IoTプログラミング教材シリーズ

リモコン サポカー EV-100
ガイドブック

初級・中級者向け

カーネルキャリアスクール

2019年10月

Kernel Concept, Inc.

はじめに（必ずお読みください）

Kernel Concept, Inc.

本教材はIoTプログラミング初級・中級レベルの方を想定した、プログラミング的思考トレーニングのための教材です。自動車の組立、配線を行うことでハードウェアの基本を理解します。さらにサンプルプログラムを使って、実際の動作を見ながらプログラミングを体験できます。

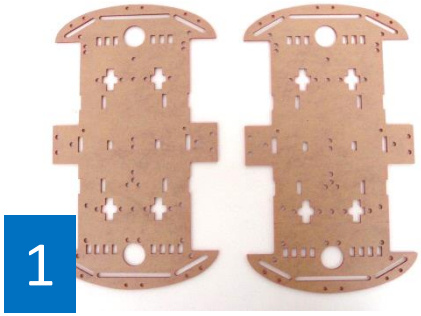
本教材のサンプルプログラムを体験した後は、オリジナルなプログラムの作成にチャレンジしましょう。はじめは、プログラムを改造するところから始めます。一箇所ずつの改造からはじめてみましょう。エラーが出たり、うまくいかない時は、直前の状態に戻して考察しましょう。
プログラミング言語（C言語）の様々なコマンドや制御については、関連参考図書やインターネットの解説情報で自習することも大切です。自分で調べる、考える、失敗することで、スキルをアップができます。うまくいかない時、わからない時が、チャンスです。
グローバルなテーマですから関連情報が英語で提供されている場合があります。英語も併せて学びましょう。

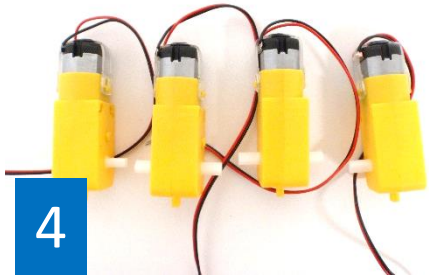
単独自習で本教材に要する時間は、組立、配線に3時間程度、プログラミングの基本的な内容について10時間程度を想定しています。

なお、本教材を利用するには、コンピュータ操作等に関して以下のスキルがあらかじめ必要となります。
Windows PCなどのコンピューターで以下の操作ができる
1）ワード、エクセルなどでの文書、表作成、ファイル操作など
2）アプリケーションソフトのインストールや設定
3）USBデバイスなど接続、設定
4）インターネットへのアクセス
5)簡単な工作や電気配線の経験または理解

内容物（車体部品）

Kernel Concept, Inc.





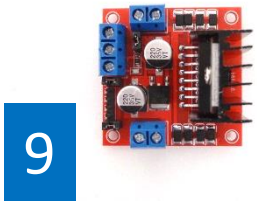
1. 車台
2. タイヤ
3. 固定用部品
4. モーター
5. 車軸用スリーブ

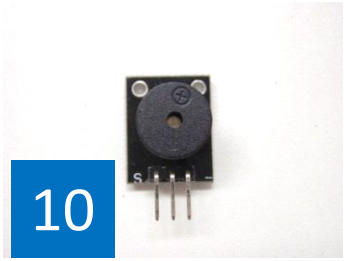
＊3.固定用部品の内、
黒色円形状部品（エンコーダー部品）、
一部のネジは教材中では使用しません。

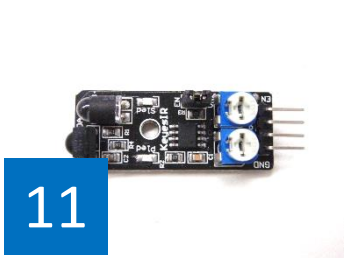
＊製造時期等によって、各部品の色、形状数量等が若干異なる場合がございます。ご了承ください。

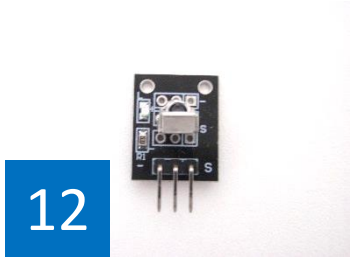
内容物（電気部品）

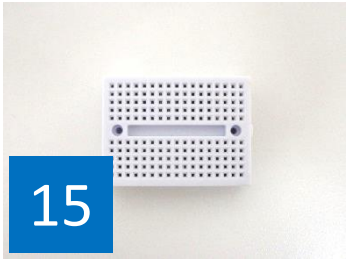
Kernel Concept, Inc.







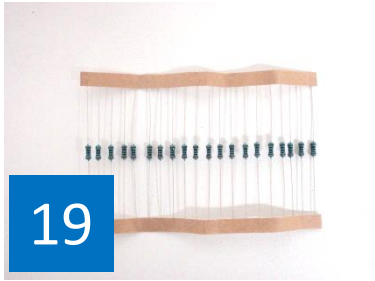


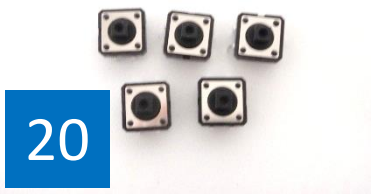


6. Arduinoマイコンボード
7. USBケーブル
8. USBメモリー
9. モータードライバー
10. ブザー
11. 赤外線近接センサー
12. リモコンレシーバー
13. リモコン
14. 電池ケース
15. ブレッドボード
16. 配線用リード線3種
17. 両面テープ

内容物（その他オプション）

Kernel Concept, Inc.





18. LED（4色）
19. 抵抗（200Ω）
20. ボタンスイッチ

＊これらは、皆さんが応用で自由に利用する部品です。

自分で用意するもの

1. Windows PC
2. インターネット接続環境
3. ハサミ
4. ドライバー（プラス、マイナス）

ご使用上の注意

ボード裏面には電気配線があります。ショートしないよう、木や紙の上に置いてご利用ください。通電中にボードを金属など電気を通す物の上には絶対に置かないでください。また、金属などでボードに触れないでください。電気回路のショートは故障の原因です。

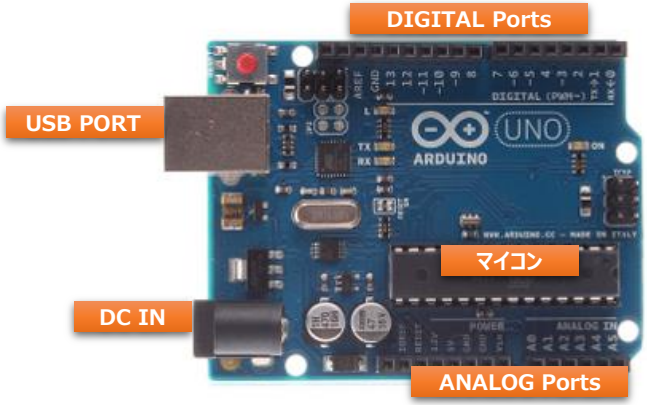
付属サンプルプログラム

Kernel Concept, Inc.

注意！
付属USBの内容を、誤って消したり、書き換えてしまったりしないように、
PCのデスクトップなどにコピーして使おう。

- Blink_LED　　１つのLEDが点滅
- Drive　　車輪を色々回転
- Drive_IRremote_Rev1　　自動車をリモコンで制御
- IR_remote　　リモコンからのコード確認

Lesson1 マイクロコンピューター



- マイコン（マイクロコンピューター）

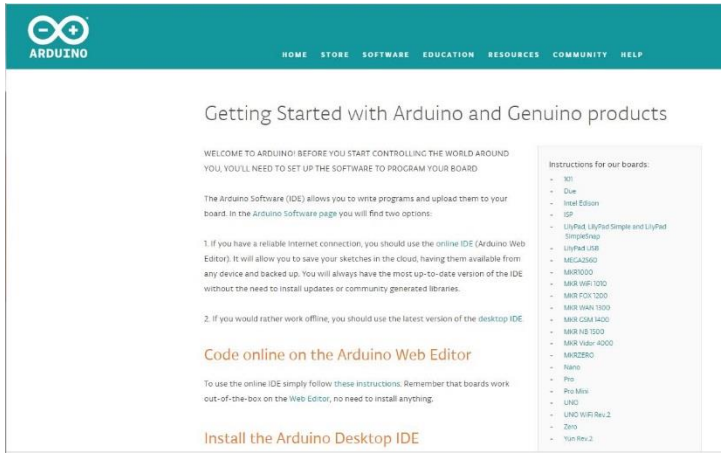
マイコンは、電気機器を制御するための電子部品、最近のほとんどの電気機器に組み込まれている。
本テキストは、Ardunio（アルディーノ）またはその互換ボードというマイコンボードを使用。
マイコンボードは、マイコンチップを扱い易くするために最低限必要な整備を併せて基盤に載せたもの。

- Ardunio（アルディーノ）または互換機

- Arduinoは、本体の回路や開発環境(IDE)が無料公開されており、情報が豊富で取扱いが容易。
- マイコンを使いやすくまとめたArduinoまたはその互換機は、プログラミング教育に向いている。

Lesson1　ソフトウェアのインストール

Kernel Concept, Inc.



