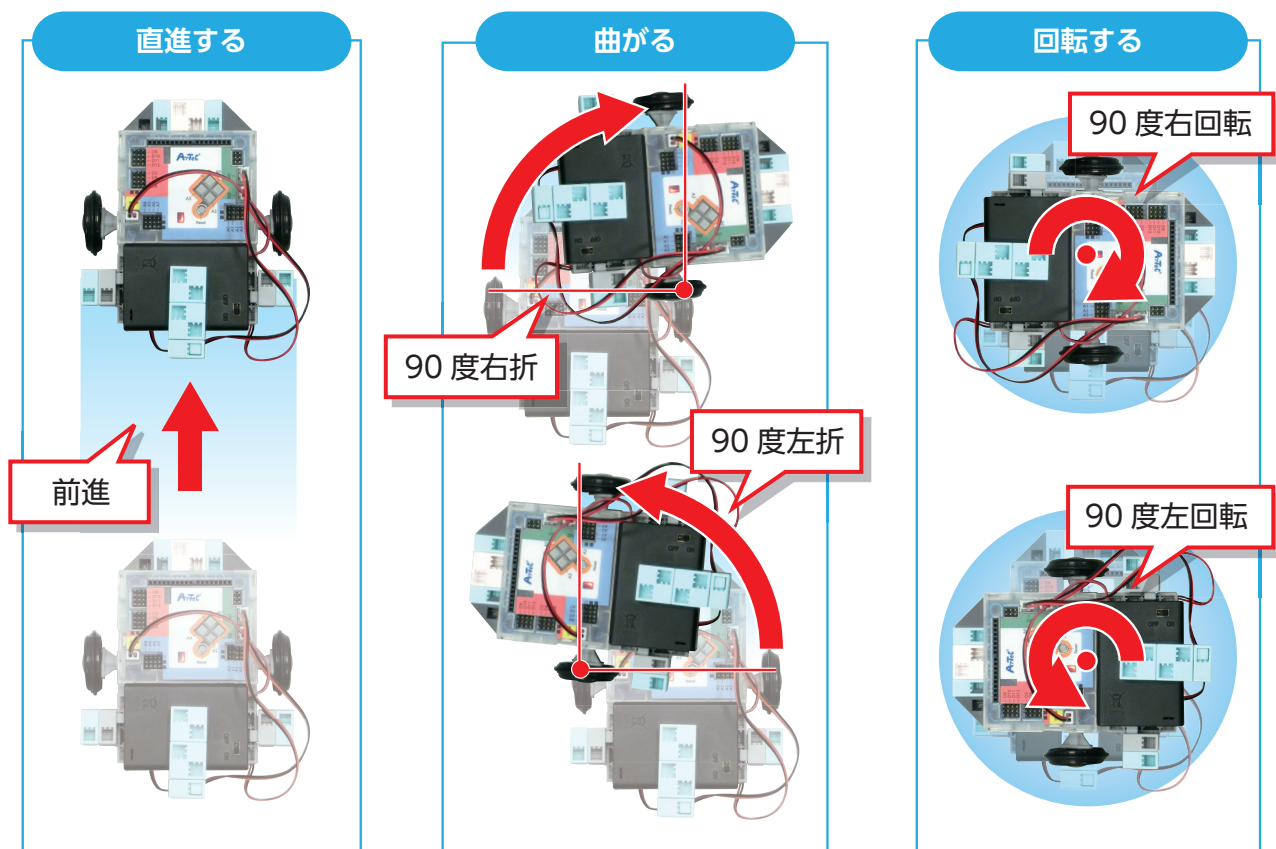
車の制御

<学習内容>

- 車の制御
- 繰り返し

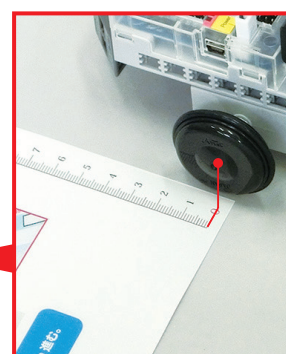
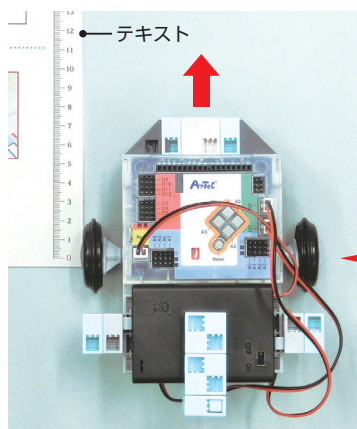
1. 車の制御

第2章と同じ車を使い、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作り、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。

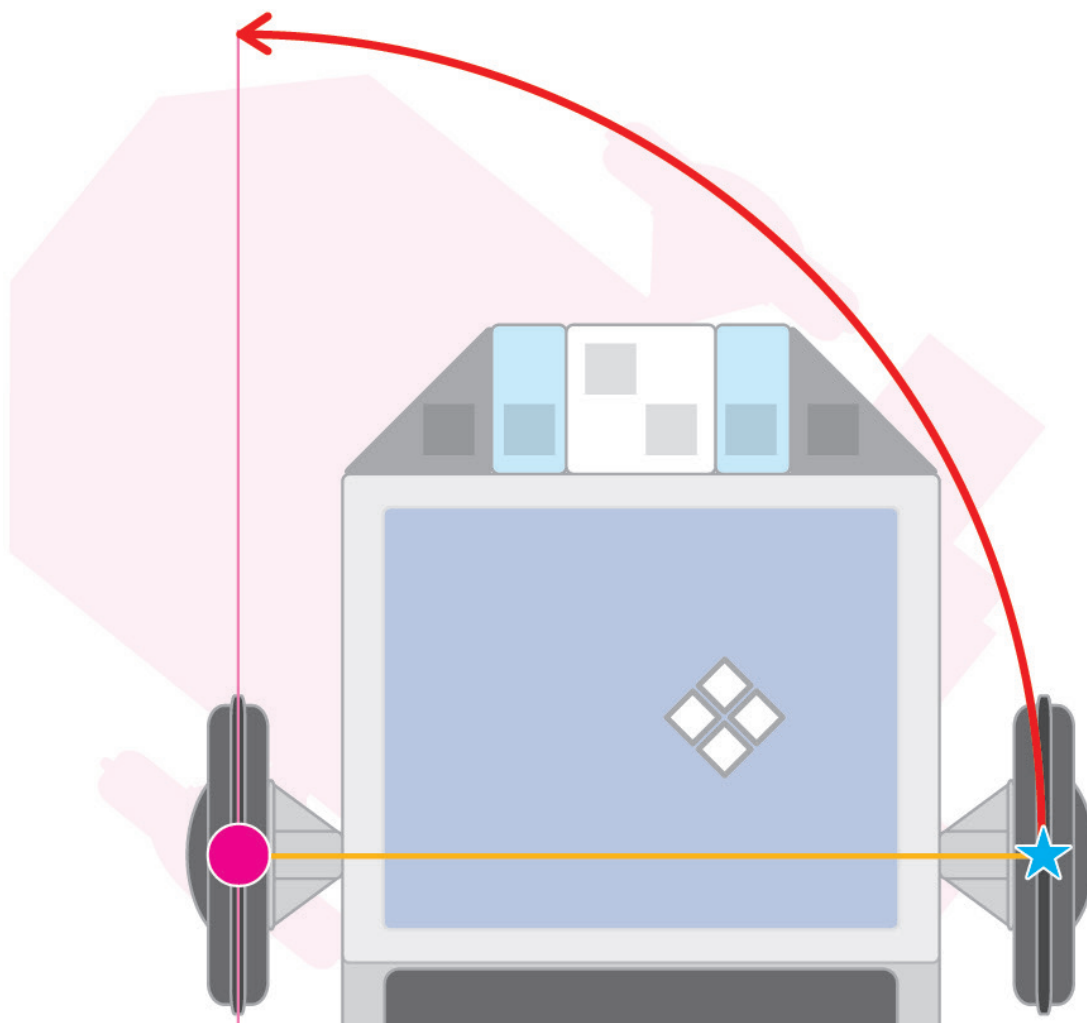
②左折

左折は右のDCモーターだけを正転させることで実現できます。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。

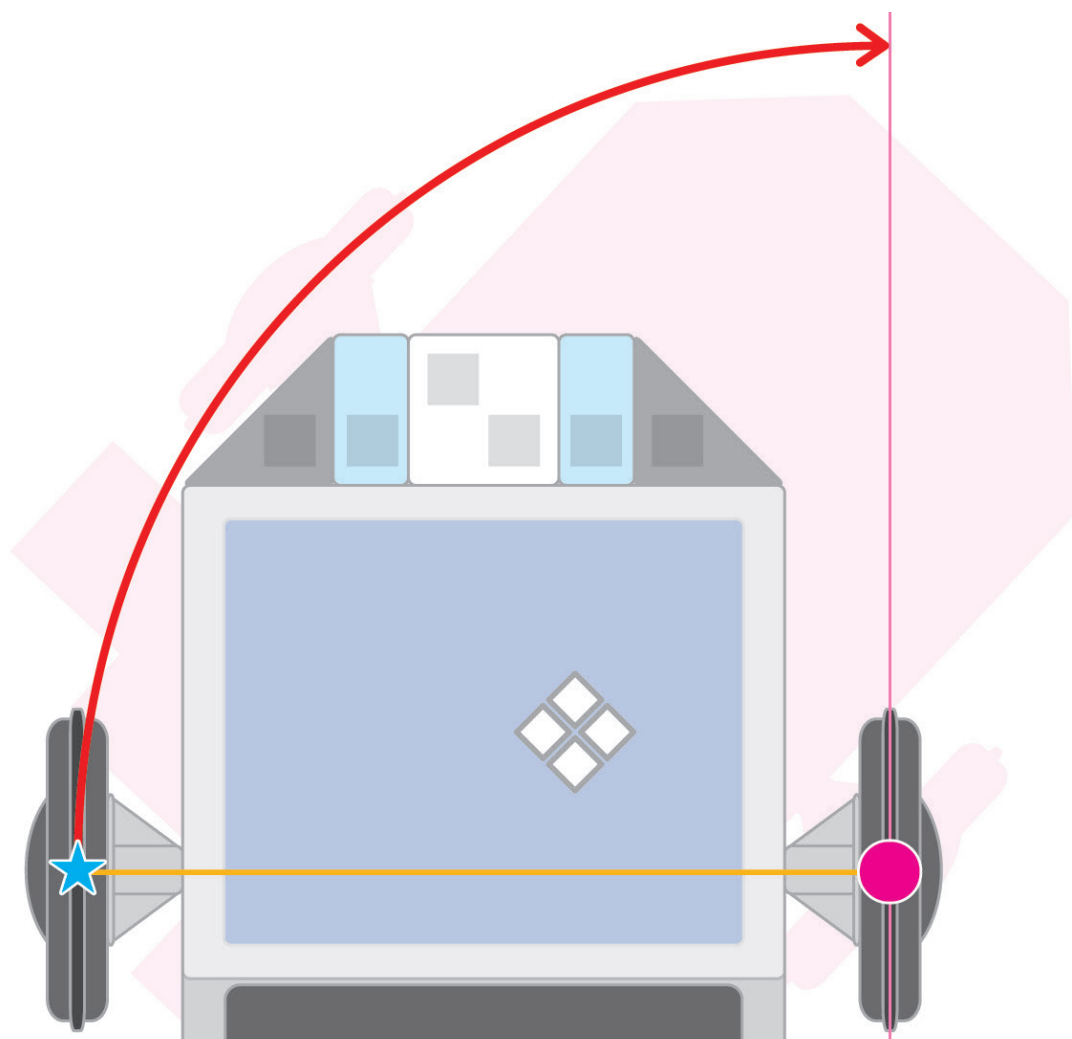


③右折

右折は左のDCモーターだけを正転させることで実現できます。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。

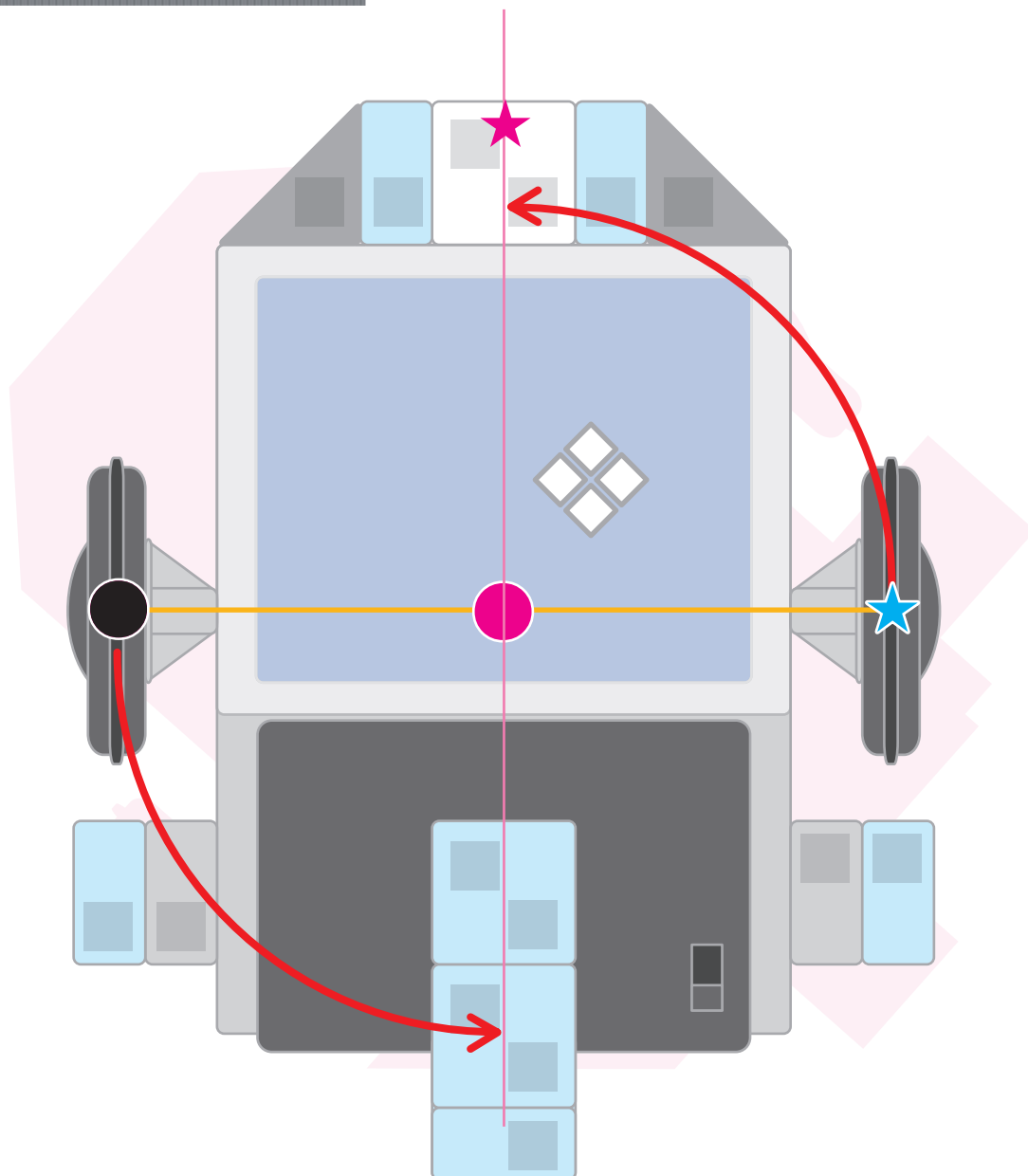


④左回転

左回転は右のDCモーターを正転させ、左のDCモーターを逆転させることで実現できます。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。

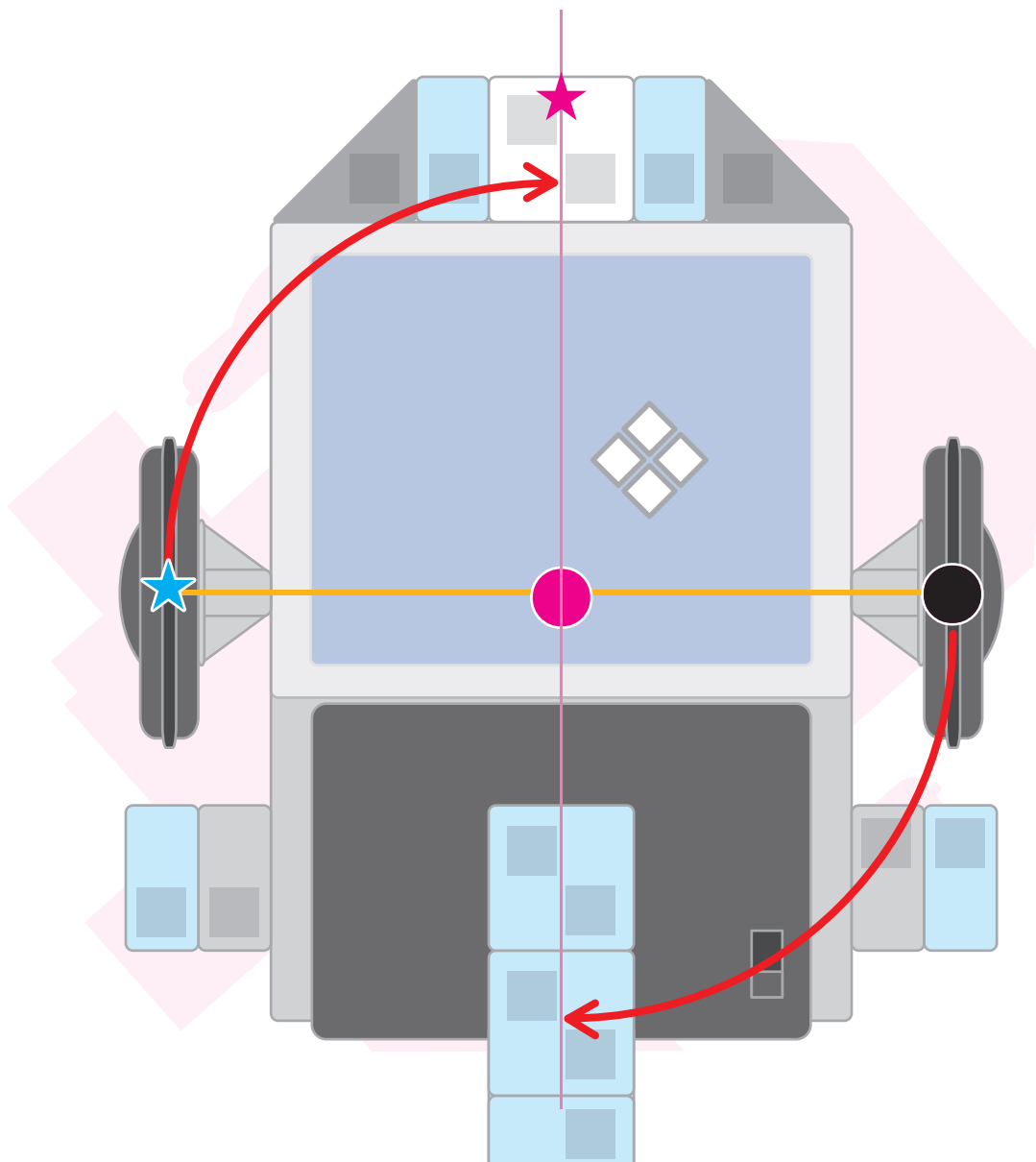


⑤右回転

右回転は左の DC モーターを正転させ、右の DC モーターを逆転させることで実現できます。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。

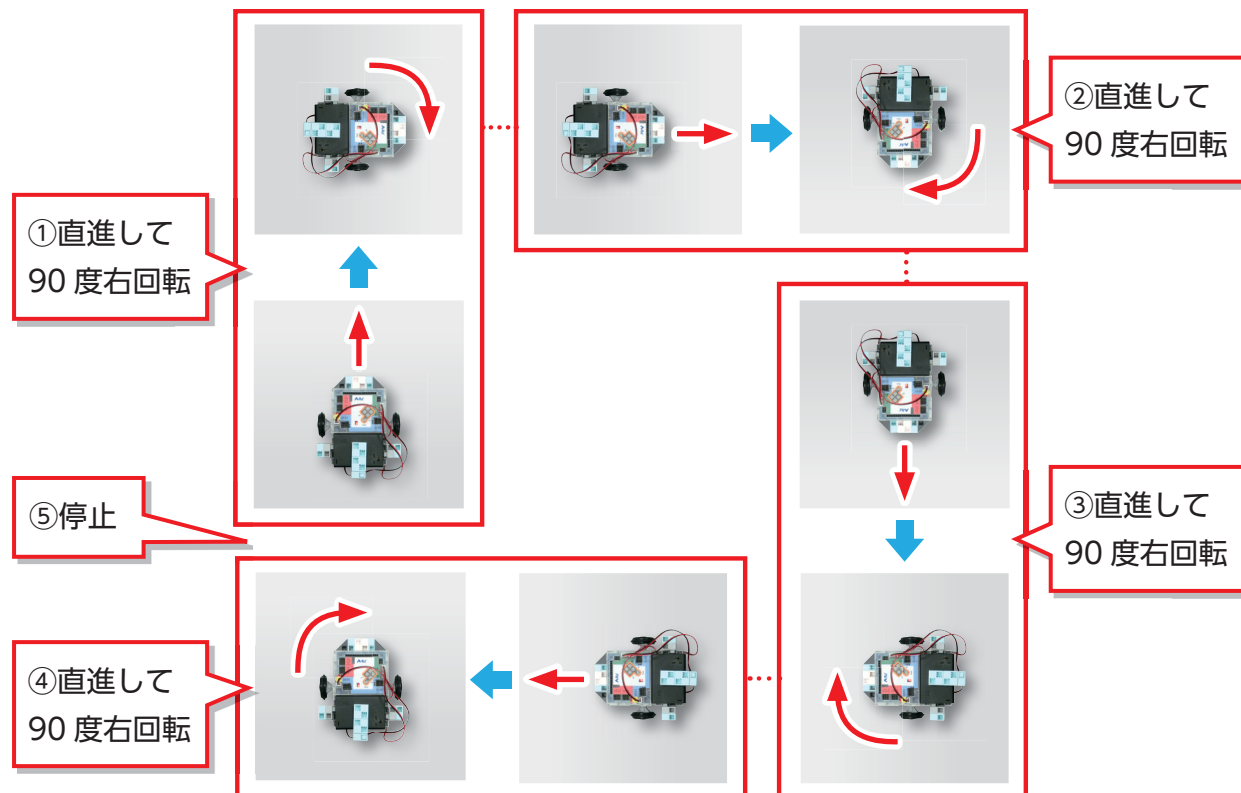


左のプログラムをつくり、「待つ」ブロックの時間を調整しましょう。



2. 繰り返しのプログラム

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。

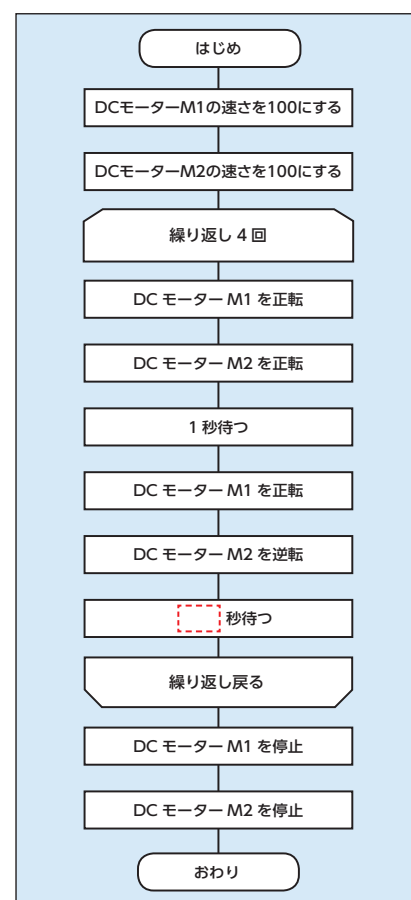


①フローチャート



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25ページで確認した秒数を参考にしてください。




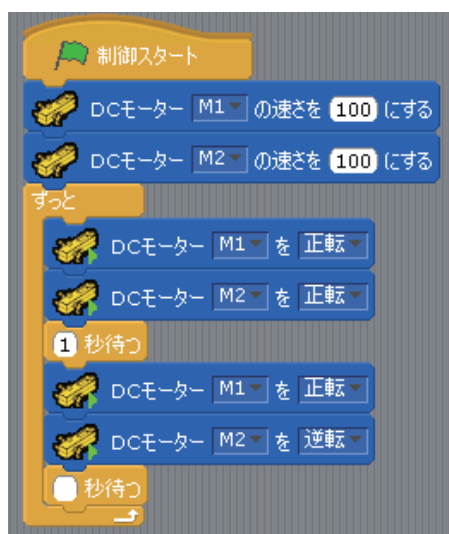
②プログラミング

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。

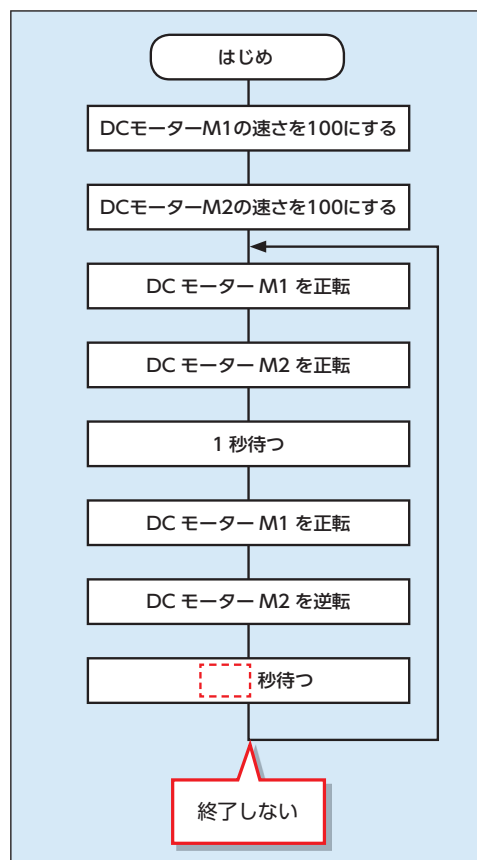


③無限に繰り返すプログラム

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します。※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。



第4章

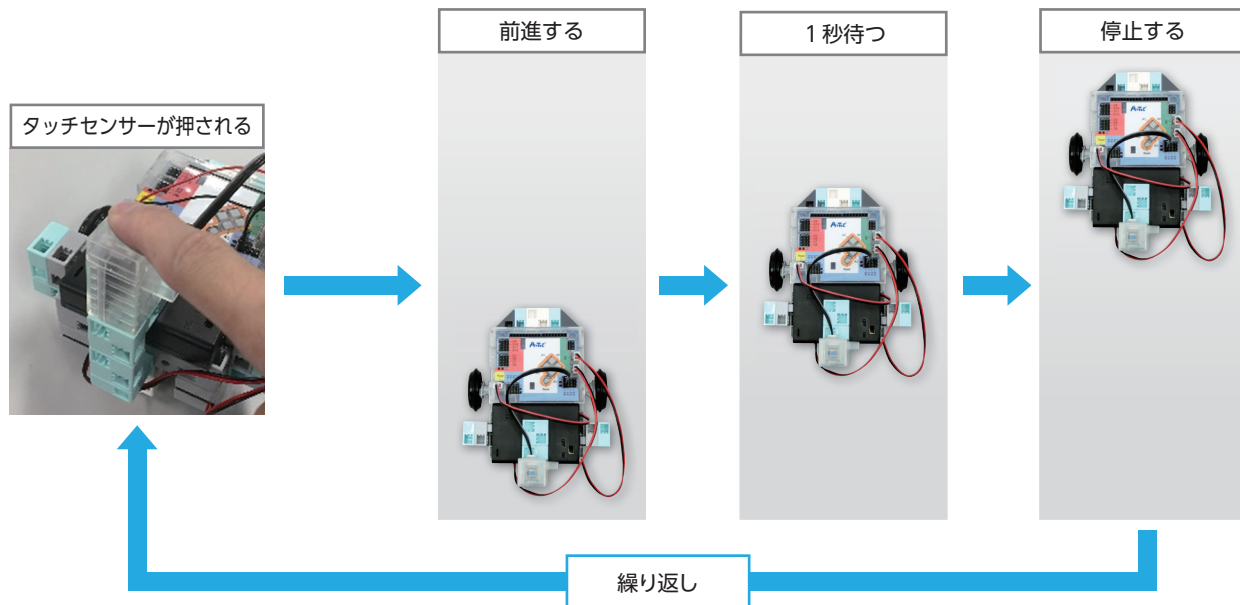
センサーによる車の操縦

<学習内容>

- 条件分岐
- タッチセンサー

1. 条件分岐のプログラム

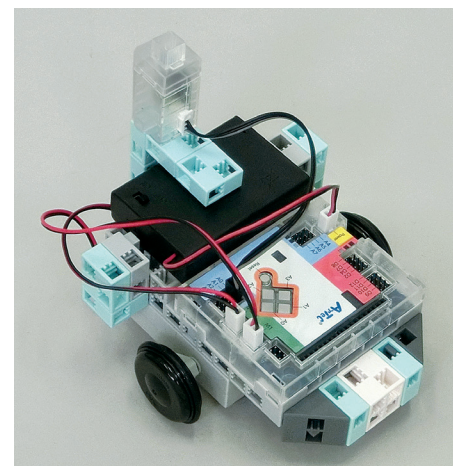
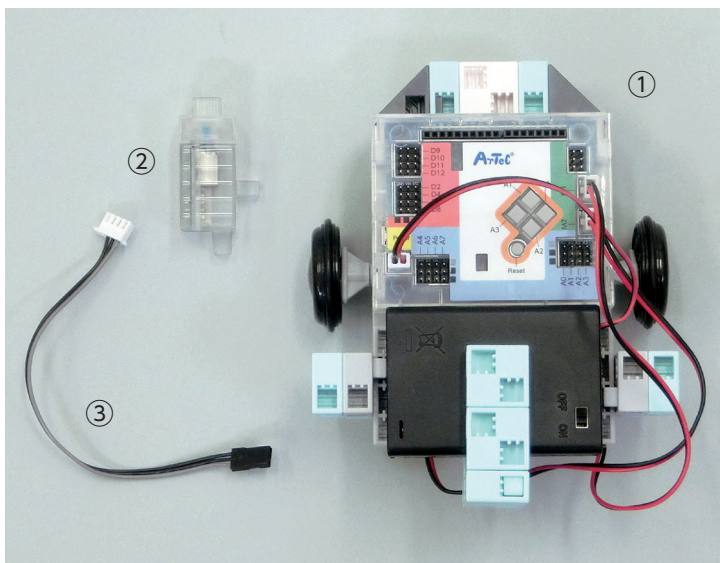
タッチセンサーを用いて、ボタンが押されたときに1秒前進する車をつくりましょう。



①組み立て

62ページの組み立て説明を見て、車にタッチセンサーを取り付けましょう。

組み立て準備



②入出力設定

A0 にタッチセンサーが追加されたので、A0 にチェックをつけて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。テストモードを実行し、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」はテストモード実行中に表示され、センサーの値がリアルタイムで確認できます。

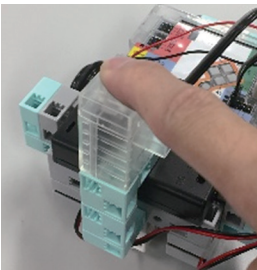
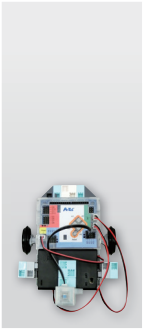

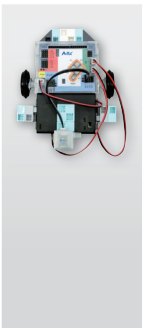
| タッチセンサーが 押されているときの数値 | タッチセンサーが 押されていないときの数値 |
|-------------------------|--------------------------|
| 0 | 1 |

タッチセンサーの数値は「調べる」カテゴリの「タッチセンサー A0 の値」で知ることができます。

また、「タッチセンサー A0 の値」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

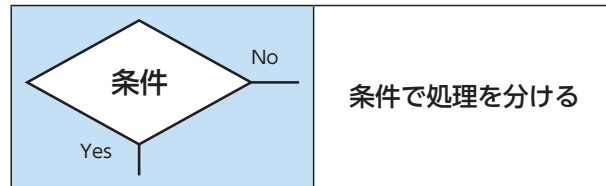
動作とプログラムの関係を整理しましょう。

| 順番 | 動作 | プログラム |
|----|---|--|
| ① | DC モーターの速さを決める | 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 |
| ② | タッチセンサーが押される  | タッチセンサー A0 の値 = 0 |
| ③ | 前進する  | DCモーター M1 を正転 DCモーター M2 を正転 |
| ④ | 1秒待つ  | 1秒待つ |
| ⑤ | 停止する  | DCモーター M1 を停止 DCモーター M2 を停止 |

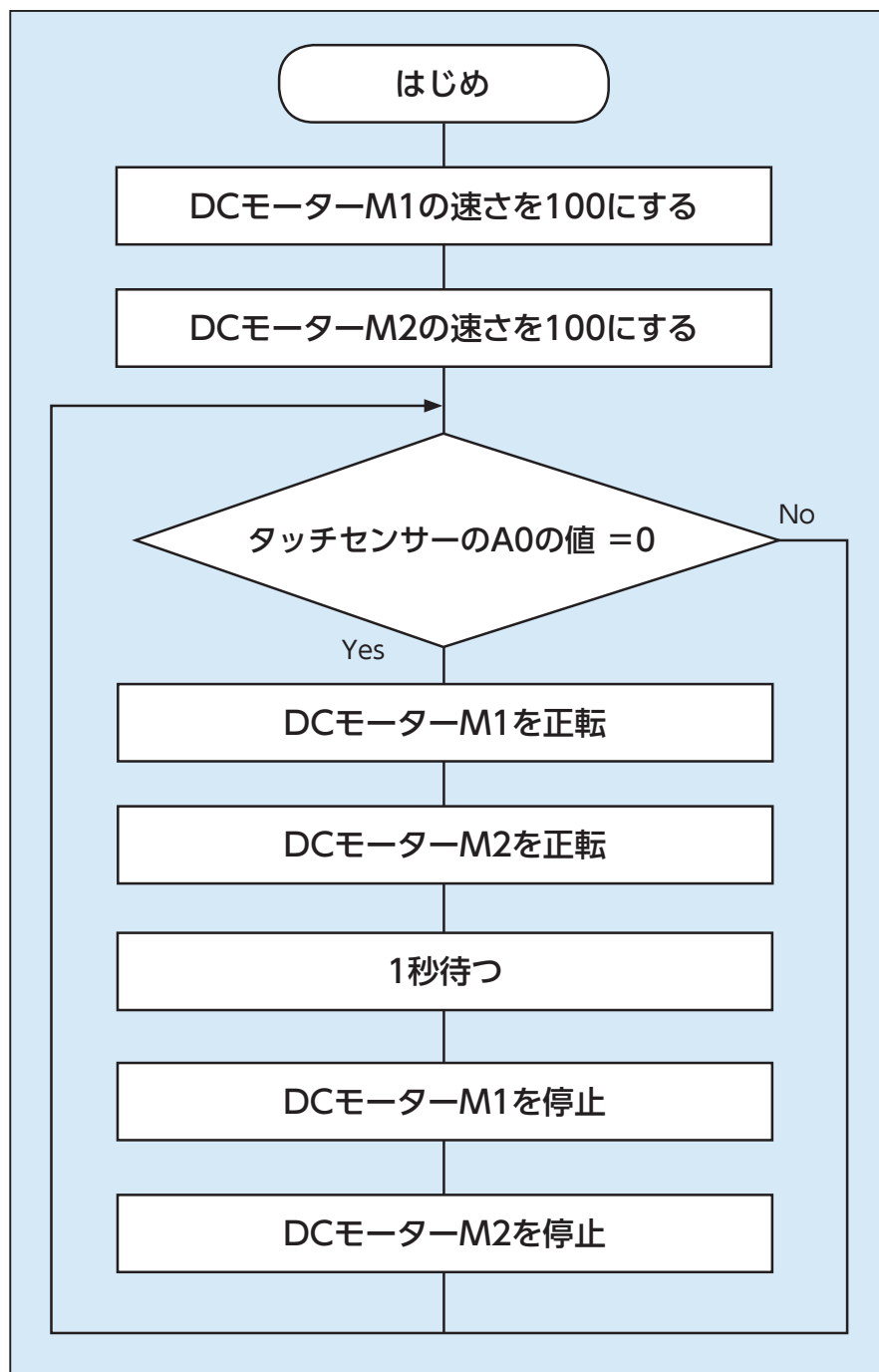
②～⑤を
ずっと繰り返す

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。

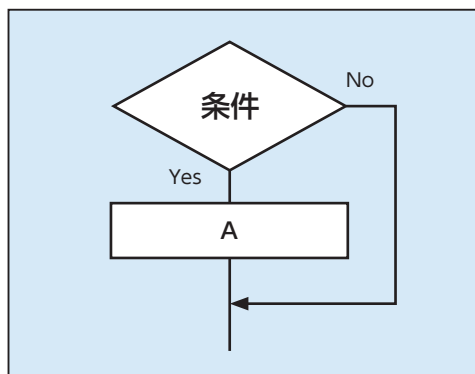


31 ページで整理した動作を参考にフローチャートをまとめましょう。フローチャートに「終了」が存在せずに繰り返しているのは、タッチセンサーが押されているかを常に確認するためです。

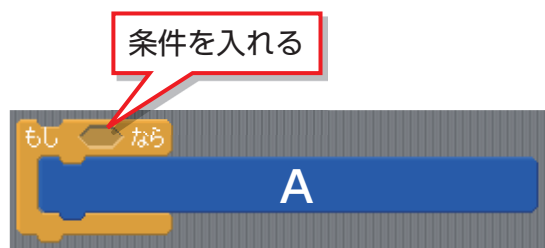


⑥プログラム作成

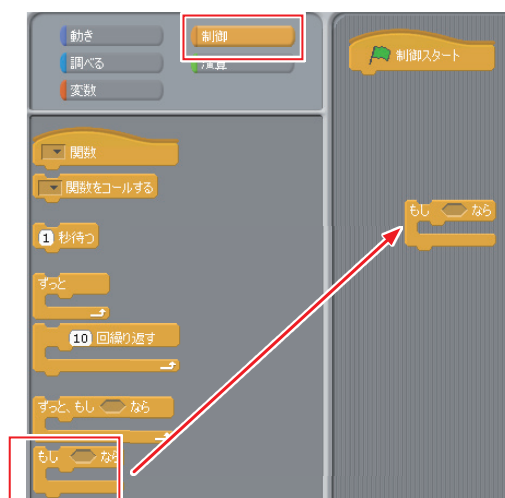
条件で処理を分けるフローチャートは次のプログラムと同じです。



=



は「制御」カテゴリーにあります。



条件は「演算」カテゴリーの  と  を組み合わせてつくることができます。



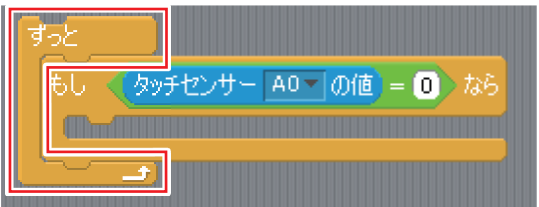

作成した条件は  の空欄に入れて使うことができ、

条件が成り立つときだけ囲まれたブロックが処理されるようになります。



⑦動作確認

つくったプログラムで想定通りに動作しない場合、次の点を確認して修正しましょう。

| 「ずっと」のブロックを使っているか | 車が前進したあとに 停止するプログラムを入れているか |
|---|--|
|  |  |

ステップ実行機能でプログラムの流れを可視化する

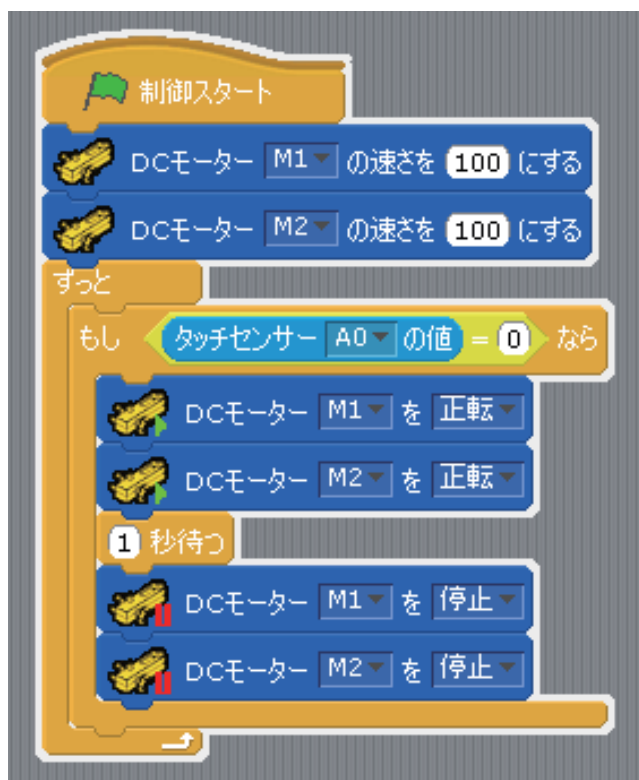
条件で処理を分けると、プログラムが少し複雑になります。複雑なプログラムを作成する際に間違えてしまうことは頻繁に起こることであり、その時に誤りを素早く発見し修正すること（＝「デバッグ」）が大切です。