- 開発環境ソフトのPCへのインストール

https://www.arduino.cc/en/Guide/Windowsから
Ardunioの開発環境（IDE）をPCにダウンロードし、インストール（最新バージョン）

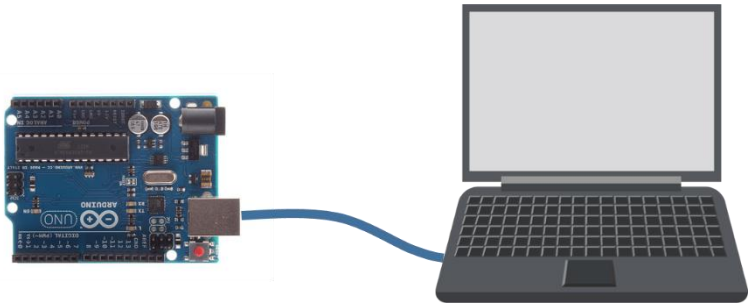
＊または、付属USBメモリのIDE
(Arduino-1.8.8-windows)をインストール

インストール方法の詳細は
https://www.arduino.cc/en/Guide/Windows
にしたがって進めましょう

Windows OSの自動更新時などはソフトのインストールやソフトの動作が停止状態になることがあります。
PCのディスクへのアクセスが無くなってから試みてください。
また、うまく動作しないときはPCを再起動してください。

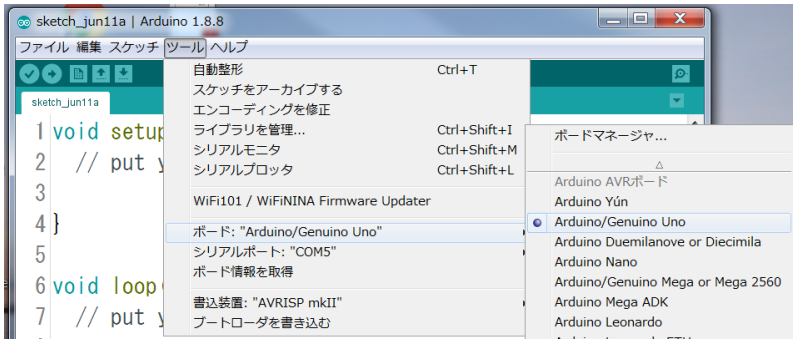
Lesson1　ソフトのインストール

Kernel Concept, Inc.

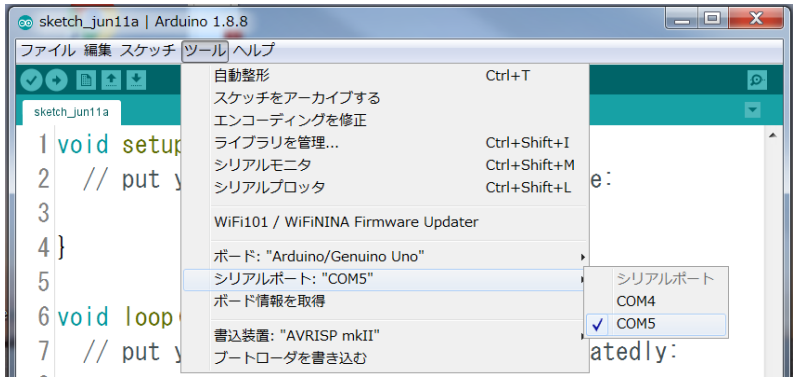


- Arduino ボードに USB ケーブルでつなぐ

PCとArdunioボード（学習ボードの下側）にUSBケーブルを差し込む



- ボード種類の設定
  - メニュー＞ツール＞ボード:”Ardunio/Genuino Uno”　でボード種類を選択

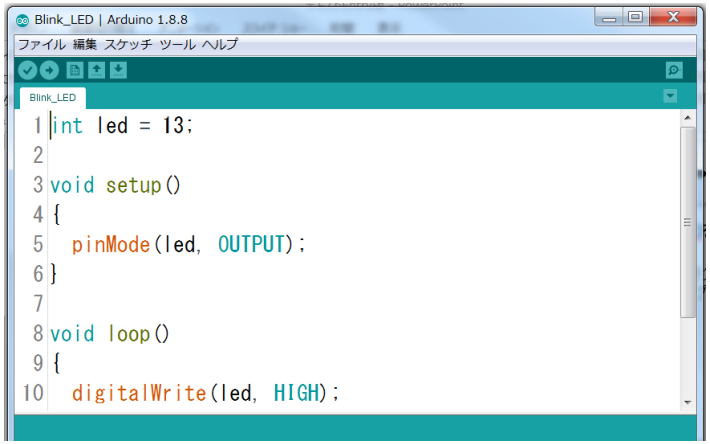


- シリアルポート番号の設定

PCとボードで通信できるようにするシリアルポート番号の設定（COM6など）
＊数字番号は使用するPCによって異なることがあります。
メニュー＞ツール＞シリアルポート＞COM6で選択

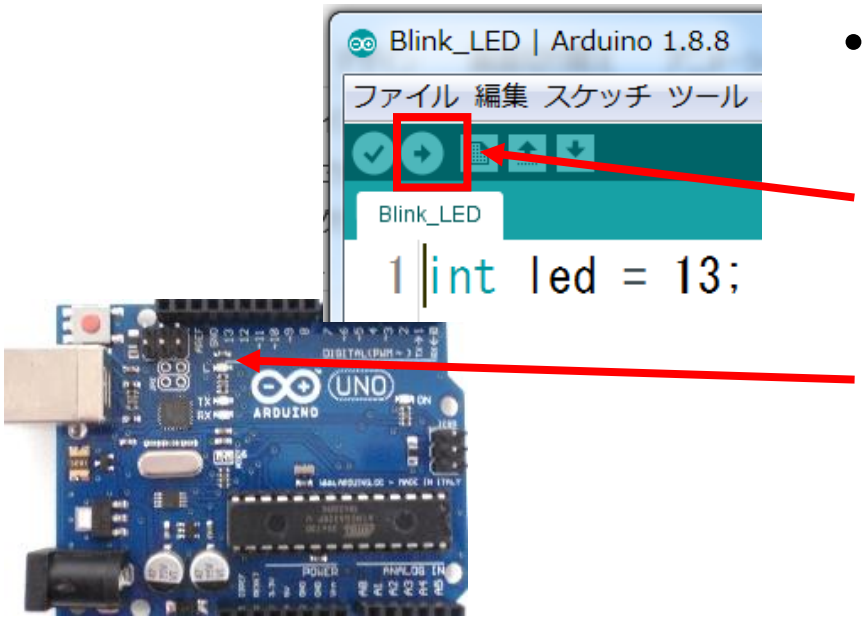
Lesson1　LEDの点滅制御

Kernel Concept, Inc.



- サンプルプログラムBlink_LEDを起動

サンプルフォルダー内のBlink_LEDをダブルクリックすると、IEDアプリが起動しサンプルプログラムが画面に表示されます。



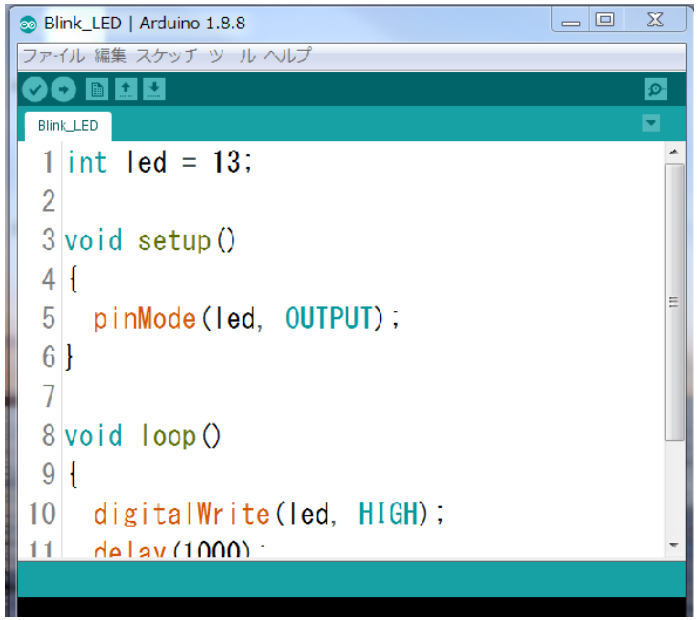
- プログラムをボードへ書き込む

をクリックすると、サンプルプログラムをマイコンボードへ送る（書きこむ）
ボードのLEDが点滅
1秒点灯、1秒消灯の繰り返し

Lesson1　LED点滅制御プログラミング

初級者・中級者

Kernel Concept, Inc.



- サンプルプログラムに手を加えてみよう

変更事項：　0.5秒点灯、0.5秒消灯の繰り返し

プログラムコマンド（命令）の説明

digitalWrite(出力先, 状態)

　　出力先を指定した状態（HIGH/LOW）にする

delay(時間ms)

　　指定しただけ時間待つ

プログラムの変更

```
void loop()
{
  digitalWrite(led, HIGH);
  delay(500);
  digitalWrite(led, LOW);
  delay(500);
}
```

ここの数値を色々変えて実験しよう！

＊変更したプログラムは再度ボードに書き込む必要があります

いろんなコマンドについての詳細は、メニュー＞ヘルプ＞レファレンスでLanguage Referenceを参照しましょう。

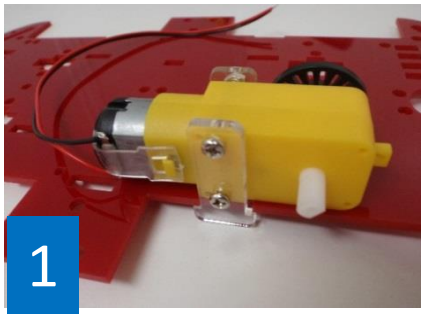
＊プログラムを変更したら、別のファイル名で保存することをお勧めします。
そのまま、ボードへ書き込むと、元のサンプルプログラムファイルが書き換えられてしまいます。

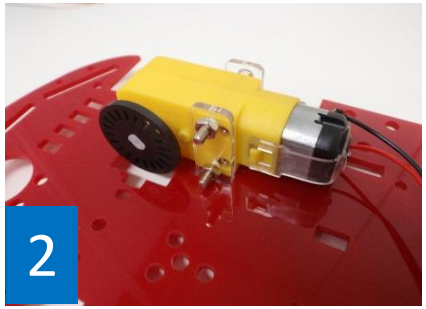
初級者・中級者

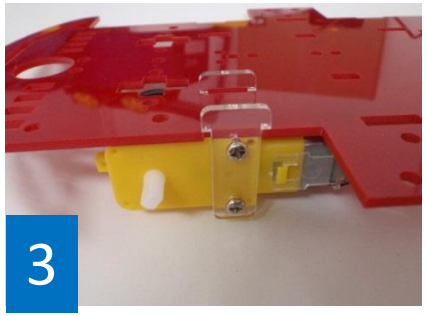
Lesson2 組立（車体）

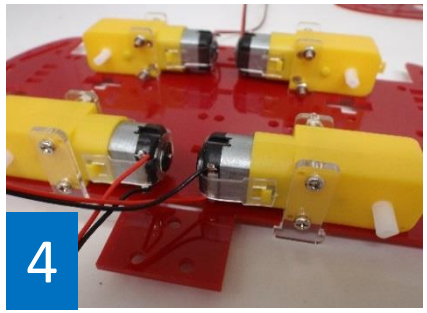
Kernel Concept, Inc.