どのような流れでプログラムが動いているのかに注目することは、誤りの発見に有効です。プログラムの処理の流れを可視化する機能として、「ステップ実行機能」があります。

ステップ実行を開始してプログラムを実行すると、そのとき処理しているブロックが黄色く光るようになります。

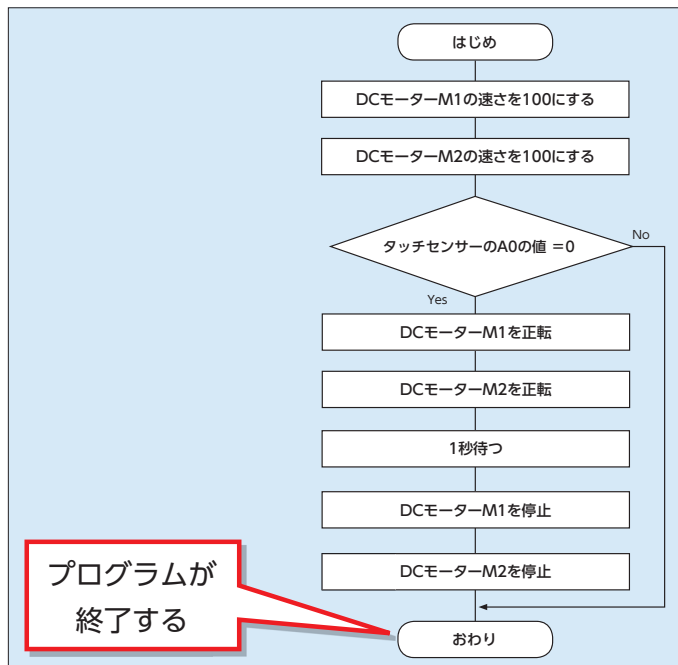
デバッグを行うときは、ステップ実行機能を活用しましょう。



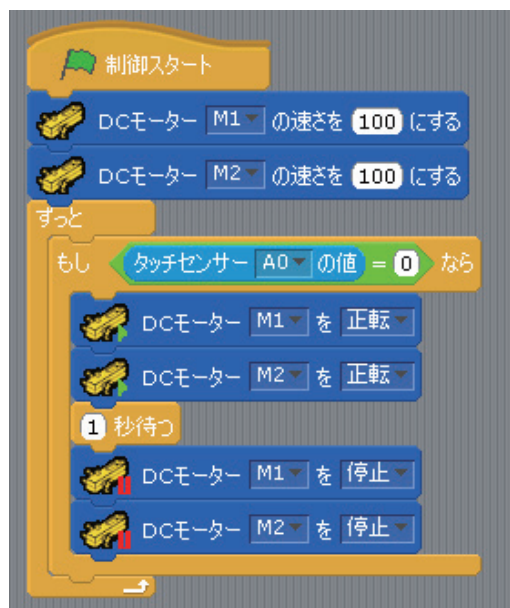
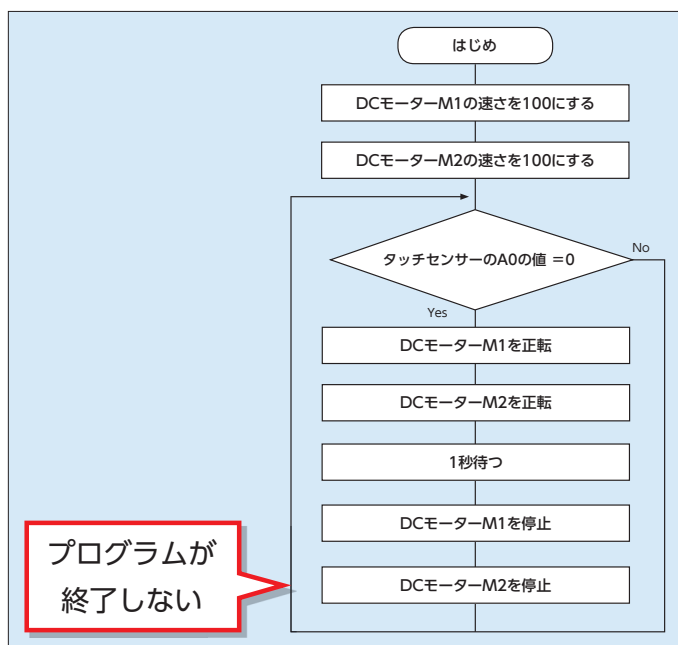
※ステップ実行時はプログラムの処理が極端に遅くなることに注意してください。

「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。



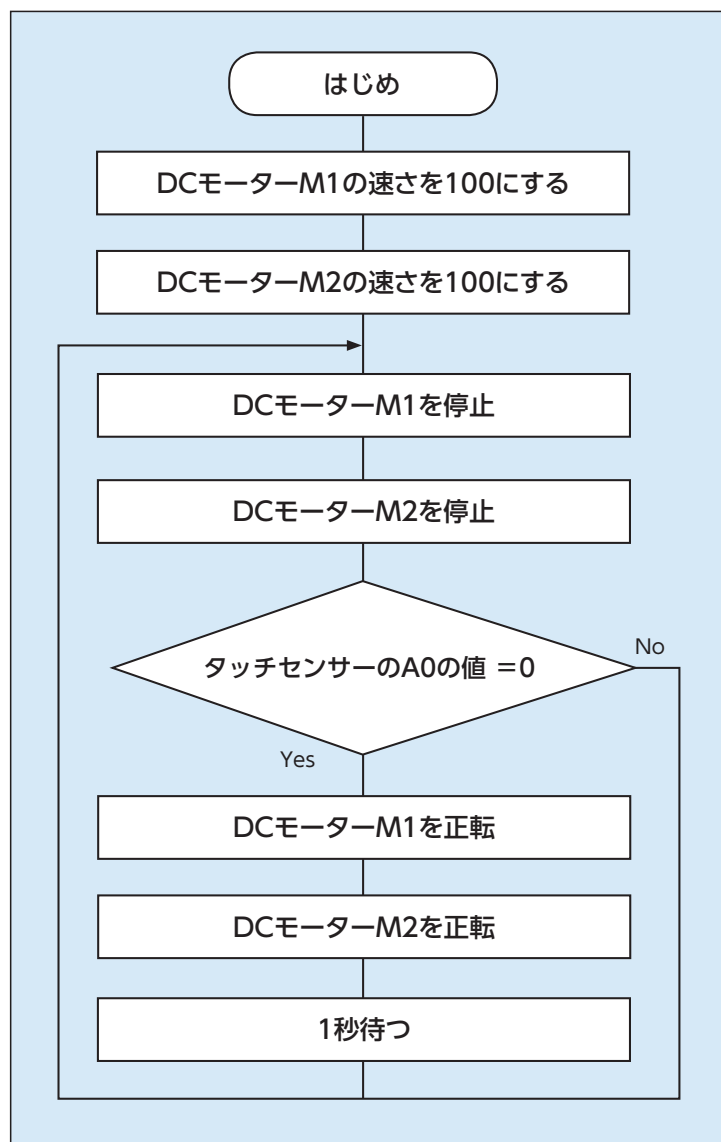
このプログラムを動作させると、タッチセンサーを押しても全く反応しなくなります。これは、**プログラムが非常に高速で処理されることで、プログラムを動作させた瞬間にプログラムが終了してしまうから**です。「ずっと」のブロックを入れることで、タッチセンサーの値を常に確認するようになるため想定通りの動作をするようになります。



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。

違うプログラムで同じ動きを実現する

以下のようなプログラムでも同じ動作を実現することができます。



プログラムの作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作ることが大切です。

第5章

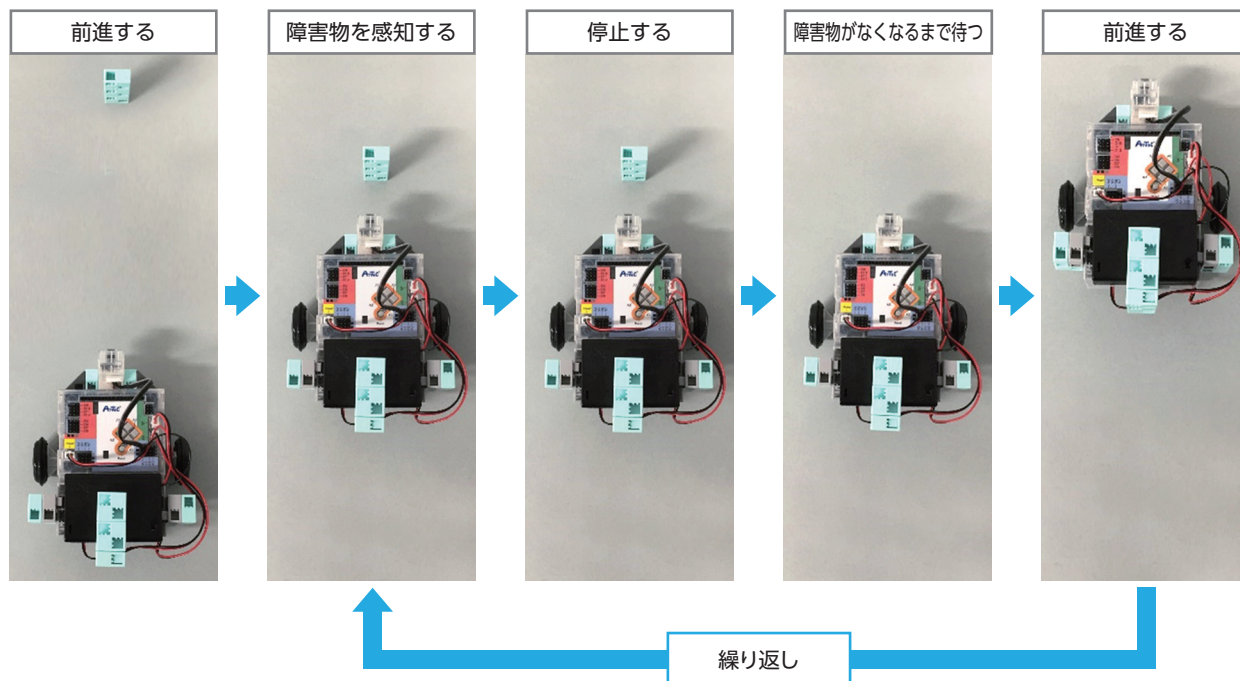
衝突回避カーの製作

<学習内容>

- 赤外線フォトリフレクタ

1. 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。

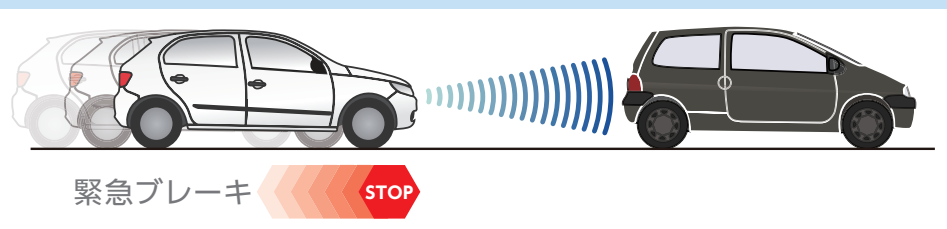


自動車の運転を補助するシステム

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。

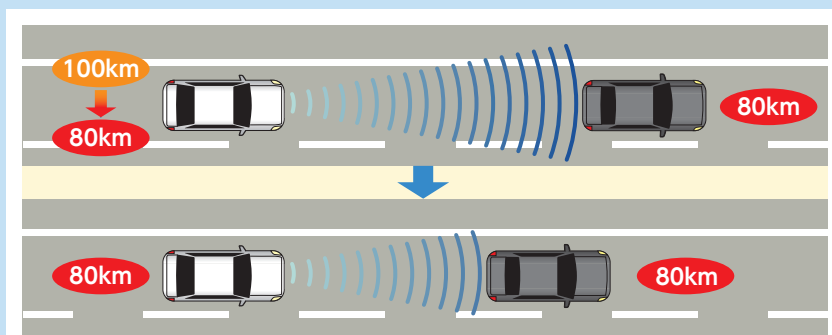
自動ブレーキシステム

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。



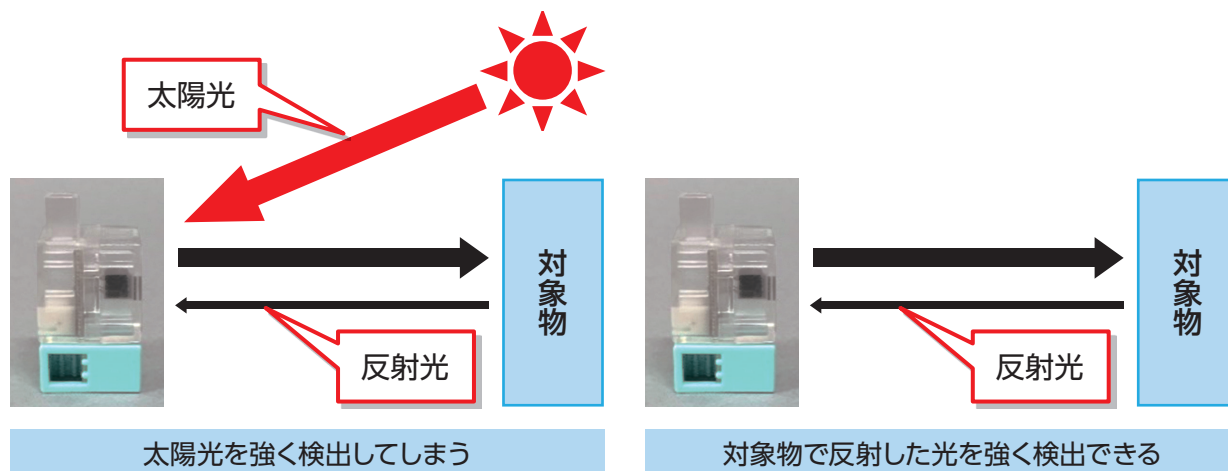
①赤外線フォトリフレクタとは

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。

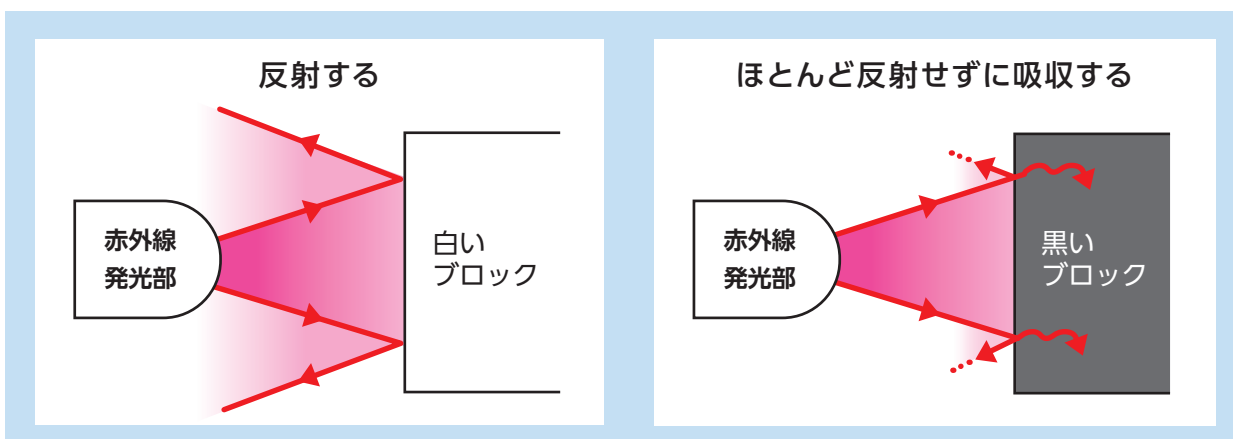


赤外線フォトリフレクタの使い方

・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。



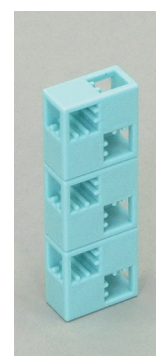
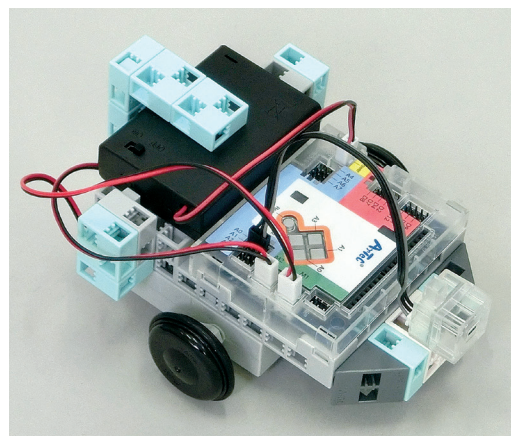
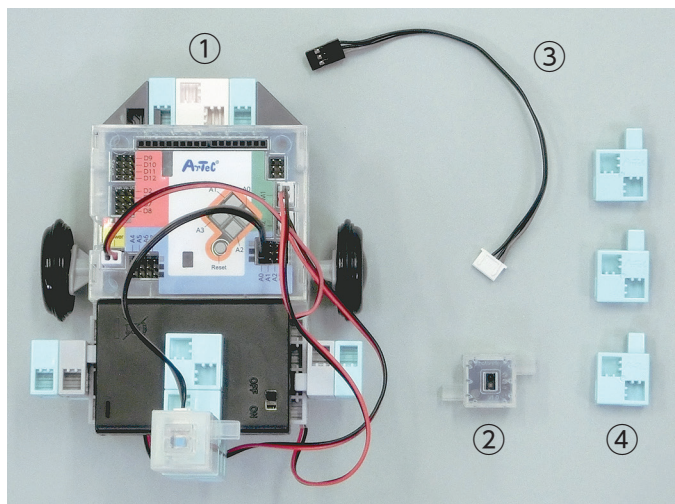
・黒系の色は赤外線を吸収しやすく反射しにくいいため、反射光の検出量が少なくなります。



②組み立て

62ページの組み立て説明を見て、4章で作った車のタッチセンサーを赤外線フォトリフレクタに付け替えて、衝突回避カーを組み立てましょう。また61ページを見て、障害物となるブロックも組み立てましょう。

組み立て準備



- ① 車(第4章のもの) 1
- ② 赤外線フォトリフレクタ 1
- ③ センサーコード 1
- ④ ハーフブロック 薄水 3

③入出力設定

M1、M2にDCモーター、A0に赤外線フォトリフレクタとなるように設定しましょう。

入出力設定

DCモーター

☒ M1 ☒ M2

サーボモーター

☐ D2 ☐ D4 ☐ D7 ☐ D8
☐ D9 ☐ D10 ☐ D11 ☐ D12

ボタン

☐ A0 ☐ A2
☐ A1 ☐ A3

センサー/LED/ブザー

☒ A0 赤外線フォトリフレクタ ☐ A4 光センサー
☐ A1 光センサー ☐ A5 光センサー
☐ A2 光センサー ☐ A6 光センサー
☐ A3 光センサー ☐ A7 光センサー