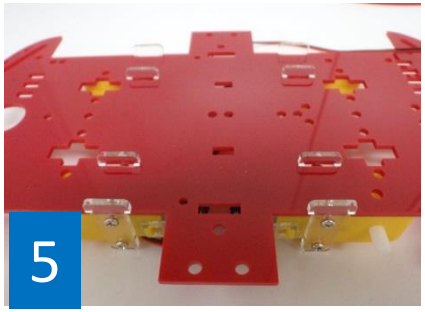
写真を参考に組み立てていこう （紙の保護シートは面倒なら剥がさなくてもOK）

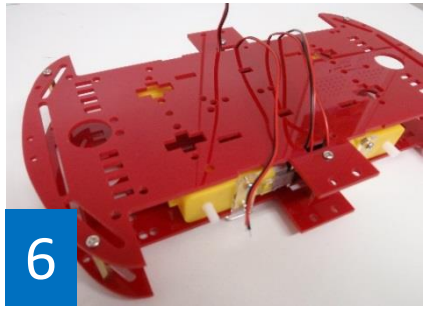


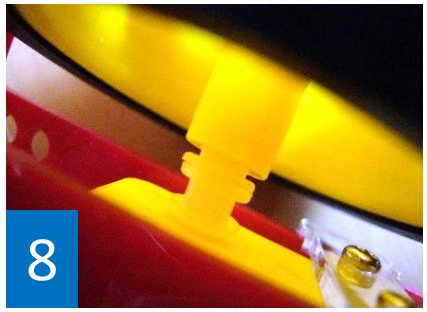


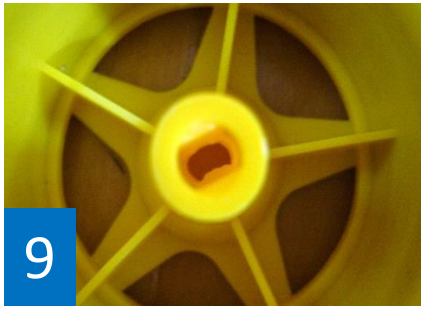






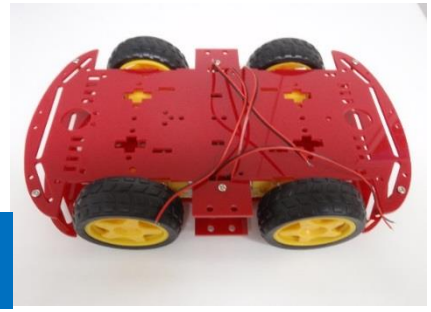






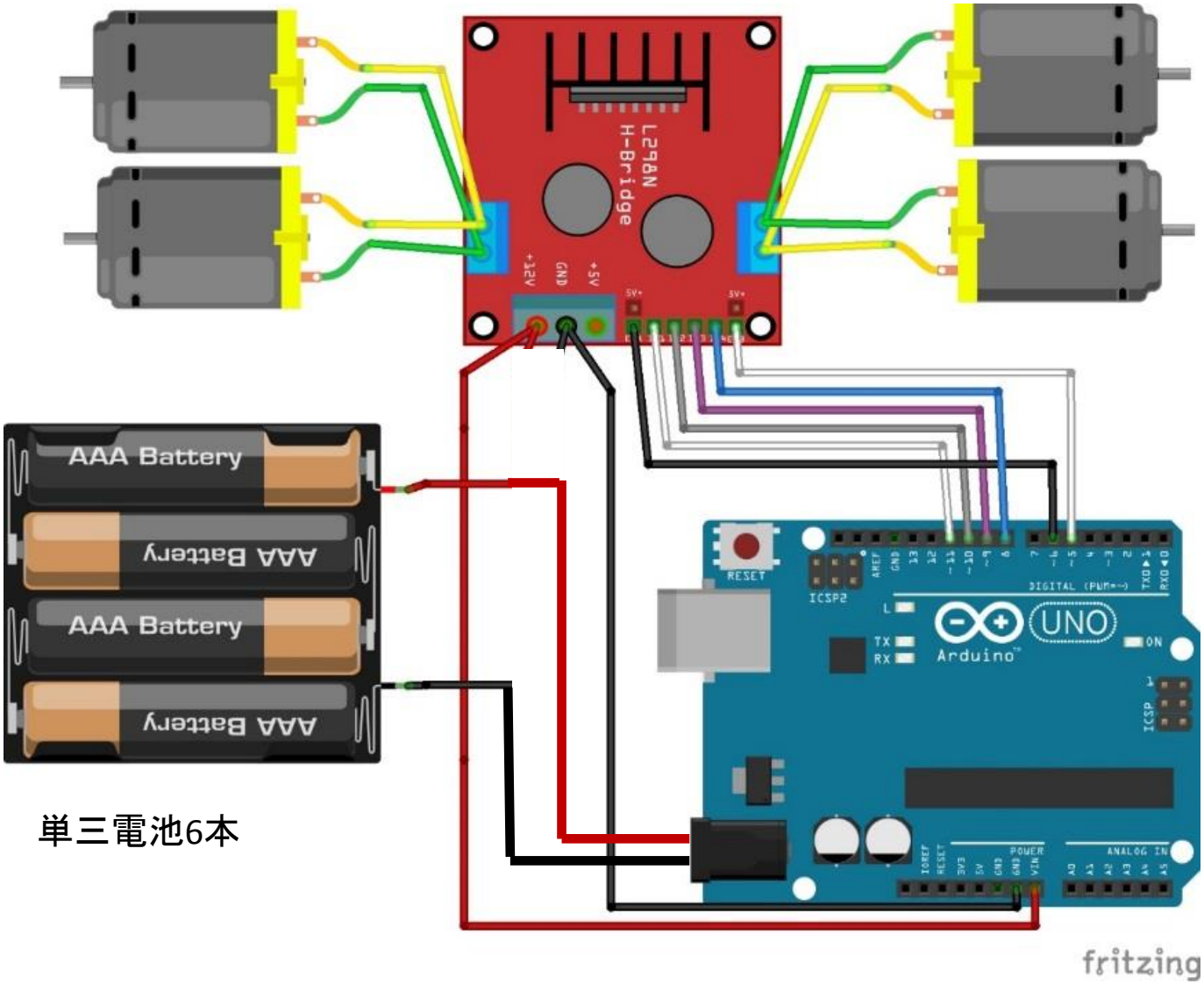
モーター軸とタイヤ穴の
取付け角度に注意

車体完成



Lesson2 組立（電気配線）

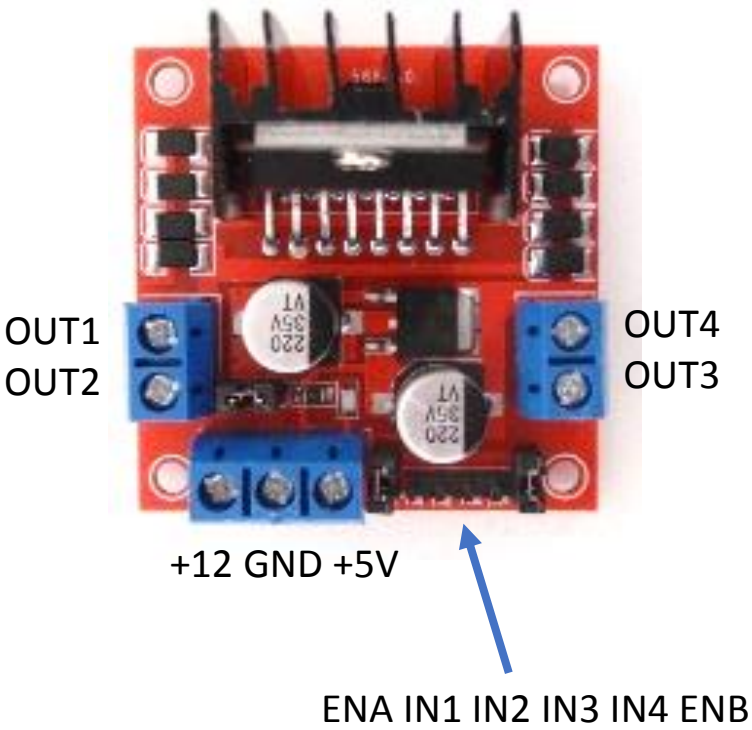
Kernel Concept, Inc.



単三電池6本

配線の色は実際のもの異なる場合があります。

Lesson2 モータードライバー接続先

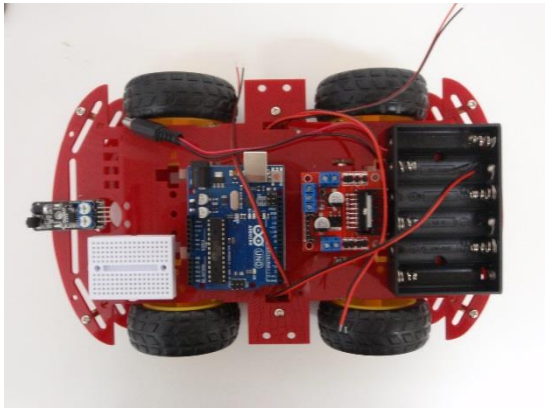


モータードライバー	接続先
OUT1	右側モーター黒線（2本）
OUT2	右側モーター赤線（2本）
OUT3	左側モーター赤線（2本）
OUT4	左側モーター黒線（2本）
+5V	未使用
GND	マイコンGND
+12V	マイコンVin
ENA	マイコン6番
IN1	マイコン11番
IN2	マイコン10番
IN3	マイコン9番
IN4	マイコン8番
ENB	マイコン5番

ENA, ENBのジャンパーソケットは外し、
IN1～4のピンの列のENA, ENBピンを使用

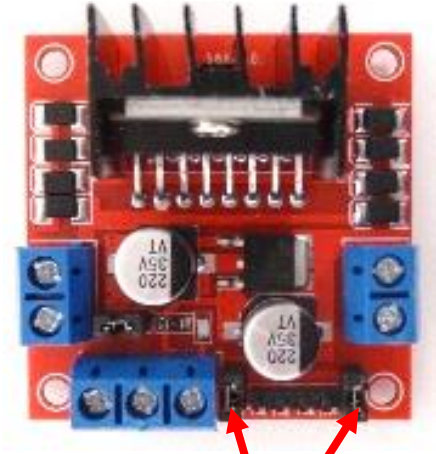
Lesson2 組立（モータードライバー）

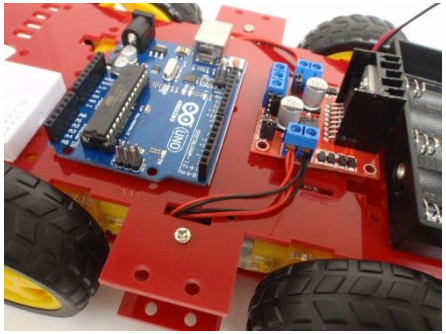
Kernel Concept, Inc.