チェックを全て外す

OK

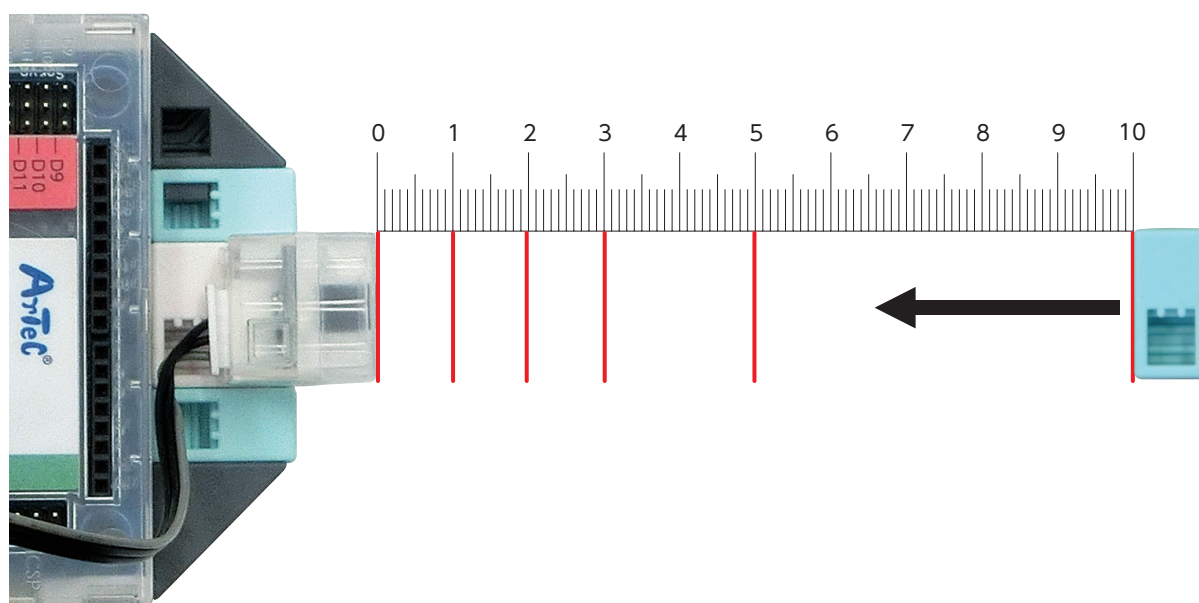
キャンセル

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 薄水）があるとときの値を確認しましょう。

太陽光を背に向けた状態で調べると正しい数値を確認しやすくなります。

| センサー・ボード | |
|------------------|---|
| [A0] 赤外線フォトリフレクタ | 4 |
| [A1] 未接続 | 0 |
| [A2] 未接続 | 0 |
| [A3] 未接続 | 0 |
| [A4] 未接続 | 0 |
| [A5] 未接続 | 0 |
| [A6] 未接続 | 0 |
| [A7] 未接続 | 0 |

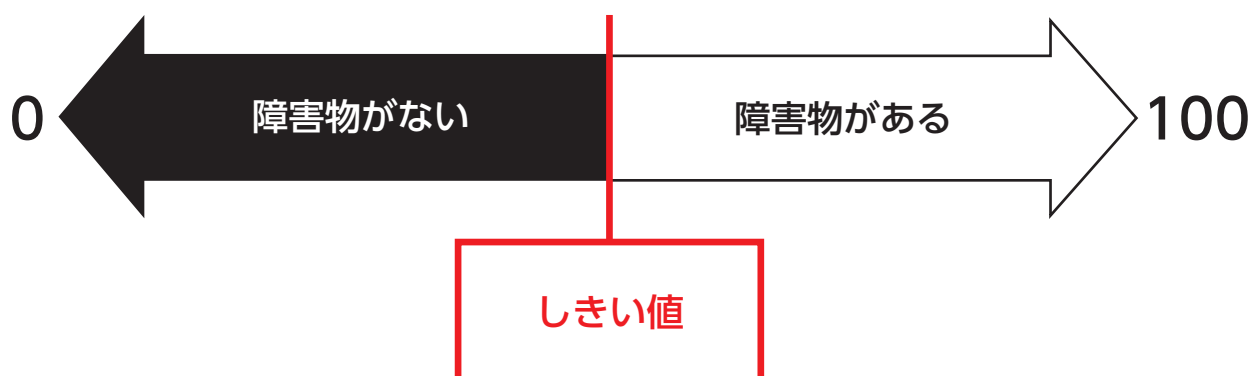


| 距離 | 0cm | 1cm | 2cm | 3cm | 5cm | 10cm |
|----|-----|-----|-----|-----|-----|------|
| 値 | | | | | | |

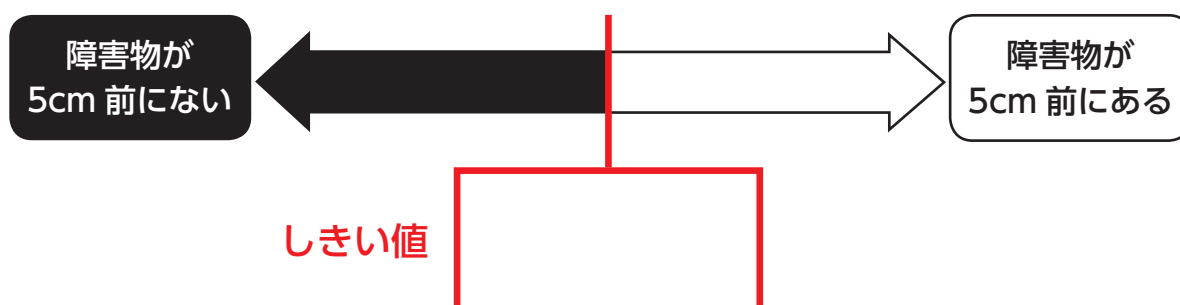
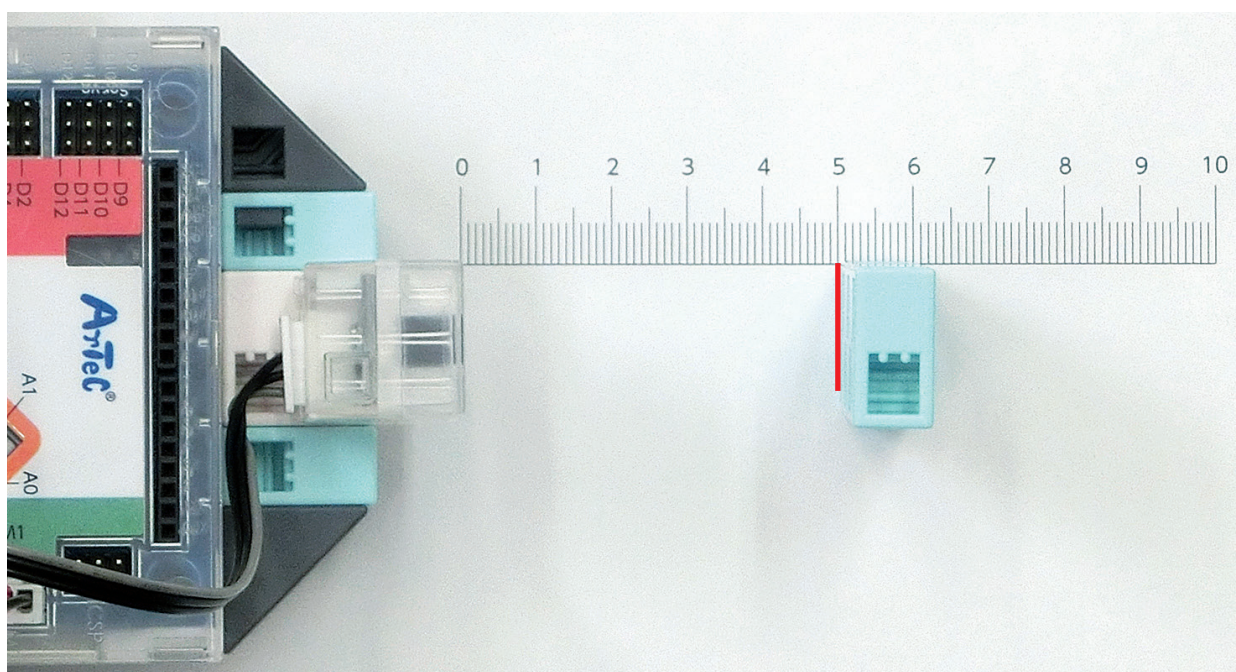
※赤外線フォトリフレクタの個体差で、同じ距離でも数値が異なる場合があります。

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。

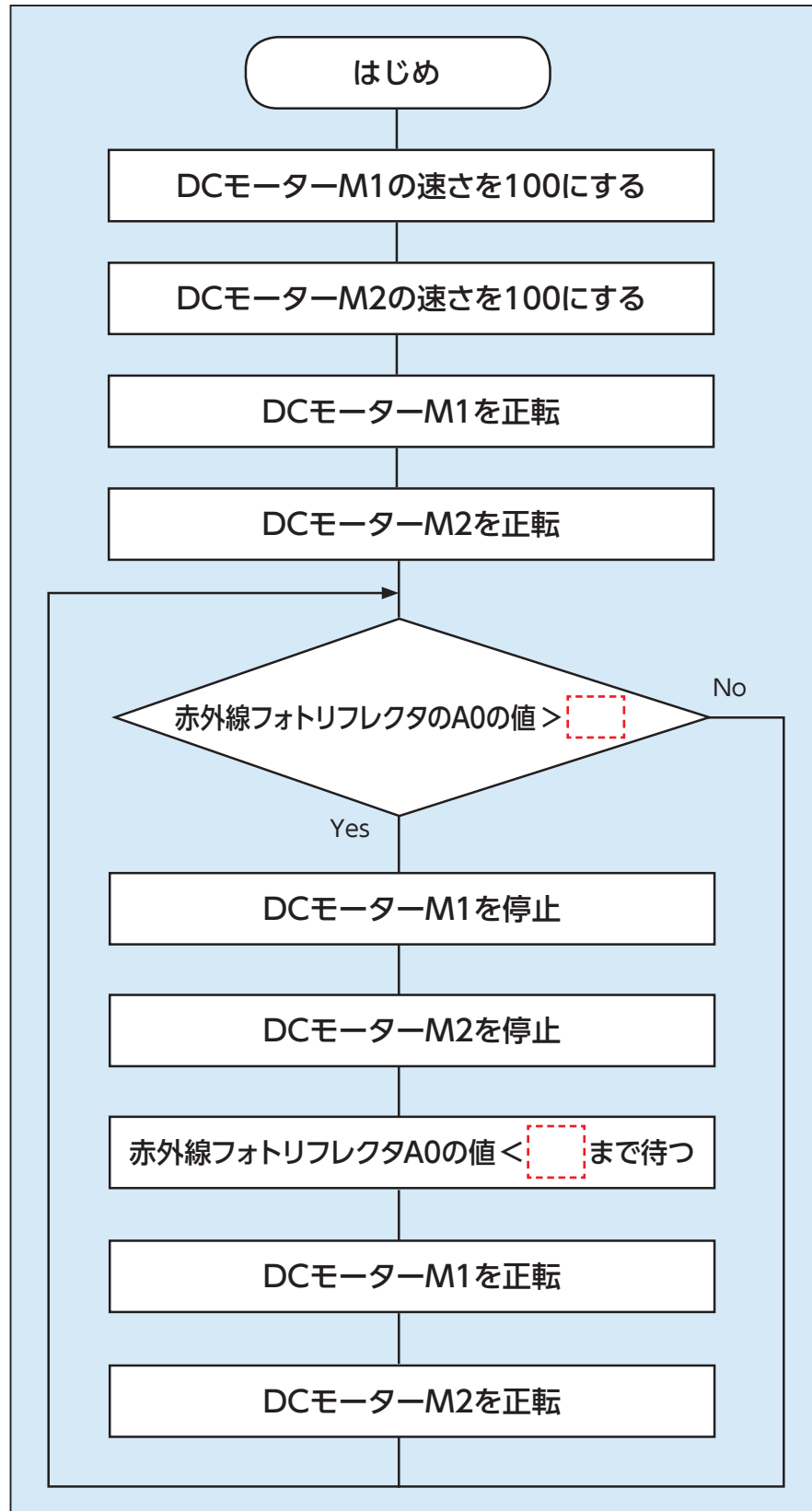


今回は車の前5cmに障害物があるときとないときを区別するようにしきい値を決めましょう。





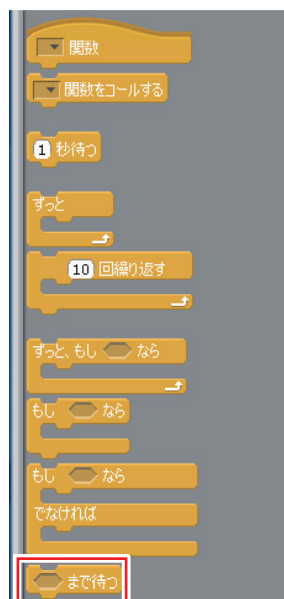
⑥フローチャート

38 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。



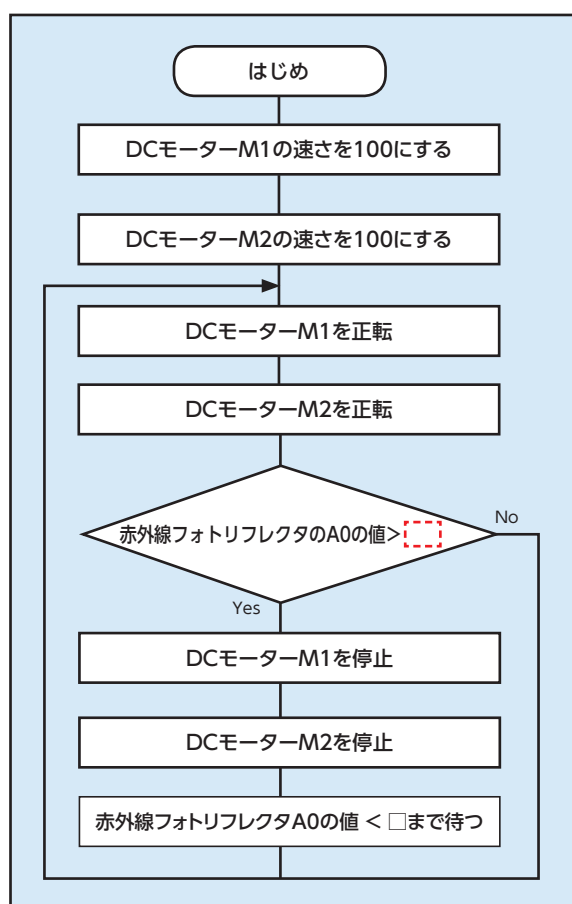
⑦プログラミング

「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。



演習 1

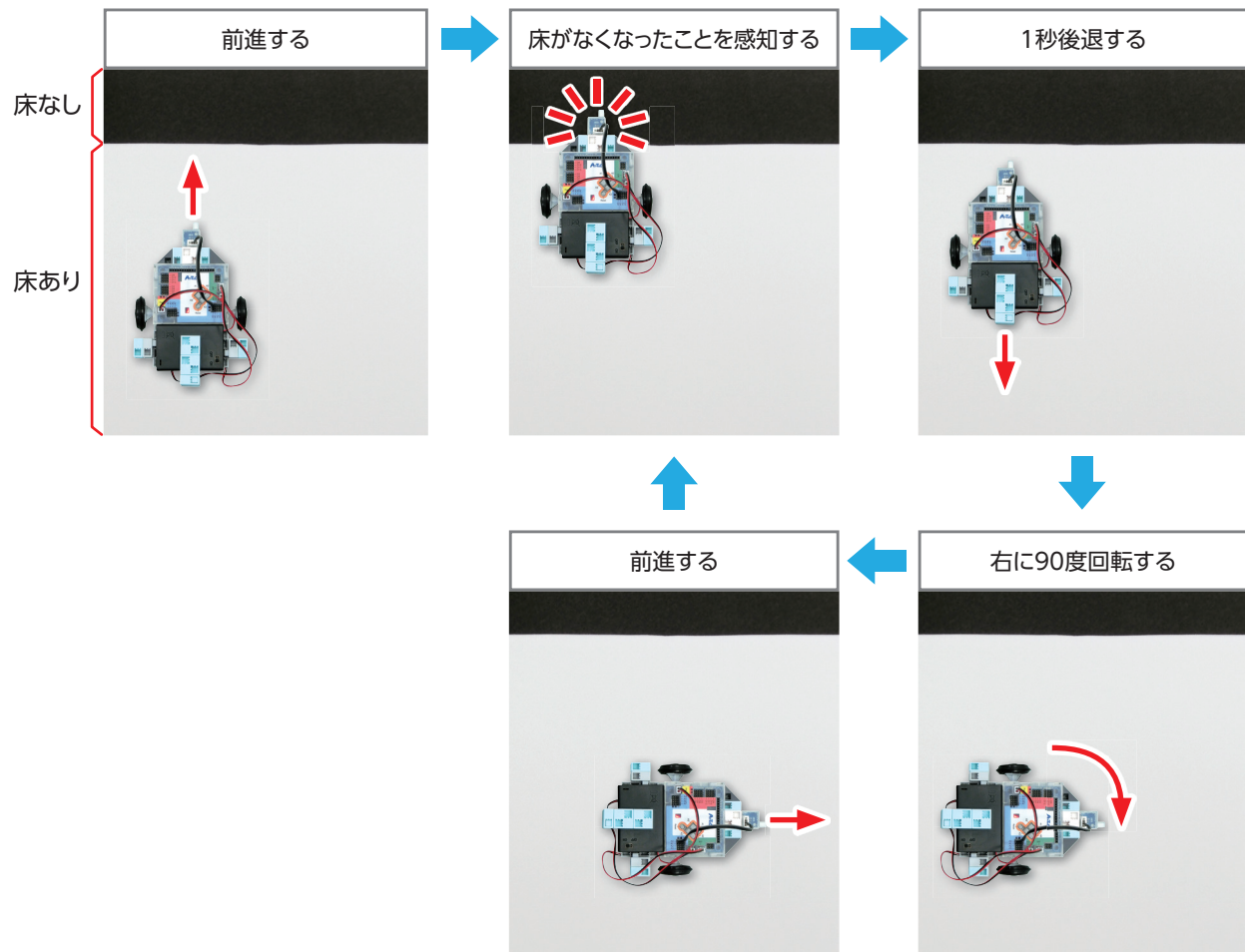
落下回避カーの製作

<学習内容>

- 赤外線フォトリフレクタの応用

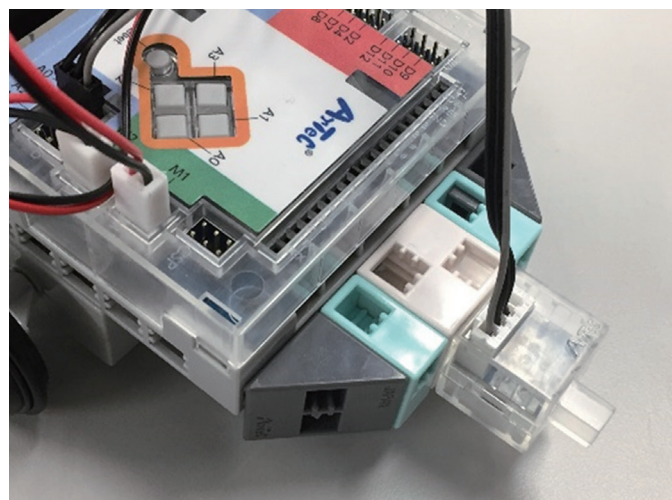
1. 落下回避カーの製作

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。

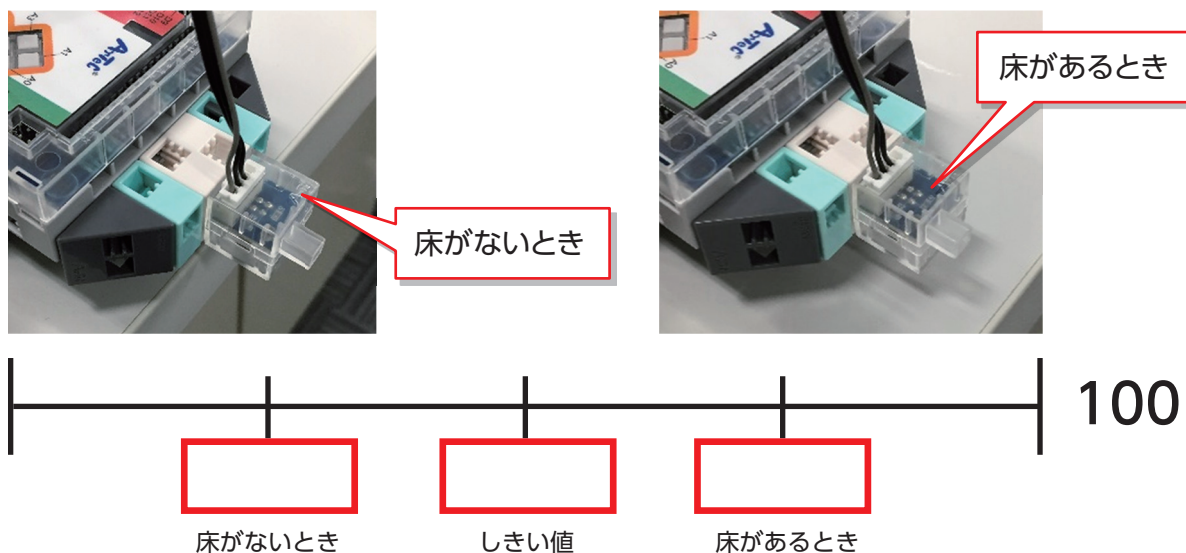


②入出力設定

M1とM2にDCモーター、A0に赤外線フォトリフレクタとなるように設定しましょう。

③しきい値の決定

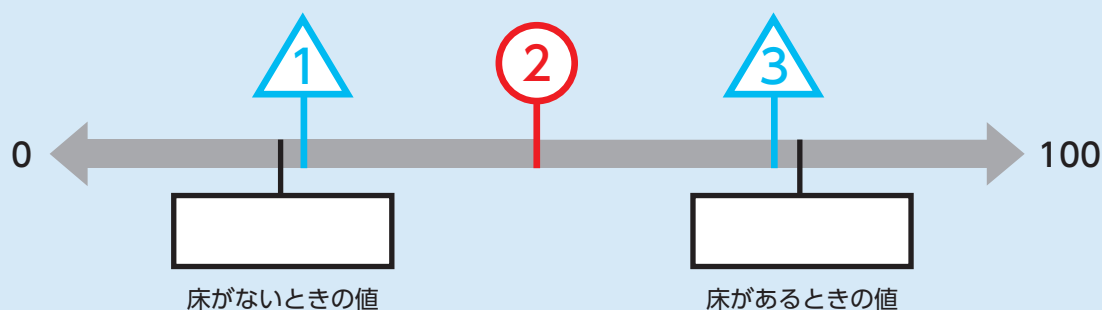
テストモードで床があるときとないときの赤外線フォトリフレクタの値を確認して、しきい値を決めましょう。