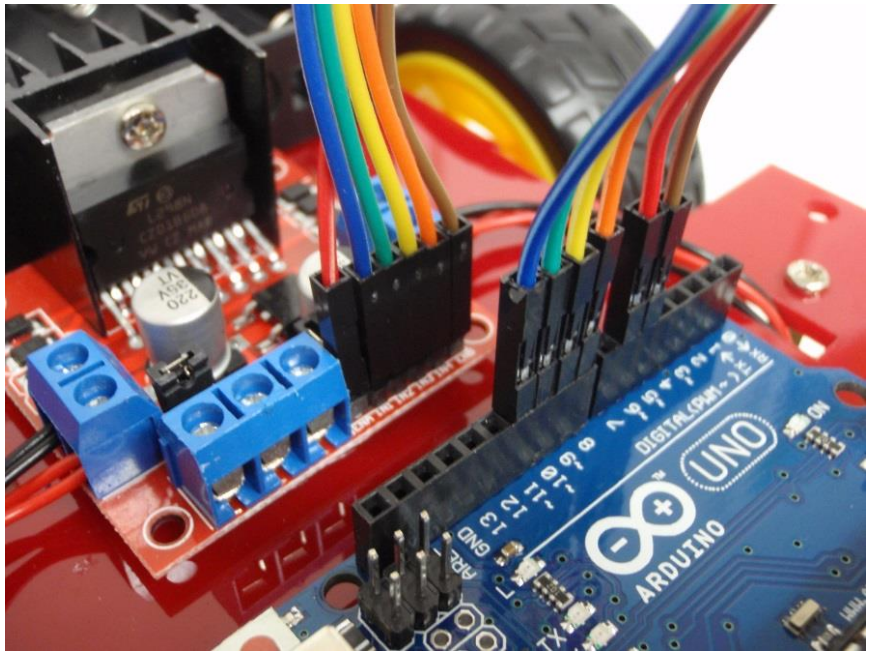


両面テープを利用して
各部品を車体上に実装します

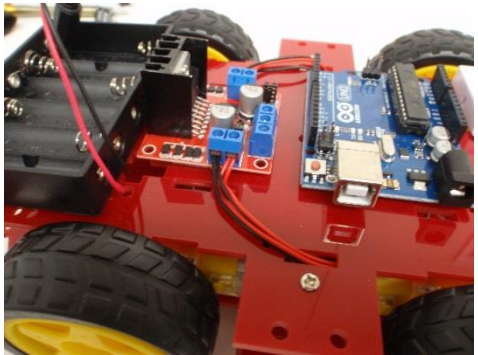
＊あとで接続するコネクタが、タイヤに干渉しないよう、バランスよく配置しましょう







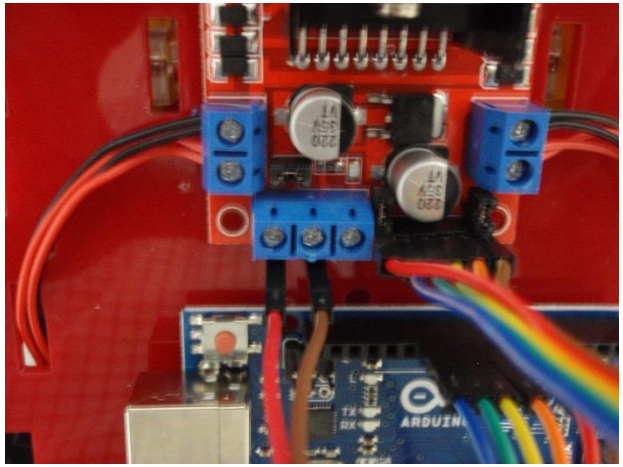
モータードライバー
（ENA, ENB）の
ジャンパーピンは外す

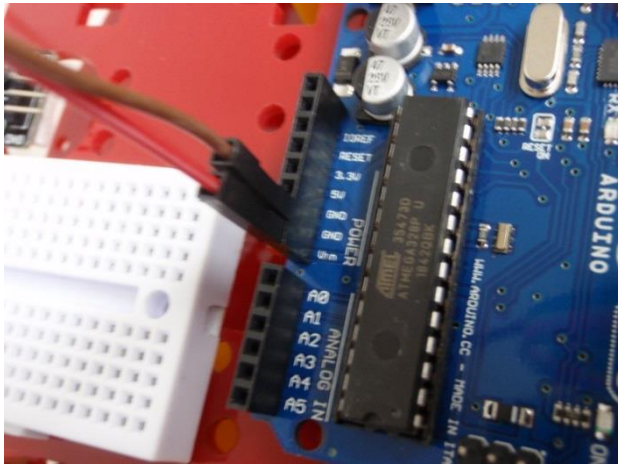


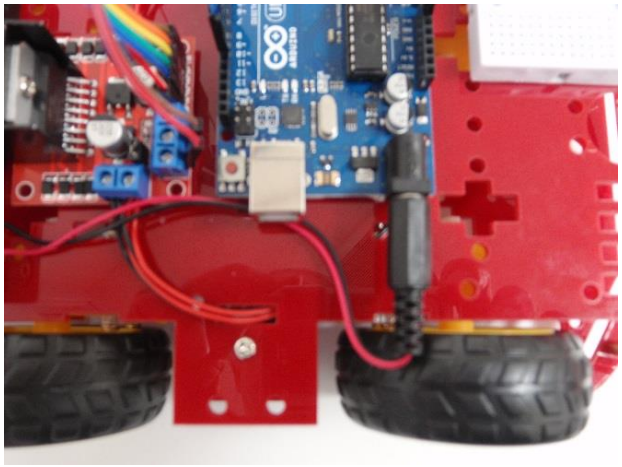
初級者・中級者

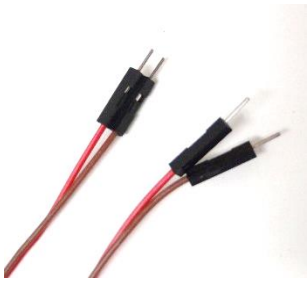
Lesson2 組立（モータードライバー）

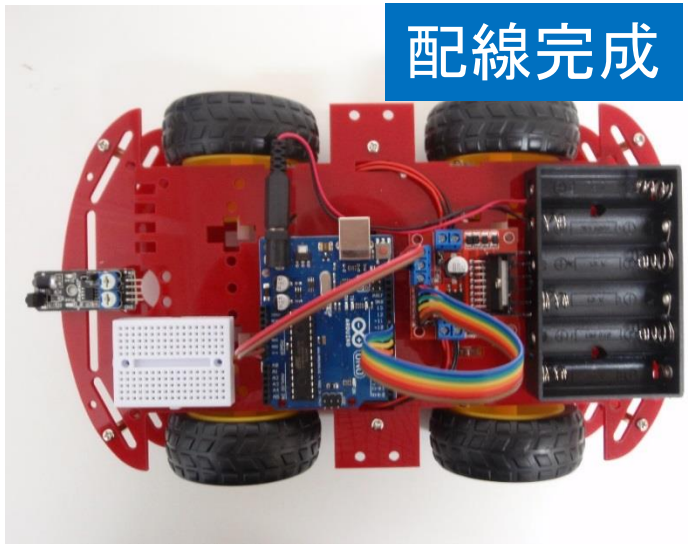
Kernel Concept, Inc.











Lesson2 配線の確認

配線が終わったら、
配線に誤りが無いかを入念に確認しましょう。
この工程は大変重要です。

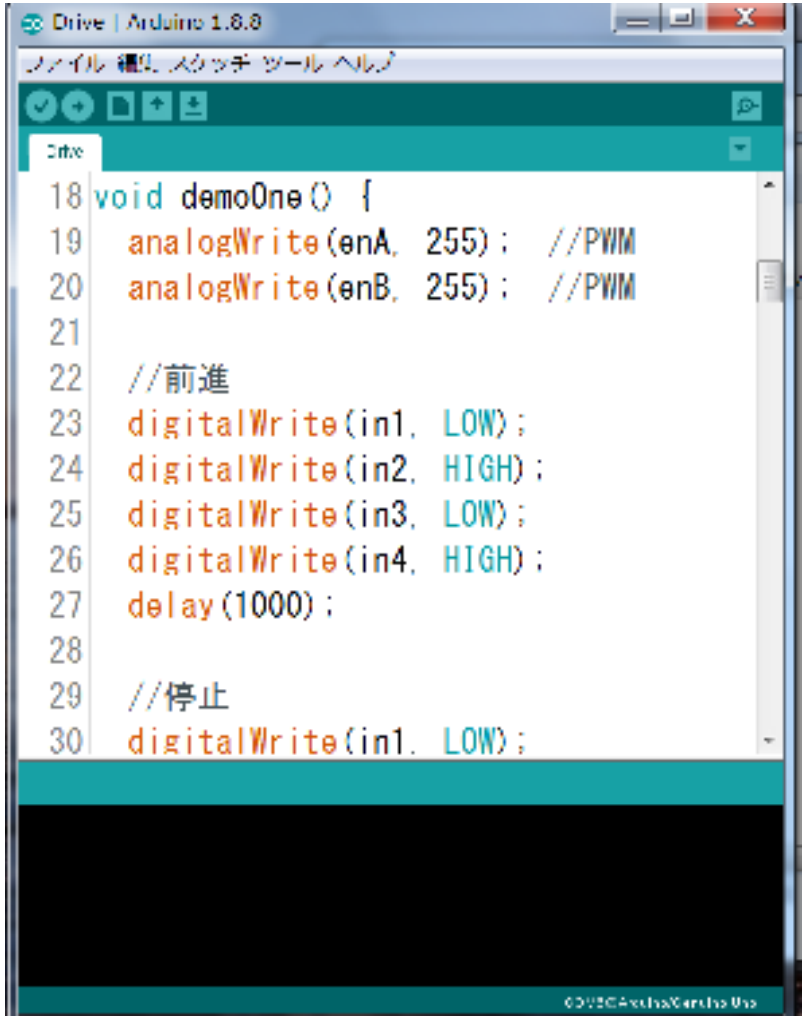
出来れば、他の人に確認してもらってください。
見落としがあるかもしれません。

特に、GNDと5Vの配線には十分気をつけてください。
誤った配線は部品の損傷、故障、発火などにつながります。

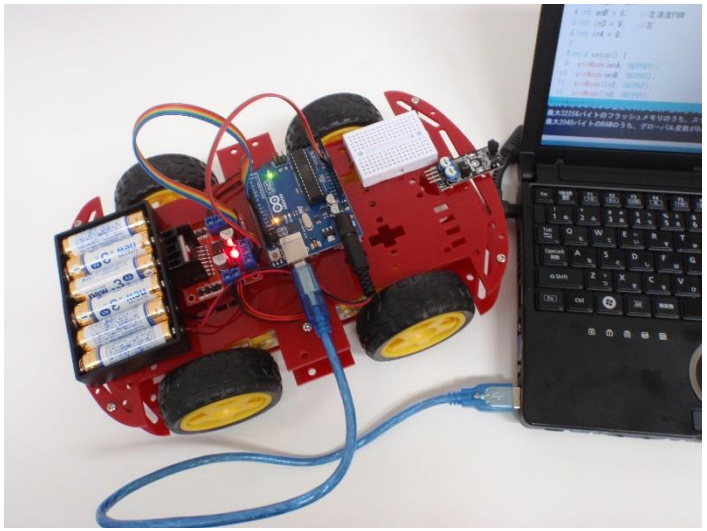
Lesson3 モーター制御

Kernel Concept, Inc.