しきい値の決定

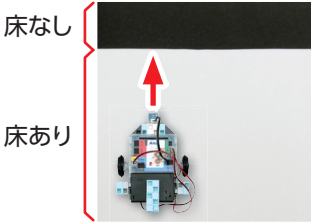
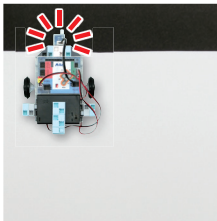
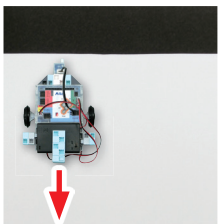
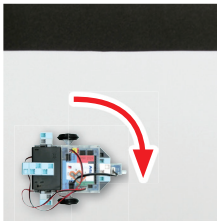
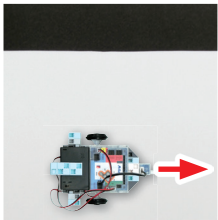
しきい値を ① や ③ のように調べたどちらかの値の近くに決めてしまうと、赤外線フォトリフレクタの値がそこから少し変わるだけで、ブロックの有無を区別してしまいます。

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

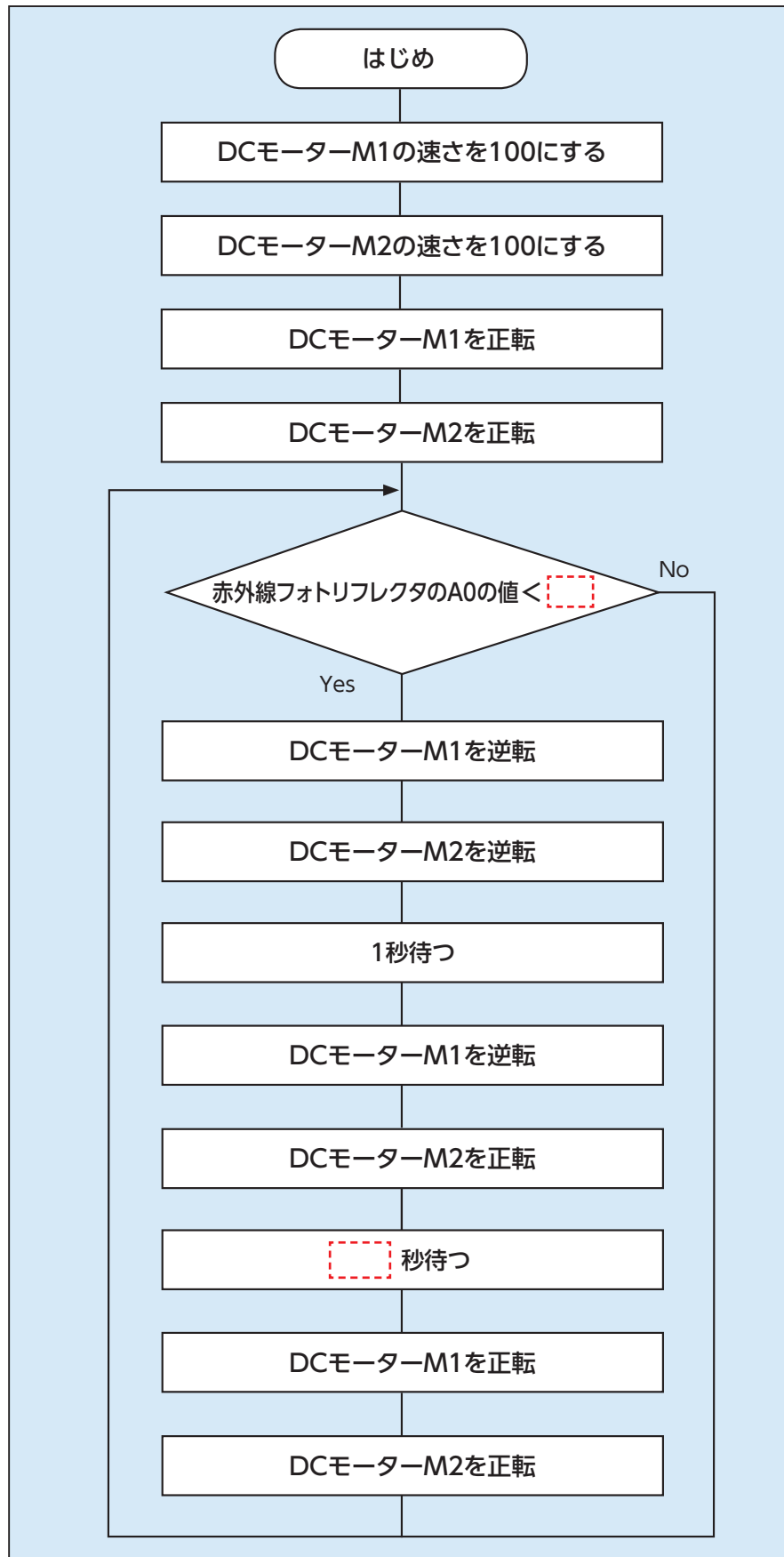
動作とプログラムの関係を整理しましょう。

| 順番 | 動作 | プログラム |
|--|---|--|
| <div>①</div> <div>②</div> <div>③</div> <div>④</div> <div>⑤</div> <div>②～⑤を ずっと繰り返す</div> | <p>前進する</p>  | <p>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 を正転 DCモーター M2 を正転</p> |
| | <p>床がなくなったことを感知する</p>  | <p>赤外線フォトリフレクタ A0 の値 < しきい値</p> |
| | <p>1秒後退する</p>  | <p>DCモーター M1 を逆転 DCモーター M2 を逆転 1秒待つ</p> |
| | <p>右に90度回転する</p>  | <p>DCモーター M1 を逆転 DCモーター M2 を正転 □ 秒待つ</p> |
| | <p>前進する</p>  | <p>DCモーター M1 を正転 DCモーター M2 を正転</p> |

※右に90度回転させる秒数は25ページで調べたものを参考にしてください。

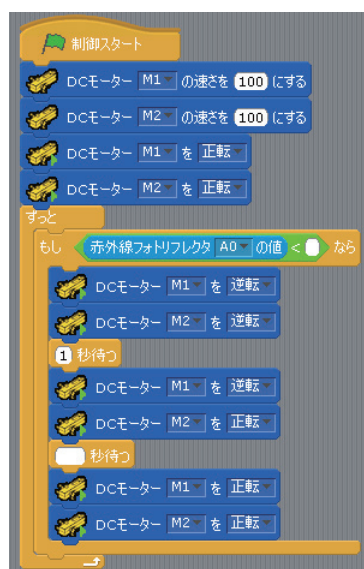
⑤フローチャート

48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。



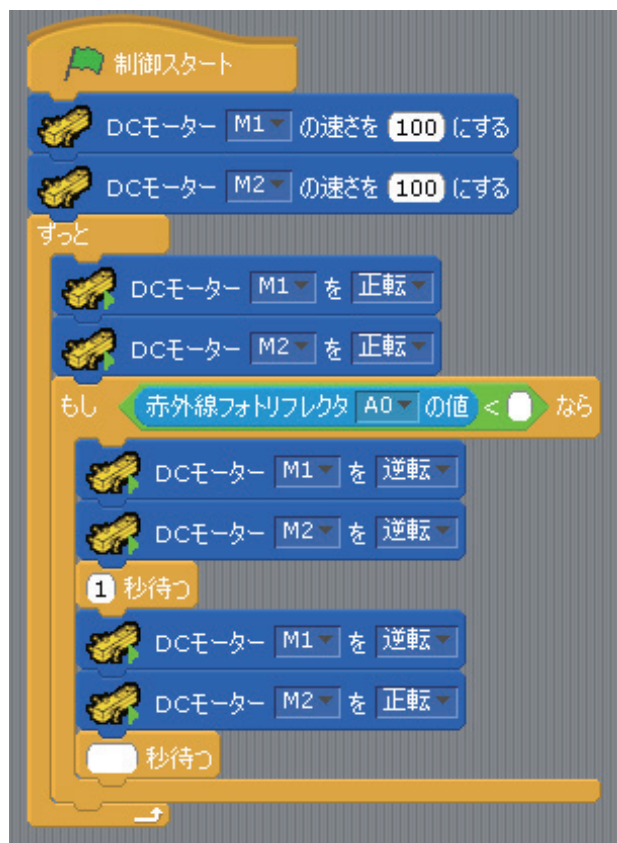
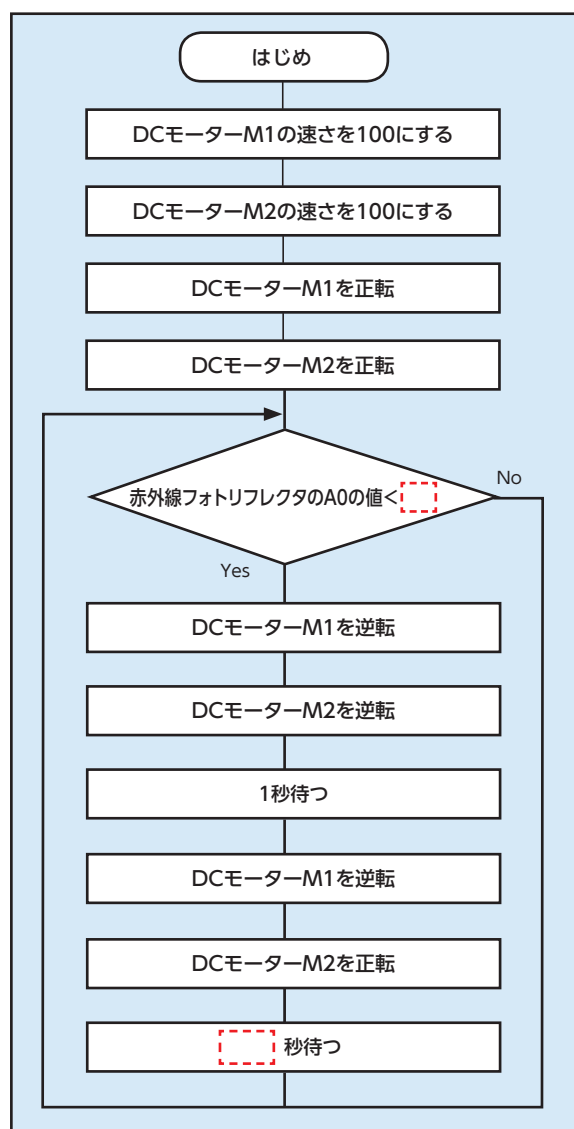
⑥プログラミング

49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。



演習 2

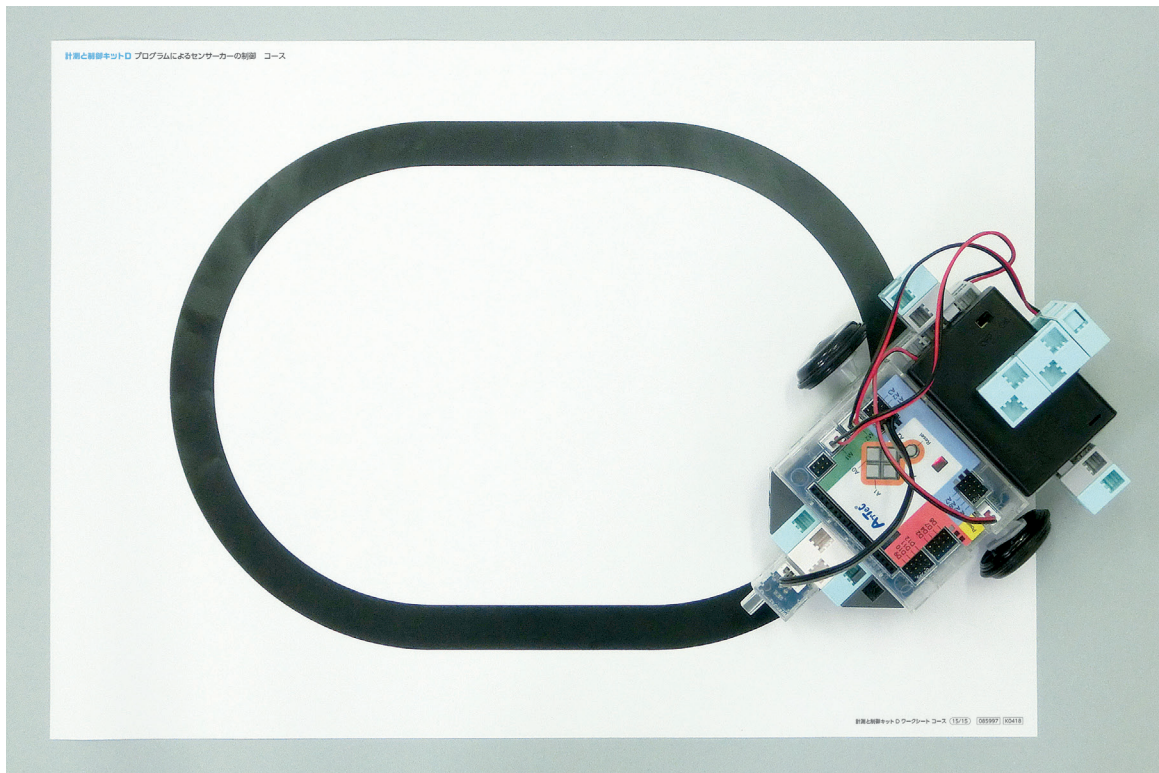
ライトレースカーの 製作

<学習内容>

- 赤外線フォトリフレクタの応用

1、ラインレースカーの製作

色によって赤外線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。



車は演習1でつくっ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、A0に赤外線フォトリフレクタとなるように設定しましょう。

入出力設定

| DCモーター | サーボモーター | ボタン |
|---|---|--|
| <input checked="" type="checkbox"/> M1 <input checked="" type="checkbox"/> M2 | <input type="checkbox"/> D2 <input type="checkbox"/> D4 <input type="checkbox"/> D7 <input type="checkbox"/> D8 <input type="checkbox"/> D9 <input type="checkbox"/> D10 <input type="checkbox"/> D11 <input type="checkbox"/> D12 | <input type="checkbox"/> A0 <input type="checkbox"/> A2 <input type="checkbox"/> A1 <input type="checkbox"/> A3 |

センサー/LED/ブザー

| | |
|--|-----------------------------------|
| <input checked="" type="checkbox"/> A0 赤外線フォトリフレクタ | <input type="checkbox"/> A4 光センサー |
| <input type="checkbox"/> A1 光センサー | <input type="checkbox"/> A5 光センサー |
| <input type="checkbox"/> A2 光センサー | <input type="checkbox"/> A6 光センサー |
| <input type="checkbox"/> A3 光センサー | <input type="checkbox"/> A7 光センサー |

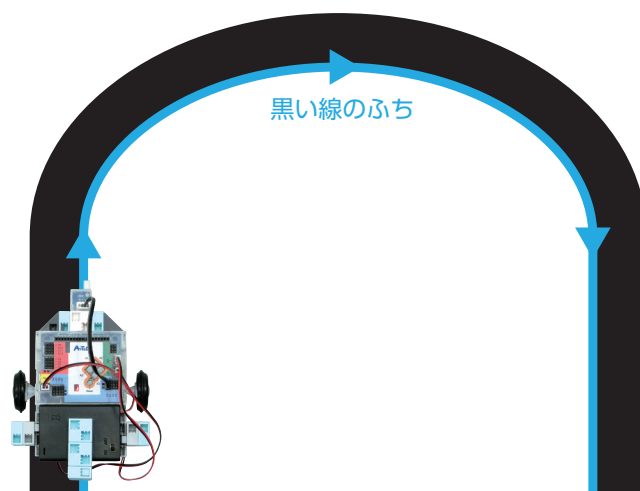
チェックを全て外す OK キャンセル

.....

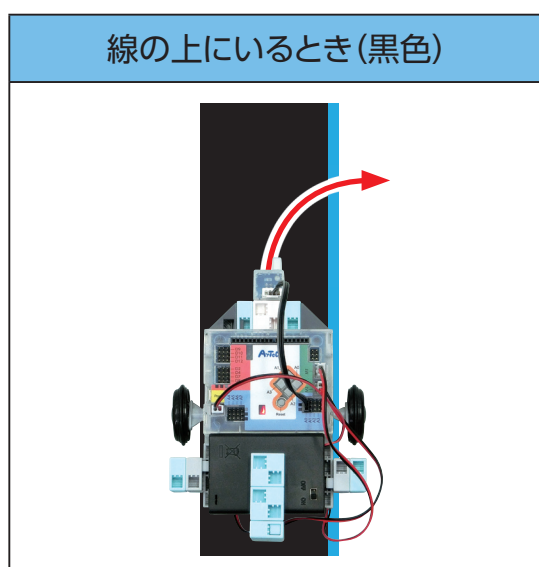


③ ライントレースカーの動作整理

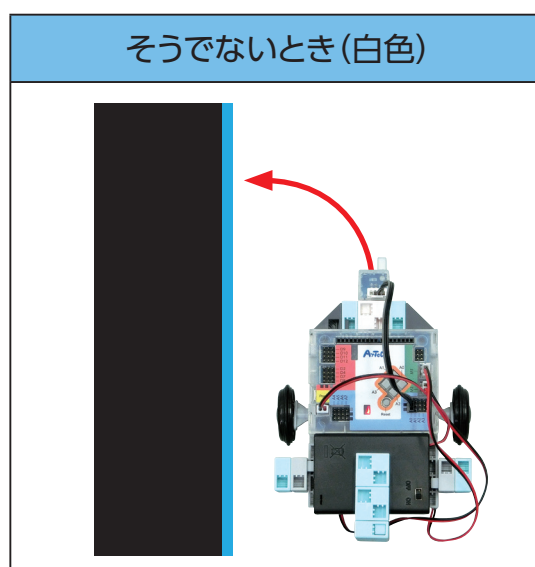
黒い線に沿って走らせる方法の一つとして、黒い線の上ではなく、黒い線のふち（黒い線とまわりとの境界線）に沿って走らせるものがあります。



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。

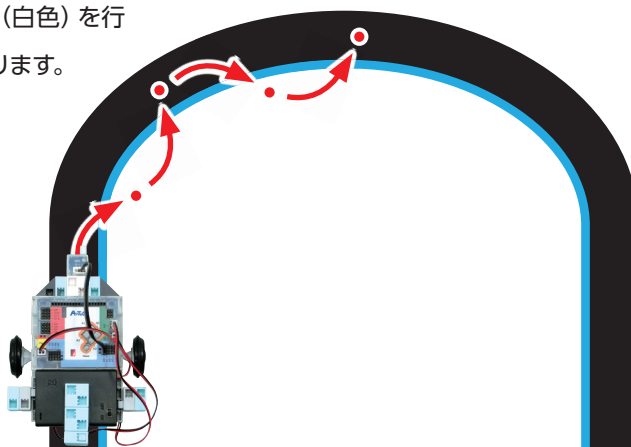


黒い線のふちに近づくように**右折する**

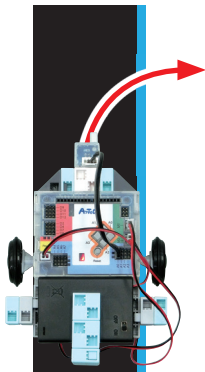
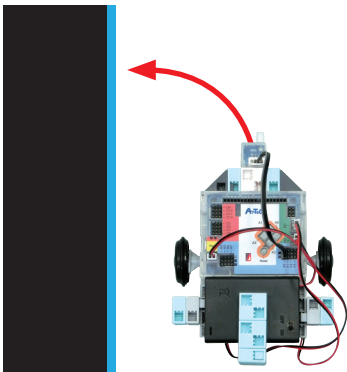