- サンプルプログラムをPCへ読み込む



付属USBメモリのサンプルプログラム（Drive）をPCへ読み込む（ダブルクリックでもよい）

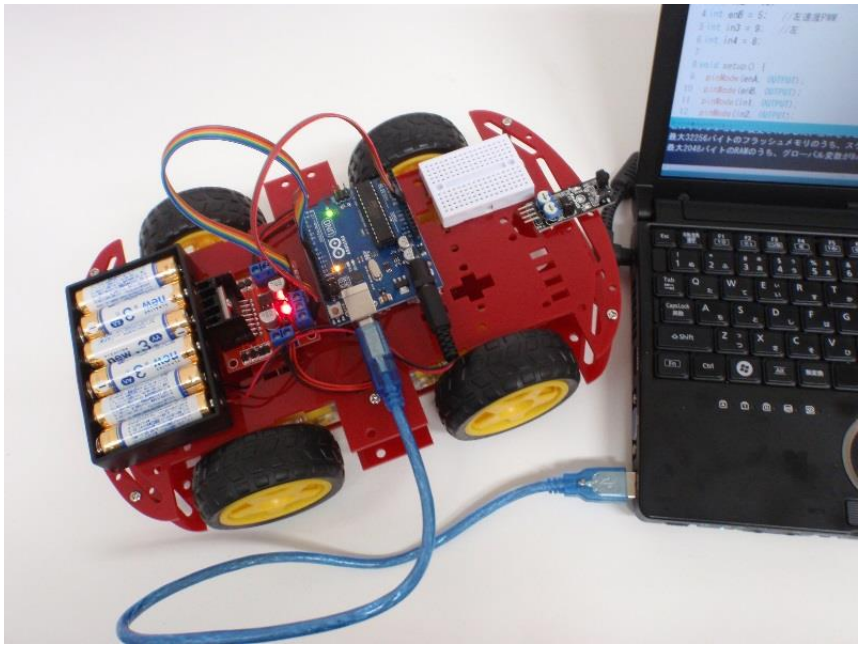


Lesson3 モーター制御

- PCからプログラムをArduinoへ送る

- タイヤが回転

自動車のタイヤが空転するように、ボディーをジャッキアップしておく
前進＞後進＞右旋回＞左旋回＞一時停止＞
　前進スローアップ＞スローダウン＞
　後進スローアップ＞スローダウン
　の繰り返し



乾電池を電池ボックスにセットするとタイヤが回転します。
止めるときは、電池を一つ外してください。

USBケーブルを自動車から外し自動車単独で動作します。

Lesson3 スピード制御

- プログラムの変更

analogWrite(出力先, 値)：出力先を指定した値（電圧）にセットする。
delay(時間ms)：指定しただけ時間待つ。

設定する値が小さい場合には、
モーター駆動の電力不足で
回転しない場合があります。
各モーター特性のばらつきもあります。

左右車輪が最大出力（値225）で正転

```
void demoOne() {
  analogWrite(enA, 255);  //PWM
  analogWrite(enB, 255);  //PWM

  //前進
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(1000);
```

右車輪が中出力（値180）で正転
左車輪が低出力（値100）で正転

```
void demoOne() {
  analogWrite(enA, 180);  //PWM
  analogWrite(enB, 100);  //PWM

  //前進
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(1000);
```

Lesson3 加速制御

- プログラムコマンド（命令）の説明

- for (条件) { コマンド } : 条件が成立中は、{ }内のコマンドを繰り返し実行する。
- int i = 0; i <= 125; i++ : iの値を0から始めて125以下（≦）の間はiを1ずつ増加する。
- Int I : 変数iは整数と定義（integer）。
- i++ : i = I + 1の短縮表記（今のiに1を加えた値をiとする）。

左右車輪が停止状態（値0）から中出力(値125)まで出力増加1で加速

```
//スローアップ

for (int i = 0; i <= 125; i++) {
   analogWrite(enA, i);
   analogWrite(enB, i);
   delay(10);
 }
```

左右車輪が停止状態（値0）から中出力(値125)まで出力増加2で2倍の加速

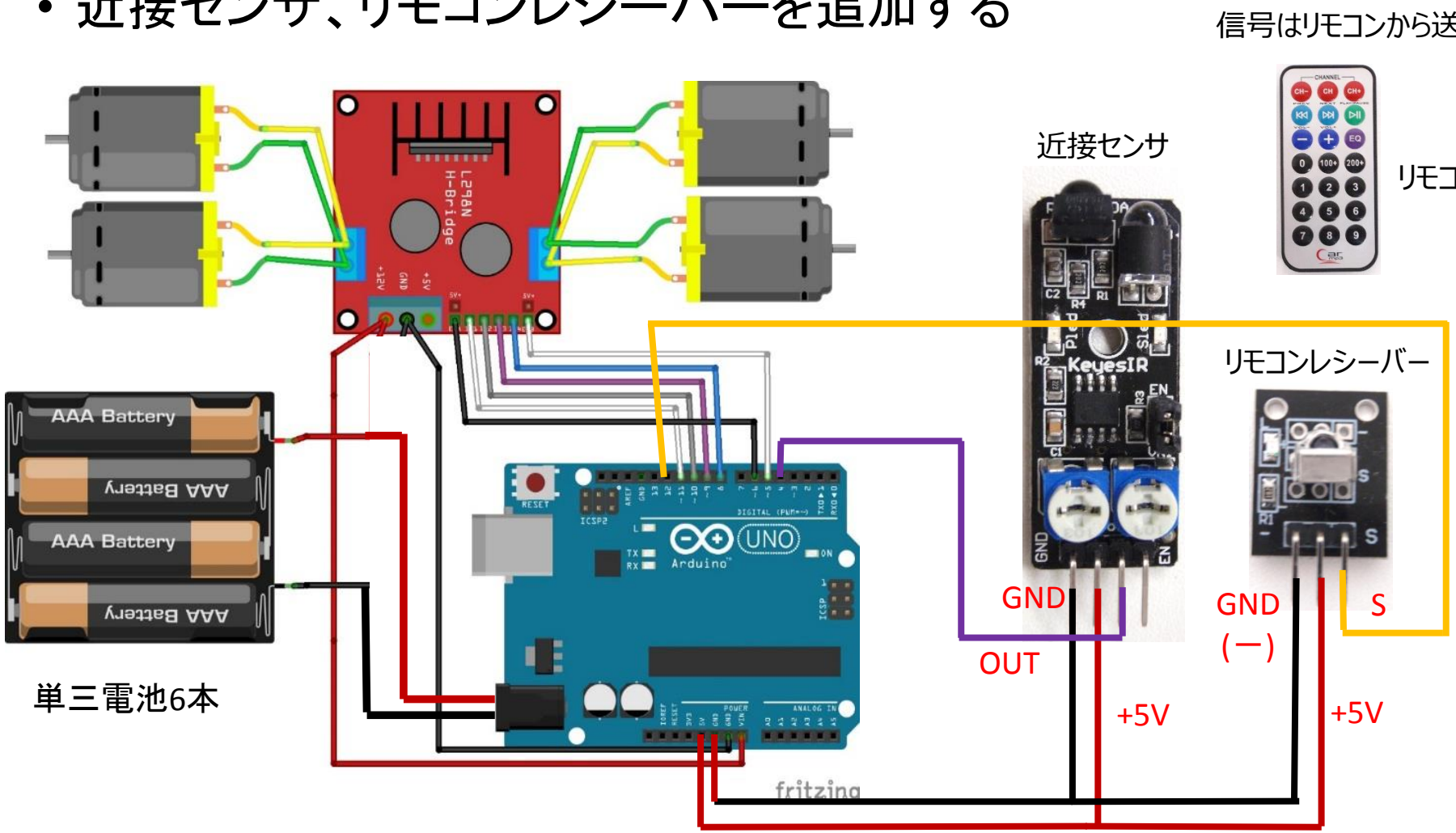
```
//スローアップ

for (int i = 0; i <= 125; i = i + 2) {
   analogWrite(enA, i);
   analogWrite(enB, i);
   delay(10);
 }
```

Lesson4　リモコン制御と自動ブレーキ

Kernel Concept, Inc.

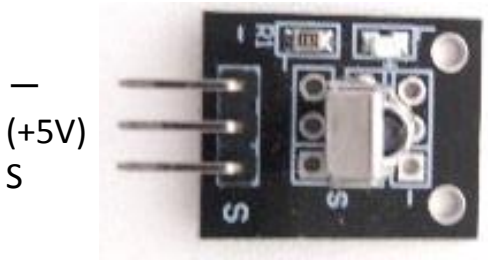
- 近接センサ、リモコンレシーバーを追加する



Lesson4　部品の接続先

Kernel Concept, Inc.