黒い線のふちに近づくように**左折する**

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ことができます。

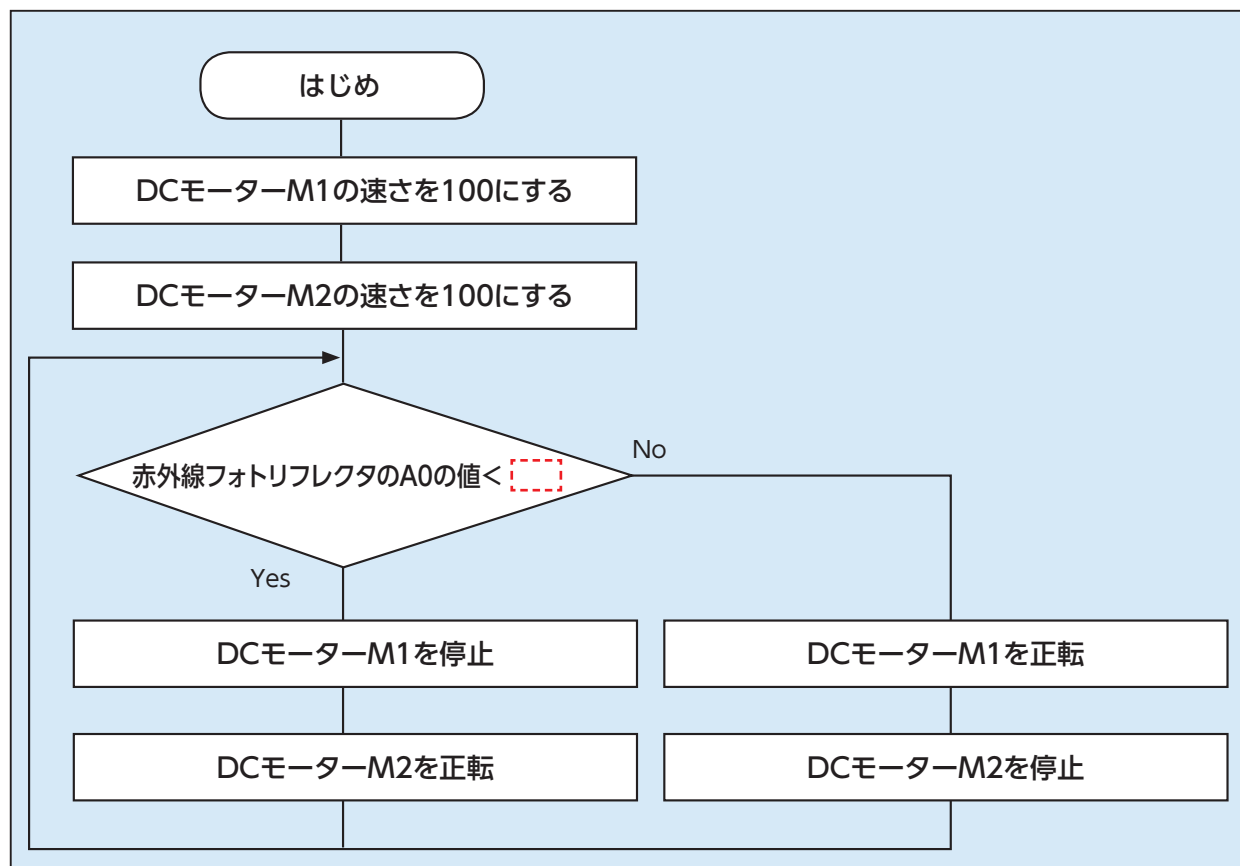


この動きをまとめると、以下のようになります。



| 状態 | 線の上にいるとき(黒色) | そうでないとき(白色) |
|-------|---|---|
| |  |  |
| 条件 | 赤外線フォトリフレクタ A0の値 くしきい値 | 左の条件以外するとき |
| 動作 | 右折する | 左折する |
| プログラム | DCモーター M1 を停止 DCモーター M2 を正転 | DCモーター M1 を正転 DCモーター M2 を停止 |

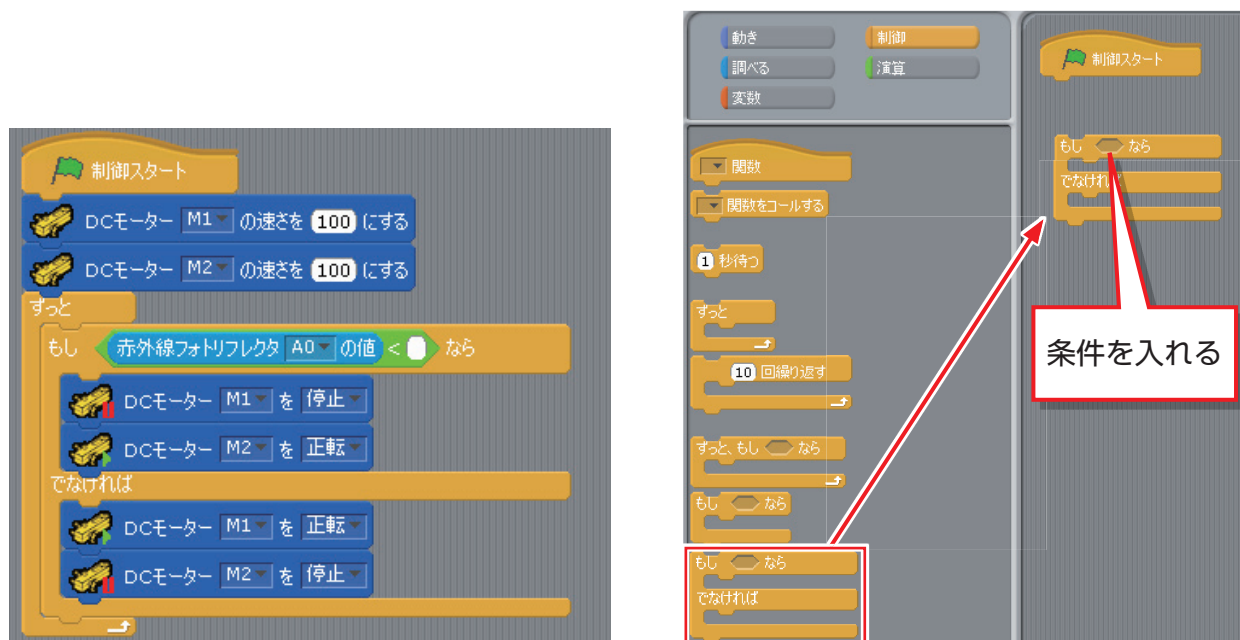
④フローチャート

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ときと越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。



ライトレースカーの別解

以下のプログラムでもライ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。



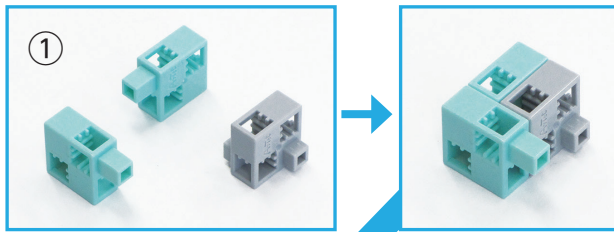
組み立て 説明

<関連する章>

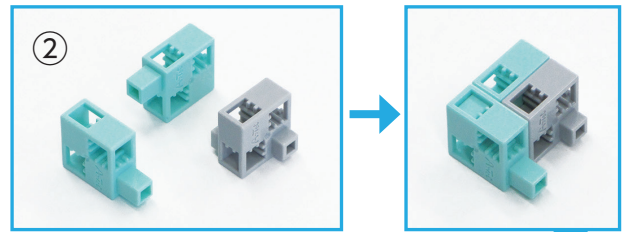
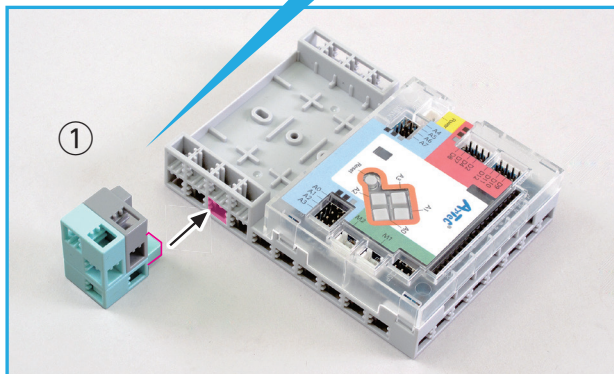
- 第2章～第5章
- 演習

第 2 章 車の組み立て

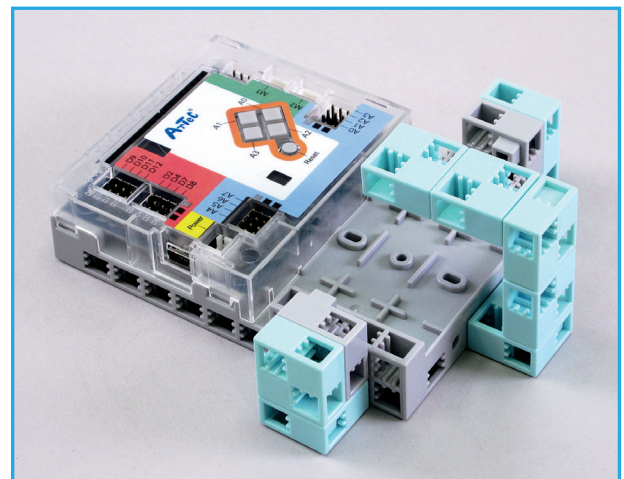
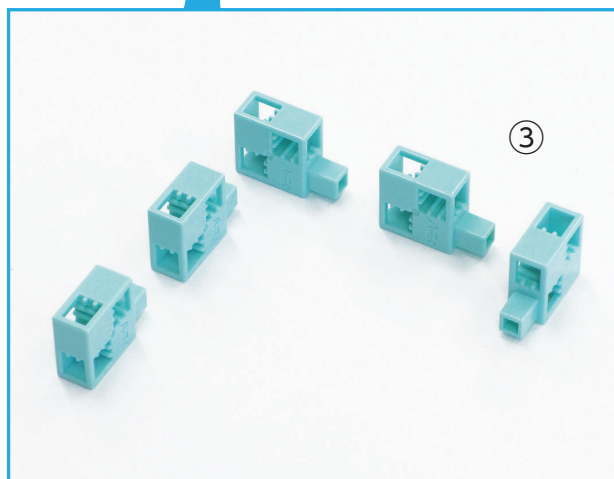
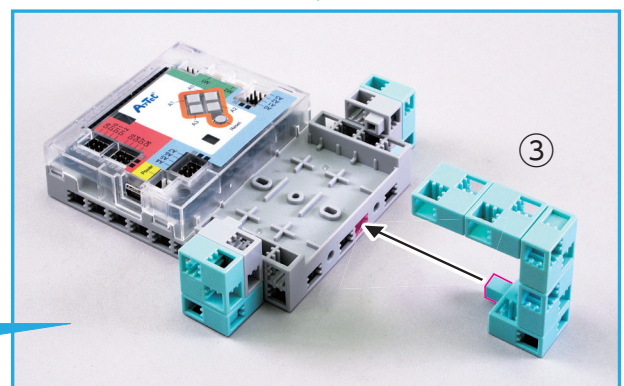
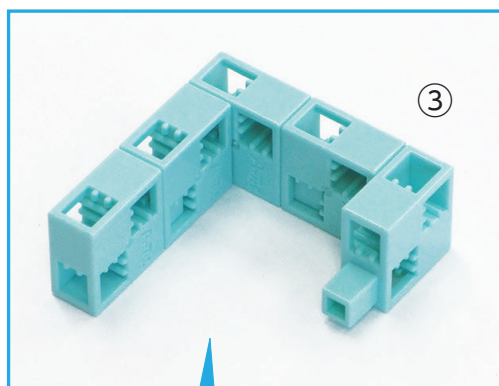
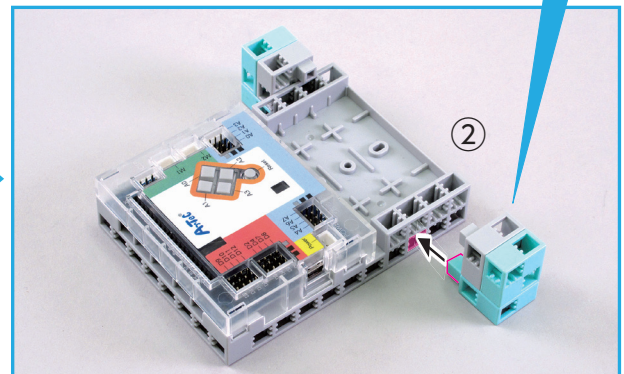
1 ブロックを図のように組み立てます。