リモコンレシーバー



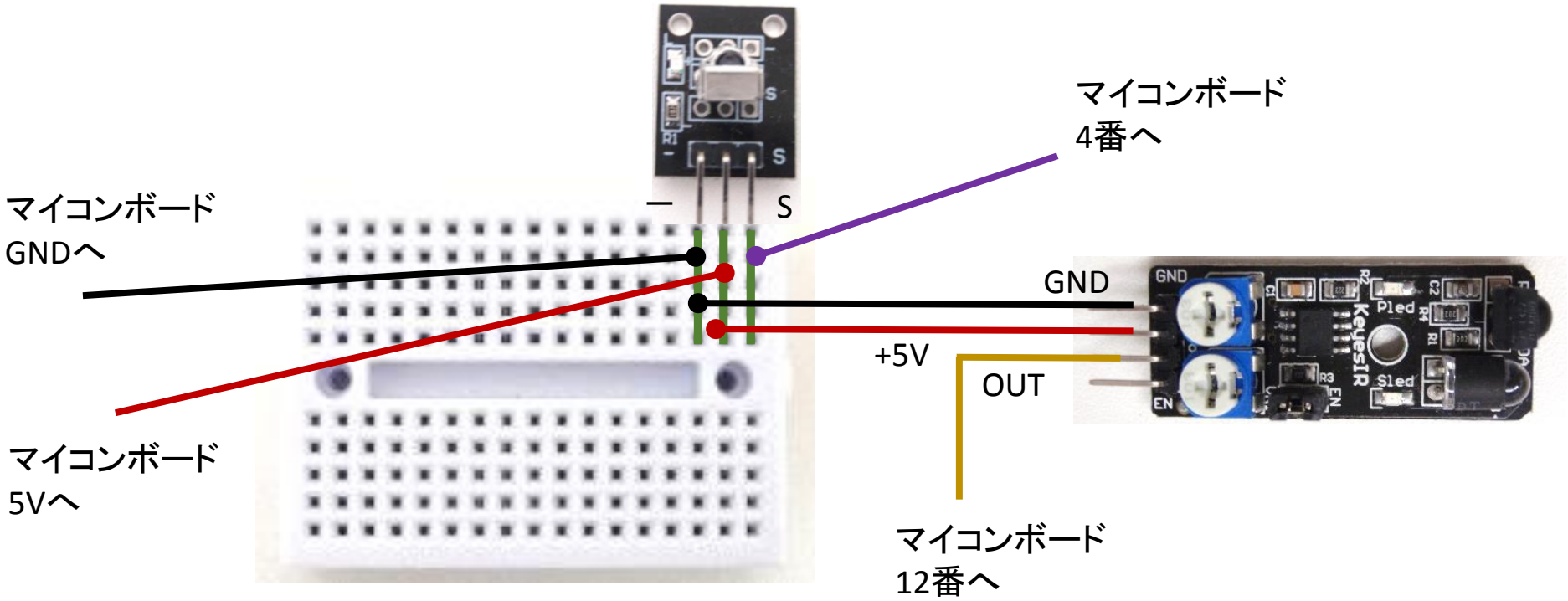
近接センサー

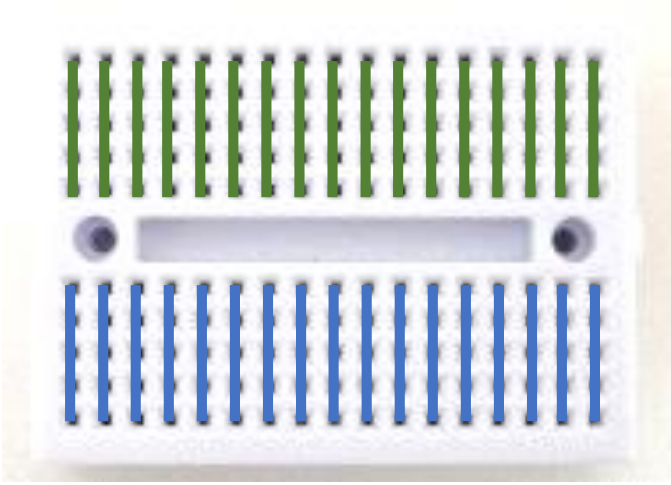
リモコンレシーバー	接続先
－	マイコンGND
(+5V)	マイコン5V
S	マイコン12番

近接センサー	接続先
GND	マイコンGND
+	マイコン5V
OUT	マイコン4番
EN	未使用

Lesson4　部品の配線

Kernel Concept, Inc.





ブレッドボード

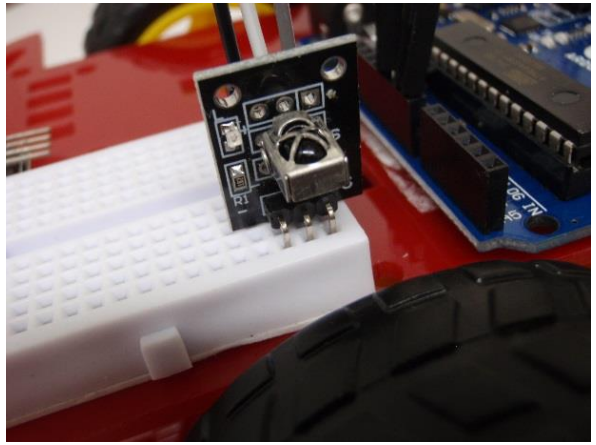
ボード内部では写真の縦方向に
電気的につながっています
（中央を境に、上下部分は独立しています）
内部配線をうまく利用して部品実装しましょう

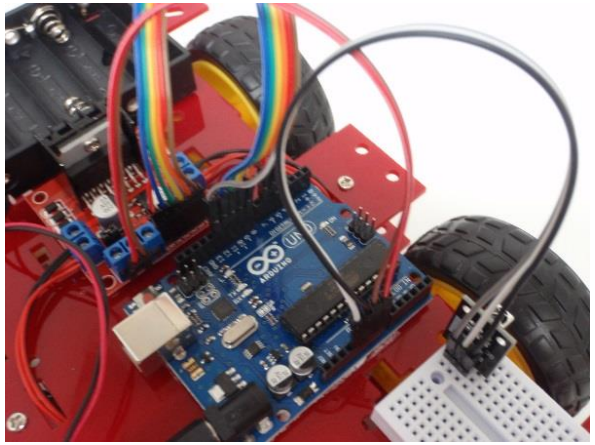
Lesson4　組立（配線）

Kernel Concept, Inc.

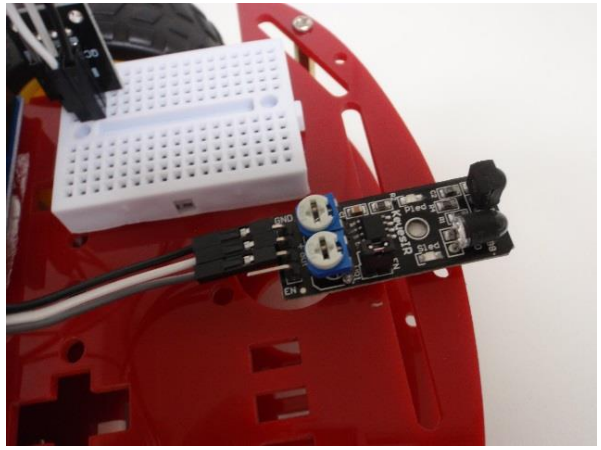
実態配線図と写真を参考に配線しよう

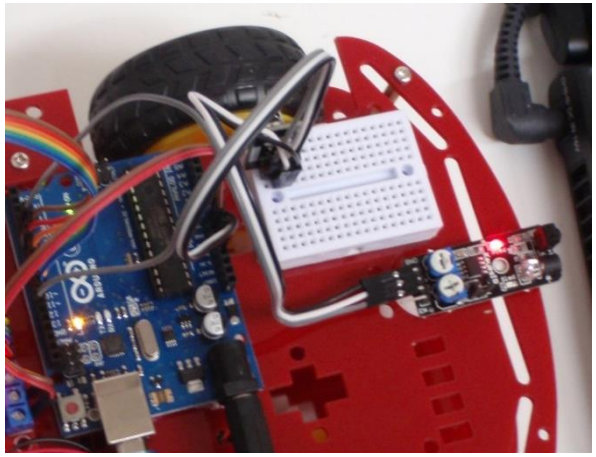
リモコンレシーバー





近接センサー





Lesson4　配線の確認

配線が終わったら、
配線に誤りが無いかを入念に確認しましょう。
この工程は大変重要です。

出来れば、他の人に確認してもらってください。
見落としがあるかもしれません。

特に、GNDと5Vの配線には十分気をつけてください。
誤った配線は部品の損傷、故障、発火などにつながります。

Lesson5　近接センサー調整

Kernel Concept, Inc.