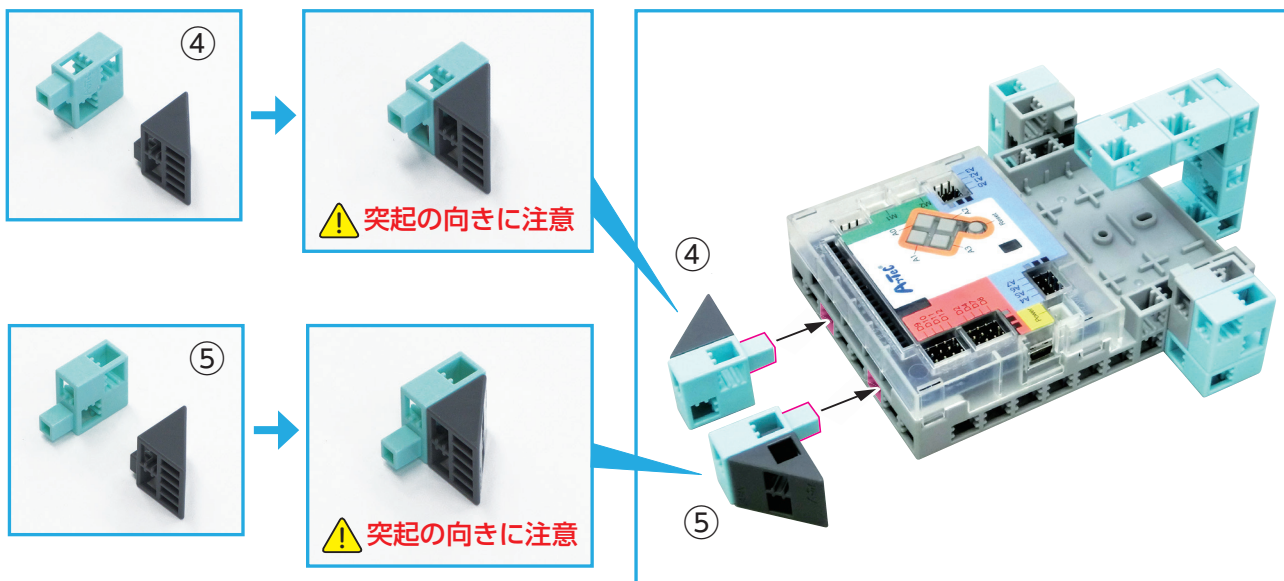
⚠ 突起の向きに注意



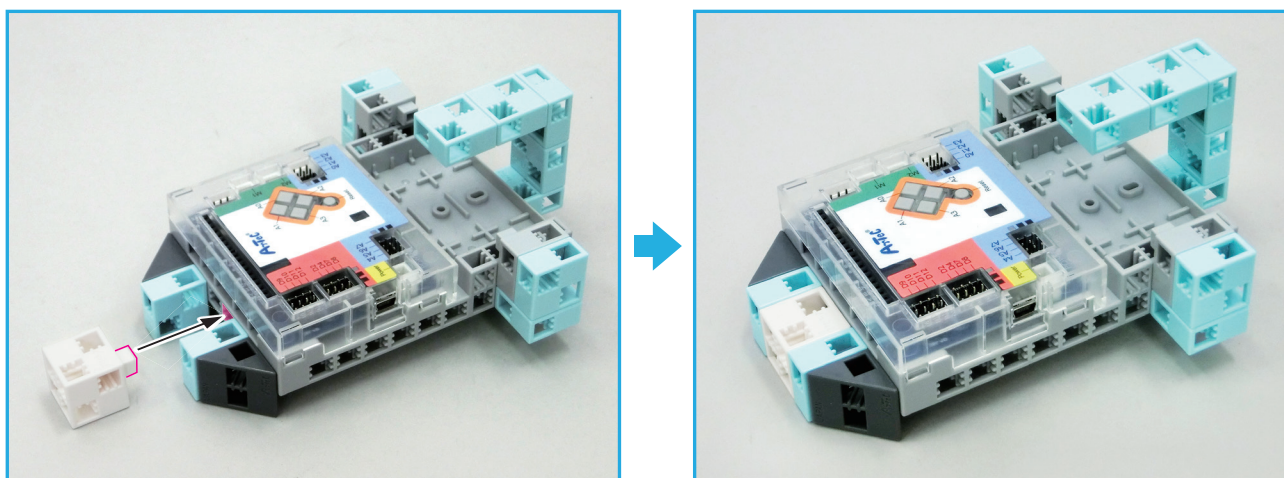
⚠ 突起の向きに注意



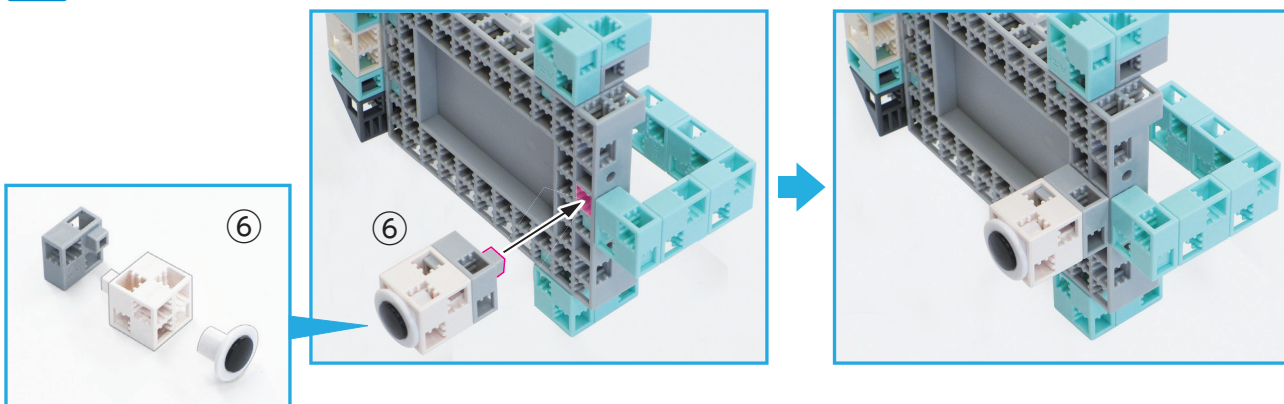
2 ブロックを図のように組み立てます。



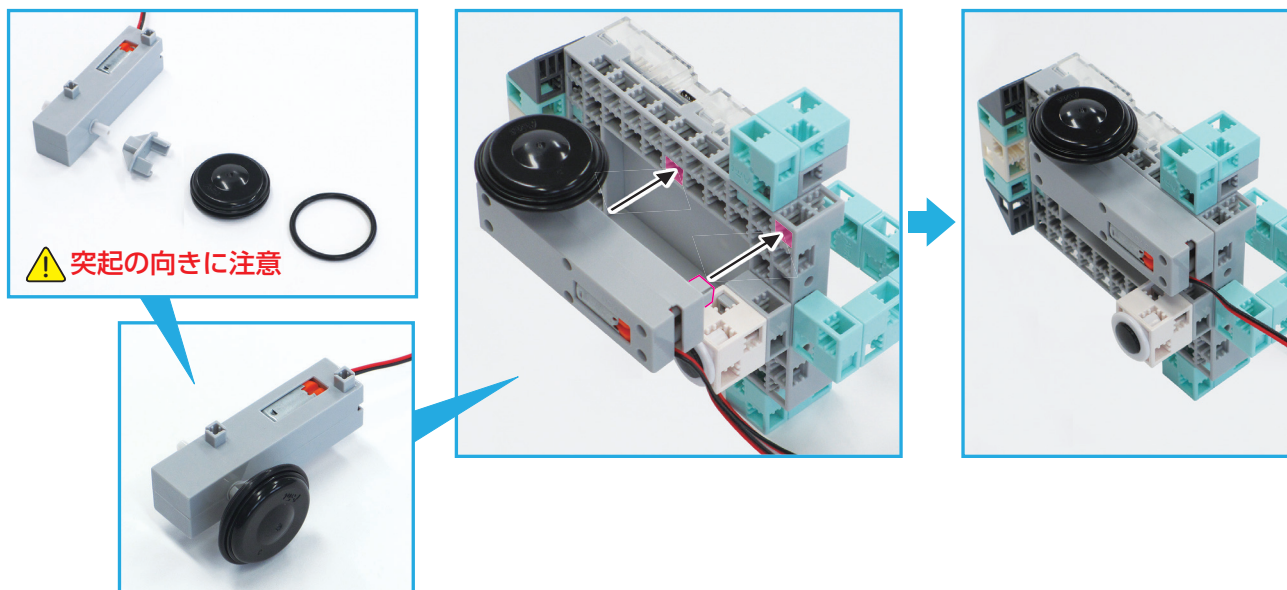
3 ブロックを図のように取り付けます。



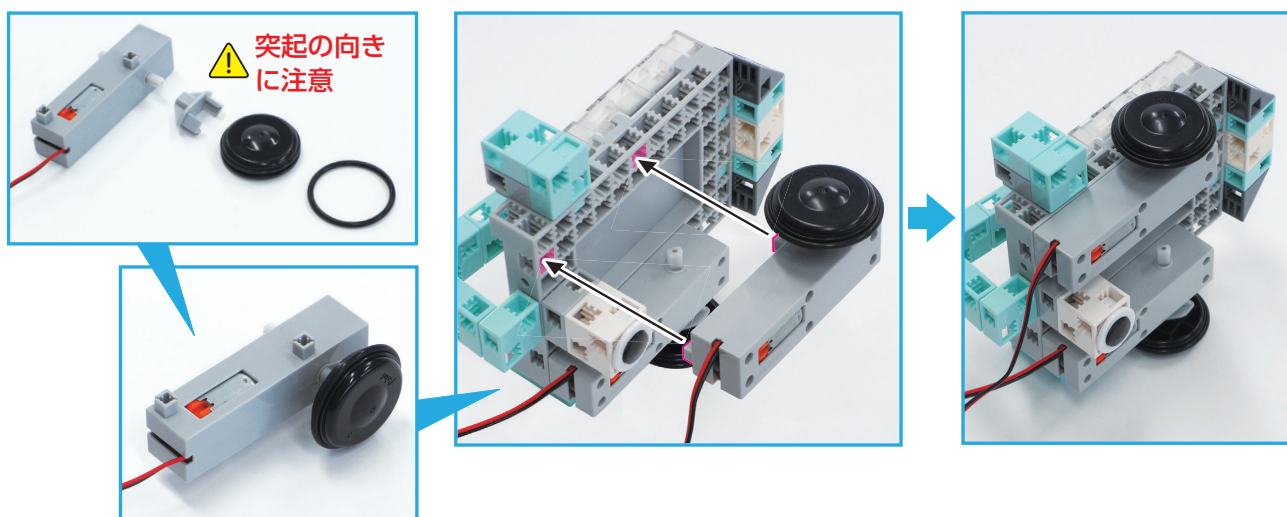
4 ブロックを図のように組み立てます。



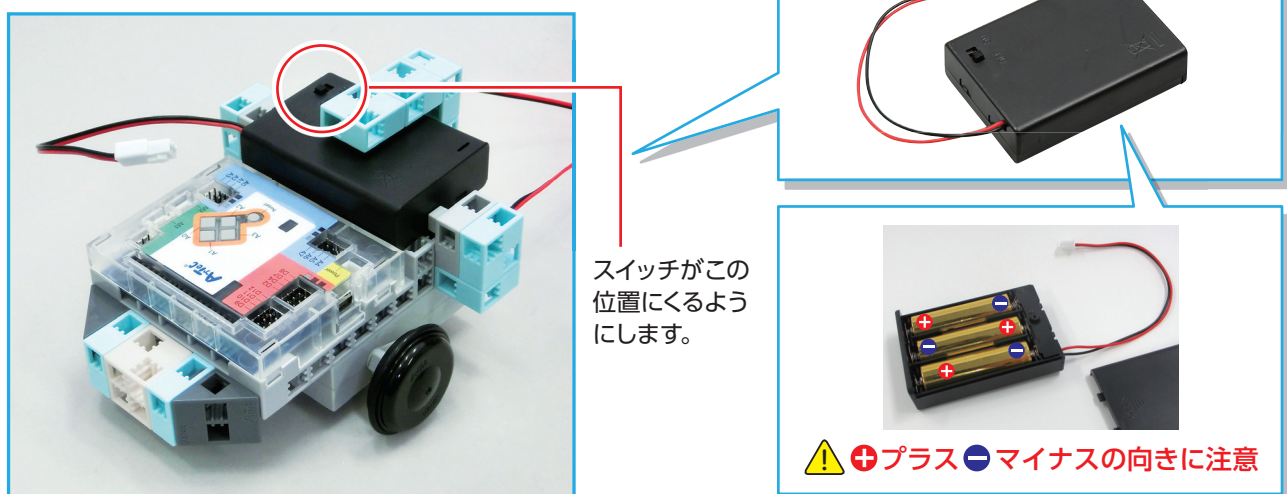
5 DC モーターを図のように組み立て、取り付けます。



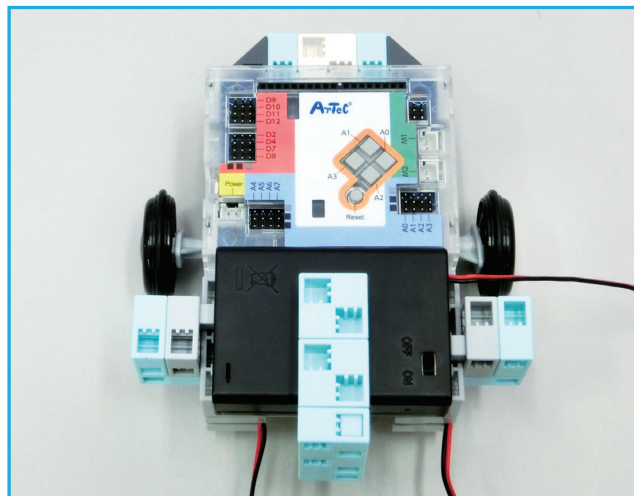
6 反対側も同様にします。



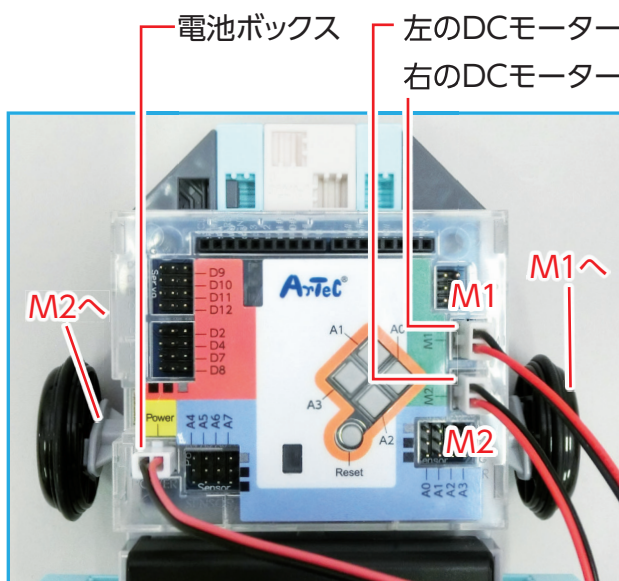
7 電池ボックスを図のようにのせます。



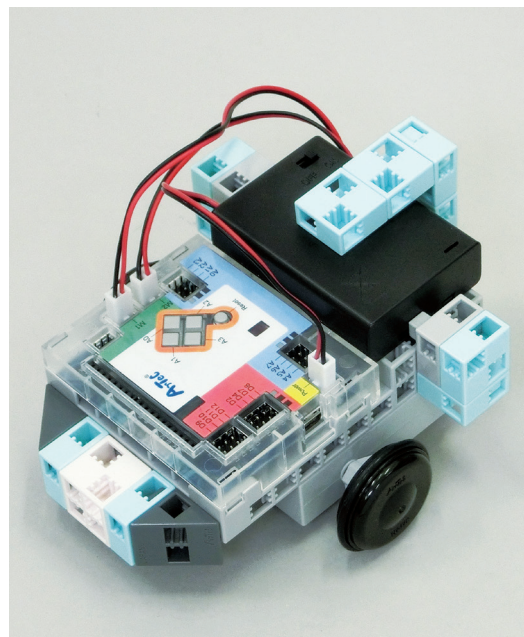
8 コードがモーターの軸に絡まっていないか確認してください。



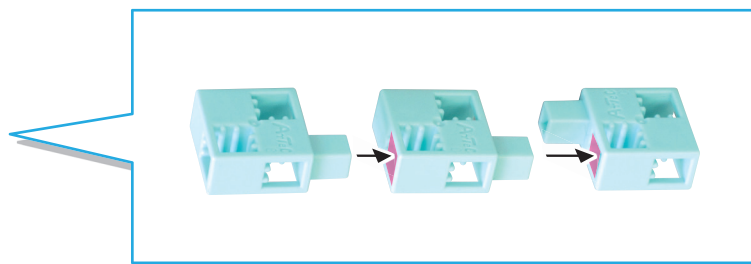
9 DC モーターと電池ボックスのコネクタを
スタディードノに取り付けます。



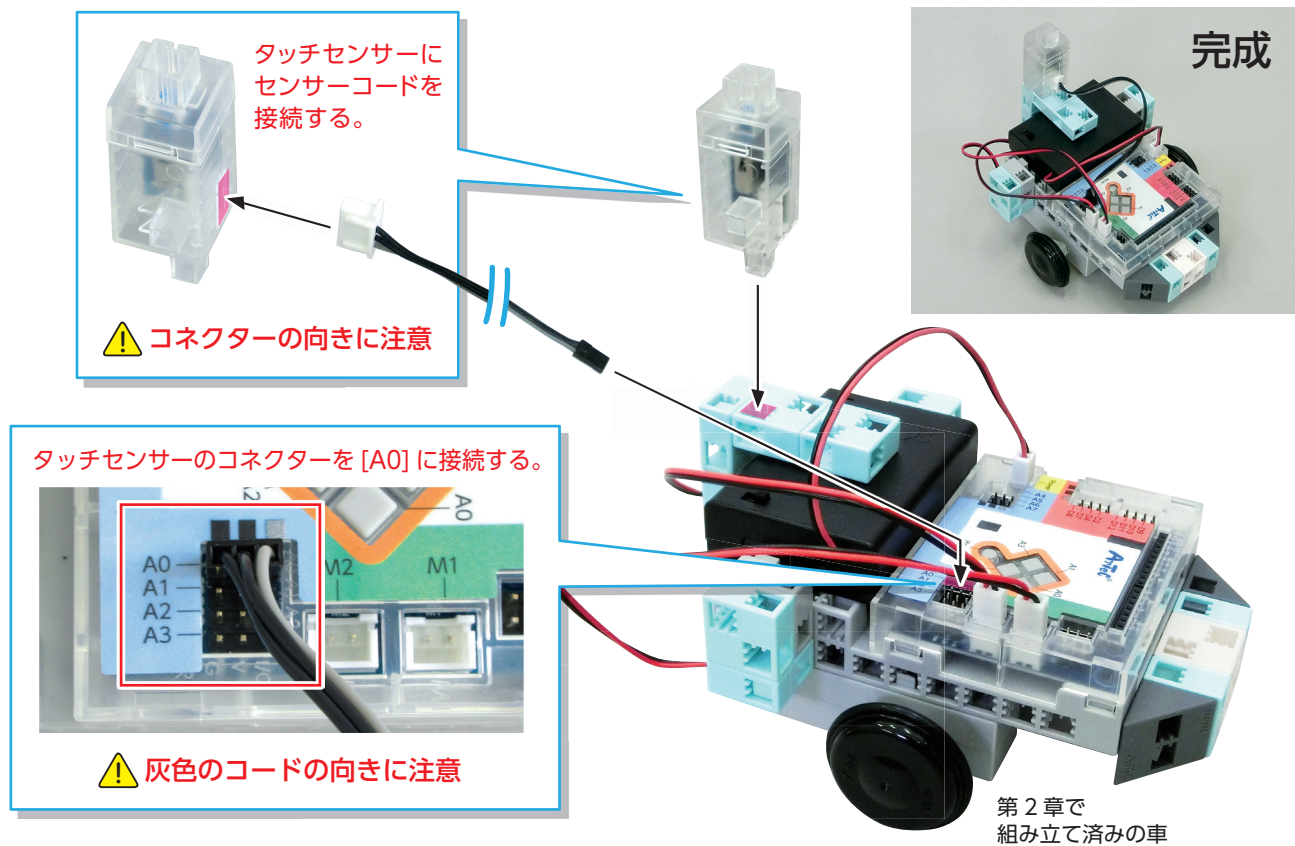
10 完成



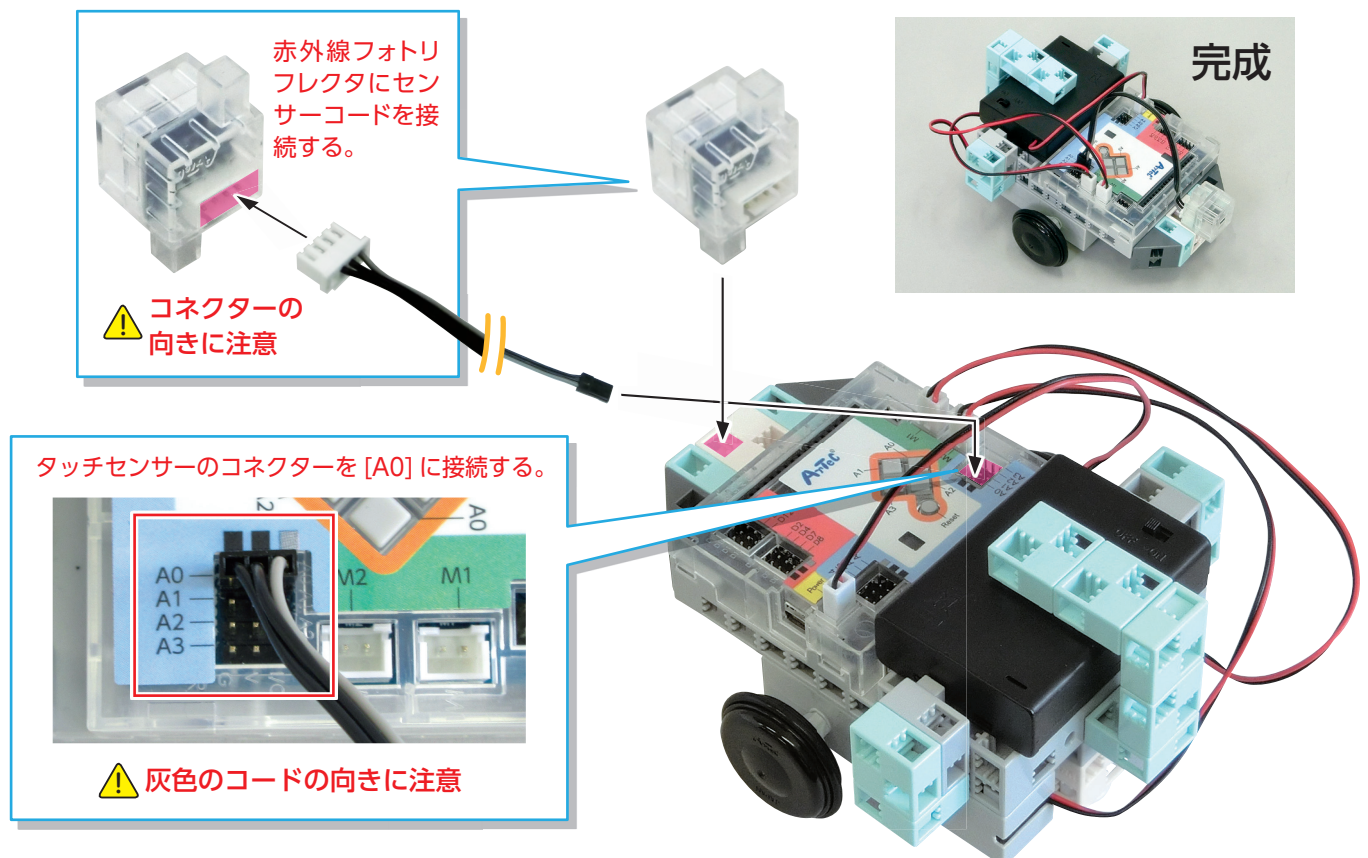
障害物の組み立て



第4章 タッチセンサーの取り付け



第5章 赤外線フォトリフレクタの取り付け



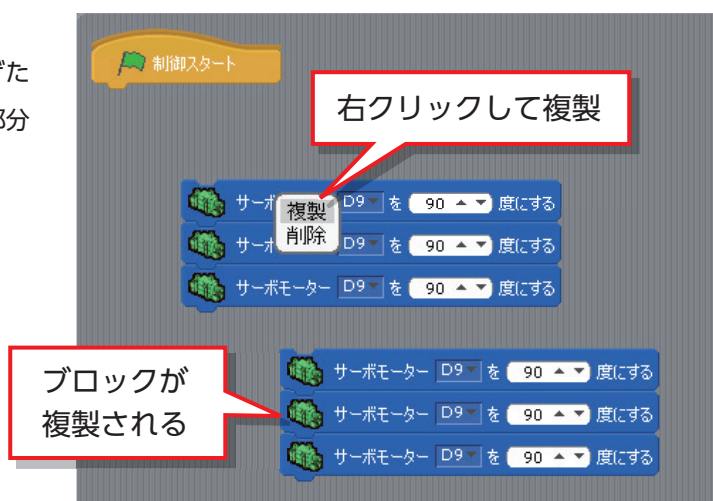
付録

- その他の操作方法
- ブロック一覧表
- 対比表

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。つけ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。



◆ ブロックの削除の取り消し

削除したブロックを戻したい場合は、「編集」から「削除の取り消し」を選択します。

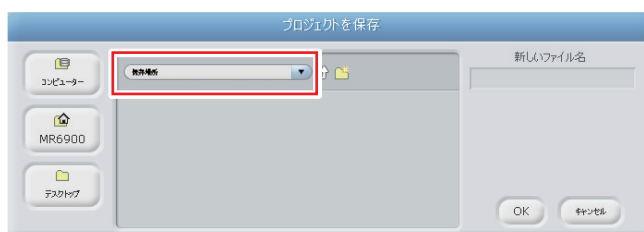


◆ プログラムの保存方法

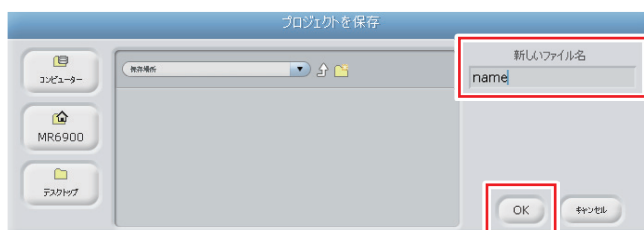
①「ファイル」から「名前を付けて保存」をクリックします。



②保存する場所を選択します。



③ファイルに名前を付けて「OK」をクリックします。




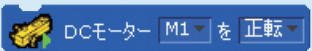
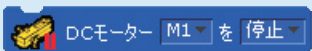








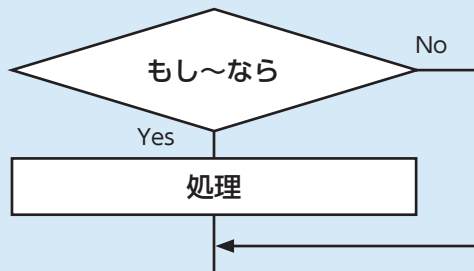


付録 2 ブロック一覧表

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 モーターの速さを指定する |
| |  | DC モーターを正転・逆転させる |
| |  | DC モーターを停止させる |
| 制御 |  | 指定した秒数待つ |
| |  | 囲われたプログラムをずっと実行する |
| |  | 囲われたプログラム指定回数実行する |
| |  | 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 |  | 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 |
| 演算 |  | 「小なり」で値を比較する |
| |  | 「等しい」で値を比較する |
| |  | 「大なり」で値を比較する |

付録3 ブロックとフローチャートの対比表

テキストで使うブロックとフローチャートの対比表です。

| 処理 | ブロック | フローチャート |
|--------------|---|--|
| 動作の処理 |    |  |
| 待つ |  |  |
| ずっと 繰り返す |  |  |
| 指定回数 繰り返す |  |  |
| もし～なら |  |  |
| ～まで待つ |  |  |

確認問題集

- ① コンピューターの動作手順を記述したものを何というでしょうか。
- ② プログラムを作成するときに使う特別な言葉を何というでしょうか。
- ③ 光、音、温度などの周囲の情報を計測する装置を何というでしょうか。
- ④ モーターなどのコンピューターからの命令に従ってエネルギーを動作に変えて仕事をするものを何というでしょうか。
- ⑤ 社会の中で計測・制御の仕組みが使われている製品を1つ挙げ、その中で使われているセンサーとその役割について説明しなさい。

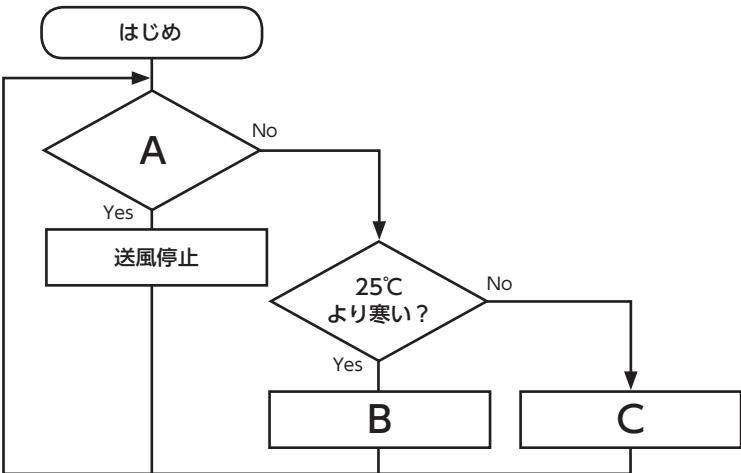
| | |
|---|------|
| ① | ② |
| ③ | ④ |
| ⑤ | 製品 |
| | センサー |
| | 役割 |

- ① 命令を順番通りに行う処理のことを何というでしょうか。
- ② 矢印や記号を用いて処理の手順を整理した図のことを何というでしょうか。
- ③ 次のA～Eの処理を表すフローチャートの記号の形をそれぞれ描きなさい。

A：処理の開始と終了 B：一般的な処理 C：繰り返しの開始

D：繰り返しの終了 E：条件分岐

- ④ 次のA～Cの空欄を埋めて室温を25℃に保つエアコンの動作を表すフローチャートを完成させなさい。



- ⑤ 車を前に3秒動かしたあと後ろに3秒動かすときのフローチャートを、以下の記号に「右・左」もしくは数字を入れて表しなさい。

はじめ

おわり

秒待つ

の DC モーターを正転

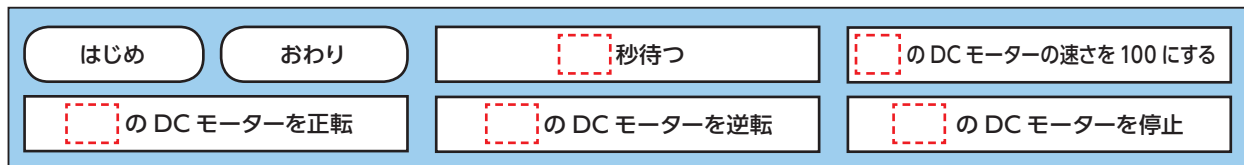
の DC モーターを逆転

の DC モーターの速さを 100 にする

の DC モーターを停止

| | | | |
|---|---|---|---|
| ① | | | ⑤ |
| ② | | | |
| ③ | A | B | |
| | C | D | |
| | E | | |
| ④ | A | | |
| | B | | |
| | C | | |

- ① 車を1秒間左折させるときと1秒間右回転させるときのフローチャートを、以下の記号に「右・左」もしくは数字を入れて表しなさい。



- ② 正方形を描くように車を動かそうと思い、プログラムをつくりました。無事、その図形のように車は動きましたが、停止のプログラムが繰り返しの中に入っているにも関わらず、途中で止まらなかったことが気になりました。その理由を正しく説明したものを選びなさい。

- A: 「速さ」ブロックが繰り返しの中に入っていないから
 B: 「正転」と「逆転」が逆になっているから
 C: 「停止」ブロックの後に「待つ」ブロックがないから



①

(1 秒間左折)

(1 秒間右回転)

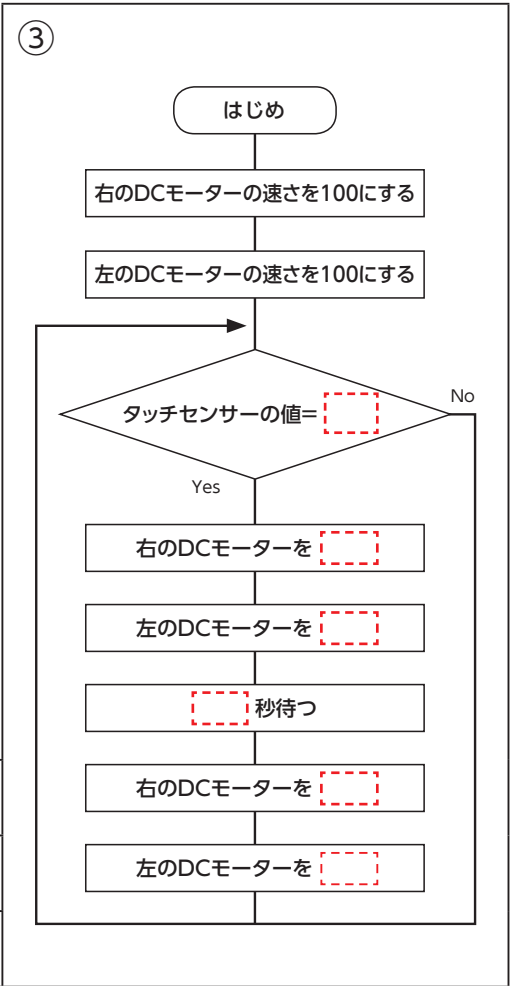
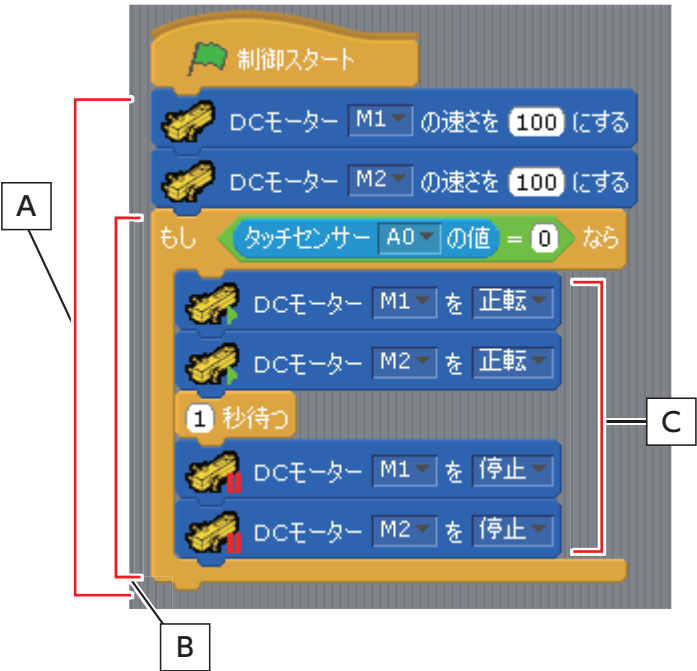
②

- ① 条件によって動作を選択する処理のことを何というでしょうか。
- ② プログラムの誤りを発見し、正しく動かすため修正することを何というでしょうか。
- ③ ボタンが押されたときのみ1秒間前進する車の動作の手順をフローチャートにまとめます。以下の情報を参考に、フローチャートに適当な値や文字を書き込みなさい。

| | タッチセンサーが押されていないとき | タッチセンサーが押されているとき |
|-----------|-------------------|------------------|
| タッチセンサーの値 | 1 | 0 |

- ④ ③のフローチャートを見てプログラムを作りましたが、プログラムを実行すると想定通りに動きません。フローチャート通りの動作をさせるようにプログラムを修正する場合の適切な修正方法を選びなさい。

- A:プログラム全体を「ずっと」ブロックで囲む
- B:「もし～なら」ブロック内のプログラムを「ずっと」ブロックで囲む
- C:「もし～なら」ブロックを「ずっと」ブロックで囲む



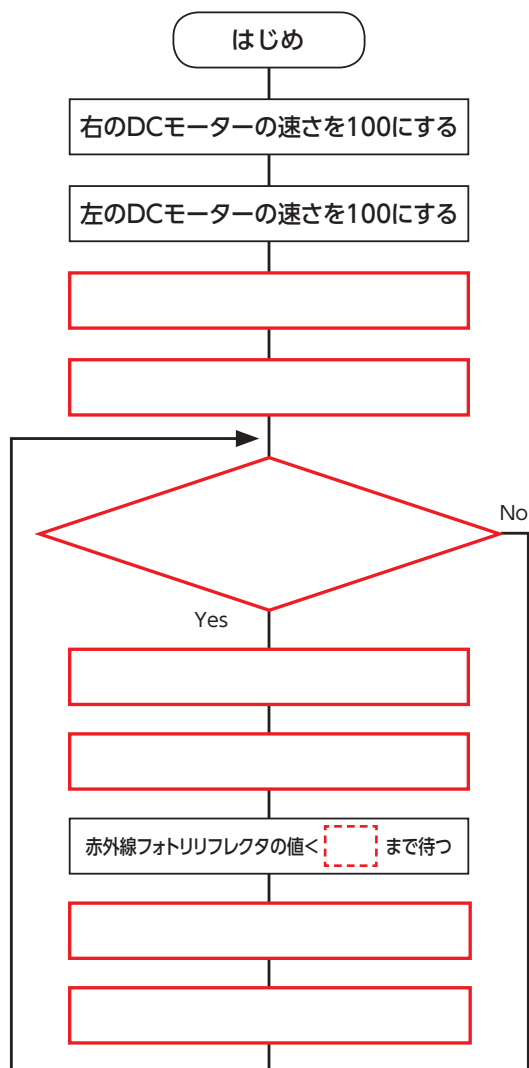
| |
|---|
| ① |
| ② |
| ④ |

- ① 窓際の席で赤外線フォトリフレクタを使うことになりました。このとき、注意しなければならないことを説明しなさい。
- ② 前進中に障害物を5cm前で感知したときに障害物が無くなるまで停止し続ける車の動作の手順をフローチャートにまとめます。以下の情報を参考にして、フローチャートを完成させなさい。

| | 障害物が 5cm 前にあるとき | 障害物がないとき |
|---------------|-----------------|----------|
| 赤外線フォトリフレクタの値 | 15 | 3 |

①

②



| | | | |
|----------|-------|---------------------|-----------|
| ① | プログラム | ② | プログラミング言語 |
| ③ | センサー | ④ | アクチュエータ |
| ⑤ (例) | 製品 | 自動ドア | |
| | センサー | 赤外線フォトリフレクタ（人感センサー） | |
| | 役割 | 人がいるかどうかを判断する | |






解説

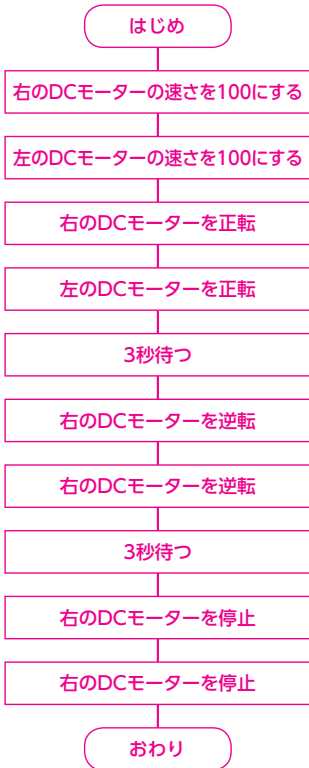
① プログラム・フローチャート・アルゴリズムの違いは以下のように説明できます。

| 手順を記述したもの | 手順を図で表したもの | 手順の考え方 |
|-----------|------------|--------|
| プログラム | フローチャート | アルゴリズム |

② 「プログラム言語」でも正解です。

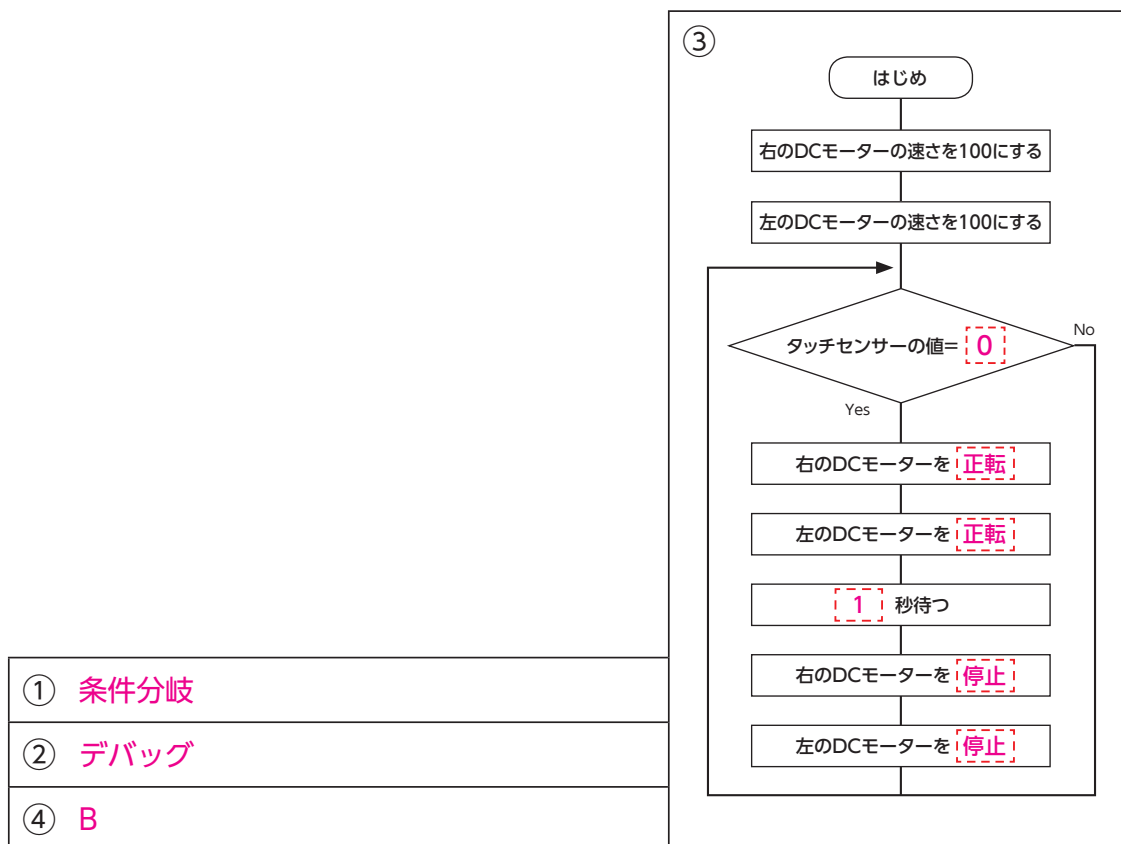
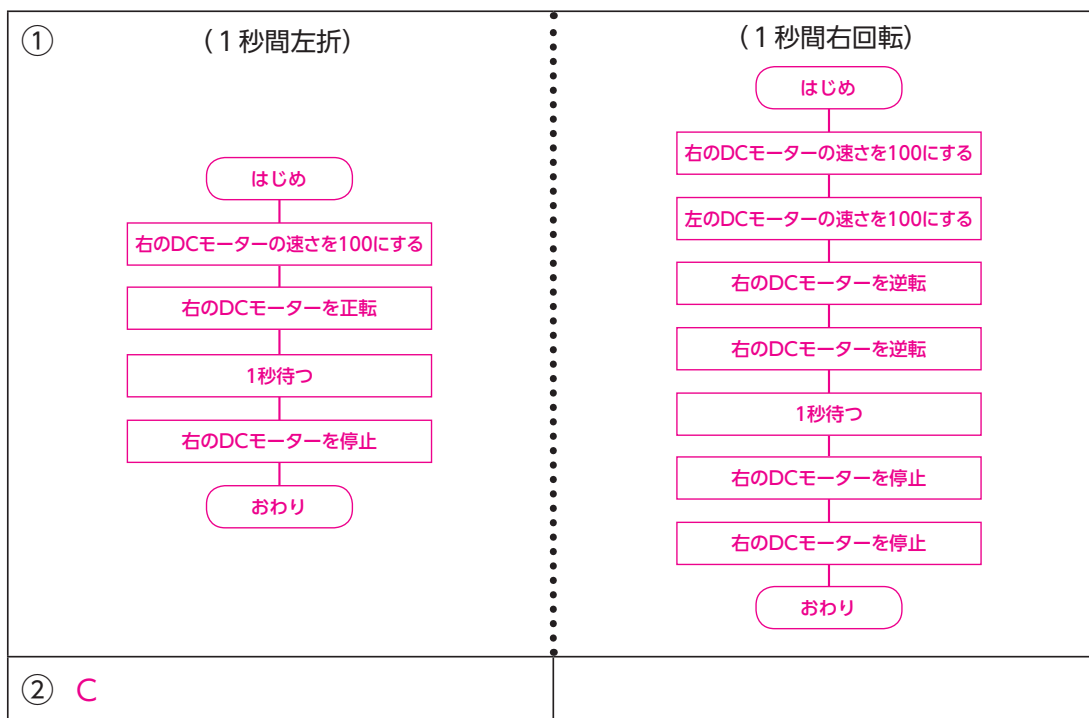
⑤ その他別解が多数あります。

| | | |
|---|---------------|---|
| ① | 順次処理 | |
| ② | フローチャート | |
| ③ | A |  |
| | B |  |
| | C |  |
| ④ | D |  |
| | E |  |
| | | |
| ④ | A 25℃になっているか？ | |
| | B 温風を送る | |
| | C 冷風を送る | |

| | |
|---|---|
| ⑤ |  |
|---|---|

解説

③ Cで「繰り返し〇回」、Dで「繰り返し戻る」と書いてあっても正解です。

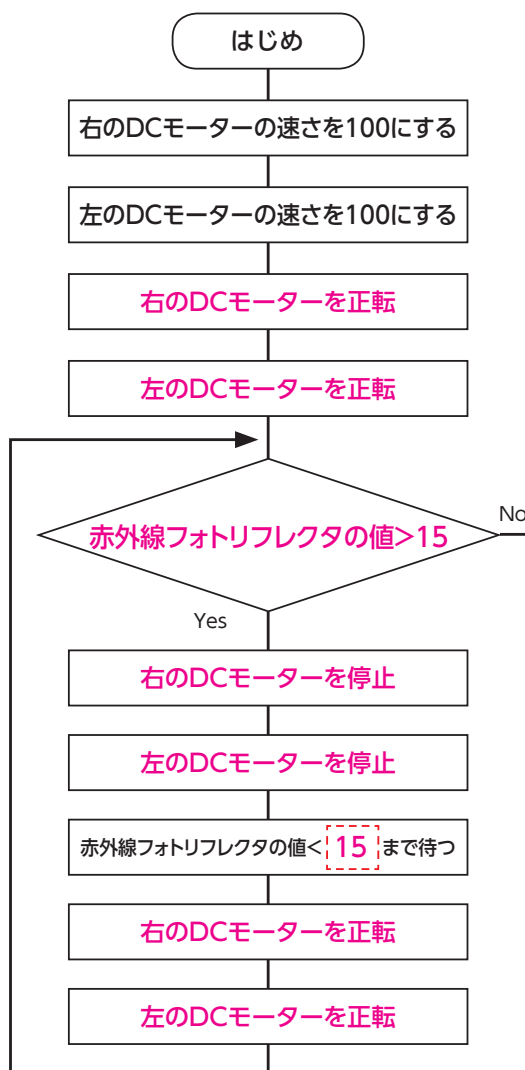


解説

- ④ この問題では、③のフローチャート通りのプログラムを作るための修正を行う必要があるため、Aの修正は不適切になります。

① 赤外線フォトリフレクタを太陽光に向かって配置しないようにすること。

②



解説

- ① 赤外線フォトリフレクタの検出部を太陽光に向けない旨が書かれていれば正解です。
- ② 障害物が5cm以内にいるとき停止するので、しきい値は15にする必要があります。障害物がない場合との中間値を取った場合、「5cm以内で停止」という目的に合わないため（5cmより遠くても停止してしまう）、正解になりません。



計測と制御キットD

プログラムによるセンサーカーの制御
教員用

テキストに関するお問い合わせ

TEL : 072-990-5514

E-mail : support@artec-kk.co.jp