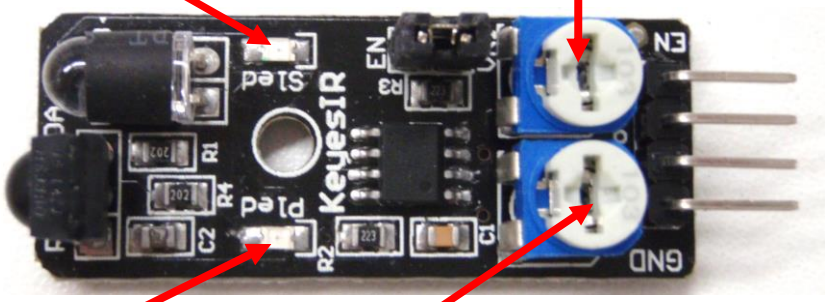
調整は、PCとUSB接続し通電状態でおこないます

近接センサー調整
障害物検知の感度調整を行ってください
太陽光が当たると正しく検知しません

障害物
（白い物）

10cm程度に
接近したらLEDが点灯

感度調整（回して調整）



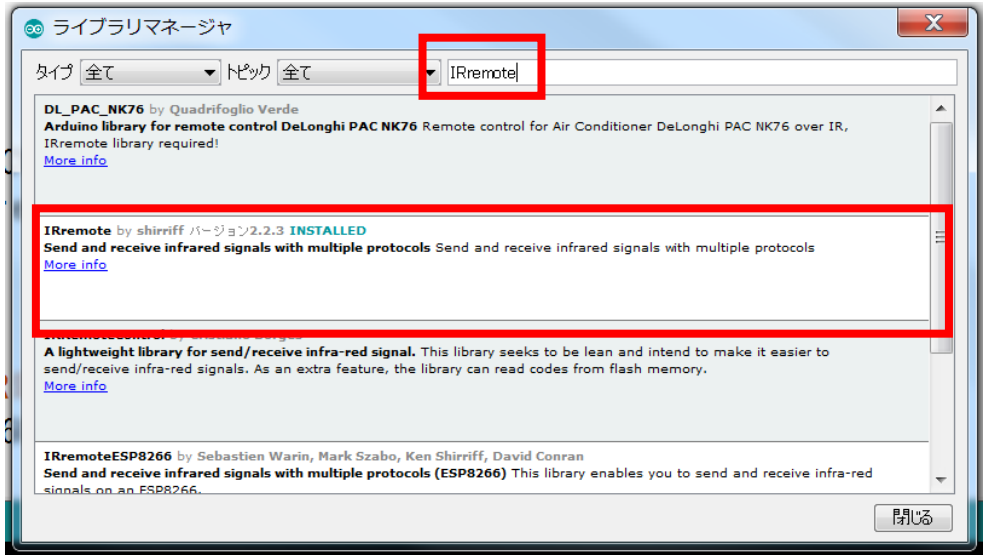
通電中表示LED
常に点灯

検知周期（写真のように中央程度にセット）

Lesson4　リモコン用ライブラリインストール

Kernel Concept, Inc.

プログラムでリモコンを使うために必要な作業です



Arduino IDEプログラムの画面上
メニュー＞ツール＞ライブラリを管理
でライブラリマネージャ画面を開き、検索窓にIRremote
と入力し、表示されたライブラリリストからIRremoteライブラリを
選択してインストールしておく必要があります。

28

Lesson4　リモコン制御と自動ブレーキ

- リモコン制御

付属USBメモリのサンプルプログラム（Drive_IRremote_Rev1）を
PCへ読み込み、マイコンボードへ書き込む

＊自動車のタイヤが空転するように、ボディーをジャッキアップしておく

リモコンボタンに応じて自動車が反応する

リモコンボタンと動作、プログラムとの関連を調べてみよう
自動ブレーキ機能はどのボタンを押した時に動作しているだろうか

リモコンボタンと自動車の動作

1	前進（設定済み時間）
2	後進（設定済み時間）
3	右回転
4	左回転
5	ゆっくり加速前進
6	ゆっくり加速後進
7	ゆっくり前進（自動ブレーキ）
8	ブザー音A
9	ブザー音B

確認時は乾電池をセットしてください

Lesson4　リモコン制御と自動ブレーキ

Kernel Concept, Inc.

- プログラムに手を加えてみよう
  - 赤外センサが障害物に反応したら
    変更前：停止しブザーを鳴らす
    変更後：片側車輪の回転を止める（旋回）

＊動作確認は自動車をジャッキアップして行おう

- プログラムコマンド（命令）の説明
  - If (条件) { コマンドA } else { コマンドB } ：条件が成立したらコマンドAを実行、それ以外はコマンドBを実行する。
  - digitalRead(入力)：入力端子の値の状態を読む。
  - 値A == 値B：値Aと値Bが等しいかどうかを比較する。

```
//定速走行
 int i = Maxspeed;
 for (int n = 0; n <= 1000; n++) {
   if (digitalRead(avoid) == HIGH) {
     analogWrite(enA, i);
     analogWrite(enB, i);
}
   else {
     Breaking();
     Buzz(1000);
     break;
   }
   delay(10);
 }
```

```
//定速走行
 int i = Maxspeed;
 for (int n = 0; n <= 1000; n++) {
   if (digitalRead(avoid) == HIGH) {
     analogWrite(enA, i);
     analogWrite(enB, i);
}
   else {
     analogWrite(enA, Maxspeed);
     analogWrite(enB, 0);
     Buzz(1000);
     break;
   }
   delay(10);
 }
```


30

ここからは中級者コース

ここからは中級レベルです

ある程度経験を積んでからチャレンジしてみましょう

Lesson5　リモコン制御信号

- リモコン信号を調べる

リモコンの各ボタンを押したときにどのような赤外線信号が送出されているか

付属USBメモリのサンプルプログラム（IR_remote）をPCに読み込み、ボードへ書き込む

シリアルモニター（Serial Monitor）の画面を開く

リモコンに対する反応が悪いときは、学習ボードとリモコンの距離や角度をいろいろ変えて試みてください（例えば2m以上）

たとえば、“＋”ボタンを押すと
シリアルモニターに　FF906F　などといった6桁の記号（16進数コード）が表示される
他のボタンも調べて、表にまとめよう。
＊リモコンとリモコンレシーバーの距離や角度によって、正しいコードを受信できない場合があります

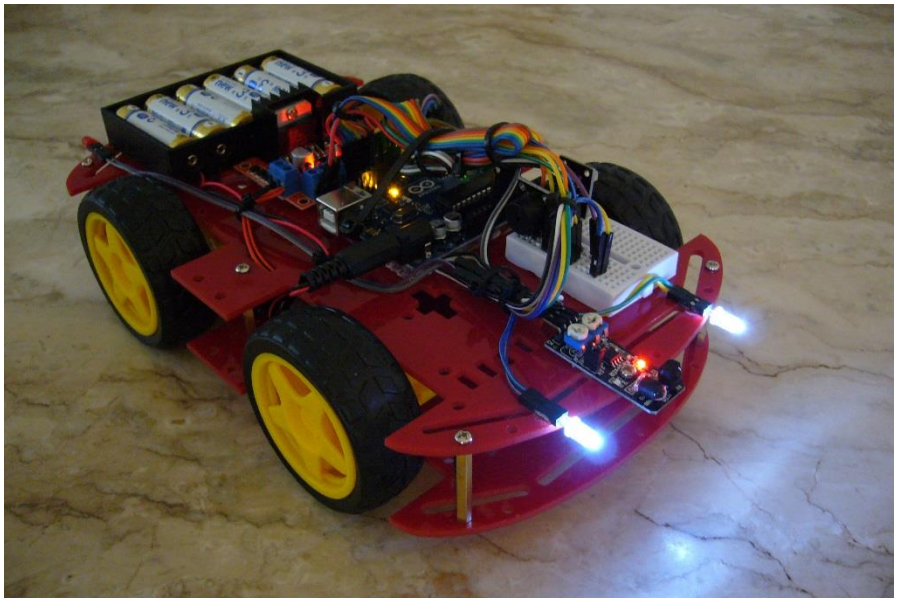
サンプルプログラム（Drive_IRremote_Rev1）のリモコンボタンの割り当てを、自分の好きなボタンへと変えてみよう

Lesson5　応用

- 応用

自動車にホーン（ブザー）やウィンカー、ヘッドライト、ブレーキランプ
などを取り付け制御してみよう

ウィンカーは進路変更時に点滅
ブレーキは減速、停止時に点灯
ヘッドライトはリモコンでON・OFF
など



接続先の例

ブザー	接続先
ー	マイコンGND
（未使用）	
S	マイコン7番
赤LED（ストップ）	
ー	マイコンGND
＋（200Ω直列）	マイコン3番
白LED（ヘッド）	
ー	マイコンGND
＋（200Ω直列）	マイコン2番

LEDの長い足が＋側
200Ω抵抗を介し
マイコン制御端子へ

マイコン3番へ

短い足はマイナス側
マイコンGNDへ

Lesson5　オリジナルプログラム

オリジナルなプログラムを作ろう！

オリジナルな機能の仕様を決定（創造）し、プログラムで実現（プログラミング的思考）してみよう。

はじめは、サンプルプログラムの組み合わせや、改造から始めてみる。

メニュー＞ヘルプ＞リファレンス
Language Referenceを眺めてみると、色々なことができそうです
サンプルプログラムで使っていなかったコマンドや制御をどんどん使ってみよう。

自分で作ったプログラムを勉強会、ブログやSNSで公表、発信して仲間を作ろう！

おわり

プログラムへの表現はオリジナリティあふれ、人それぞれです。見た目の機能は同じでも、プログラムの正解は一つではありません。自由にプログラミングしましょう。