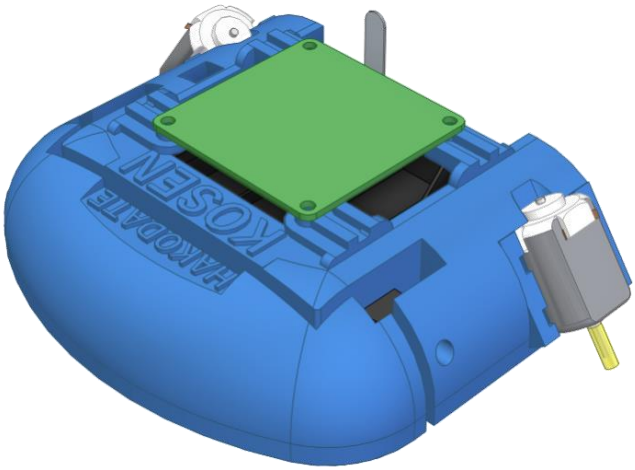
プログラミング体験！～ロボットカーを動かしてみよう～

I. プログラミングロボットについて



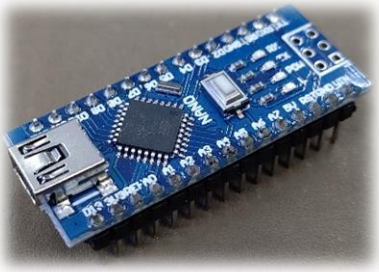
今回使用するプログラミングロボット「KS-2020C」は函館高専 技術教育支援センターが設計し、3D プリンタにより製作されたものです。

ロボットのプログラミングには USB ケーブルでパソコンと接続します。今回の講座では USB メモリにプログラミングをするためのソフトと、お試し用のプログラムが用意されています。自宅に持って帰ってから自宅にパソコンがあれば何度も自分で好きな動きをプログラムして試すことができます。

今回紹介しない難しい機能の使い方については、ちょっと難しいマニュアルも用意しましたので自宅でゆっくり挑戦してみてください。小学生では少し難しい部分もあるので保護者の方と読んでみてください。

もし、USB に入ったプログラムを試しているときにおかしくなってしまったら、函館高専 技術教育支援センターのホームページでもプログラムをダウンロードできるので新しくダウンロードしてみてください。

ロボットにはマイコンボードと呼ばれる頭脳が乗っています。このマイコンボードは上方向に引き抜くことで交換が出来るようになっているので、故障したときや、出来たプログラムを残しておきたいときには同じものを乗せ変えて使用できます。ロボットに使用されているマイコンボードは「Arduino Nano V3.0 （ATmega328P 16MHz)」と呼ばれ、互換品は大手インタネットショッピング等で購入可能です。
※本講座でも互換品を使用しています。



【正規品】

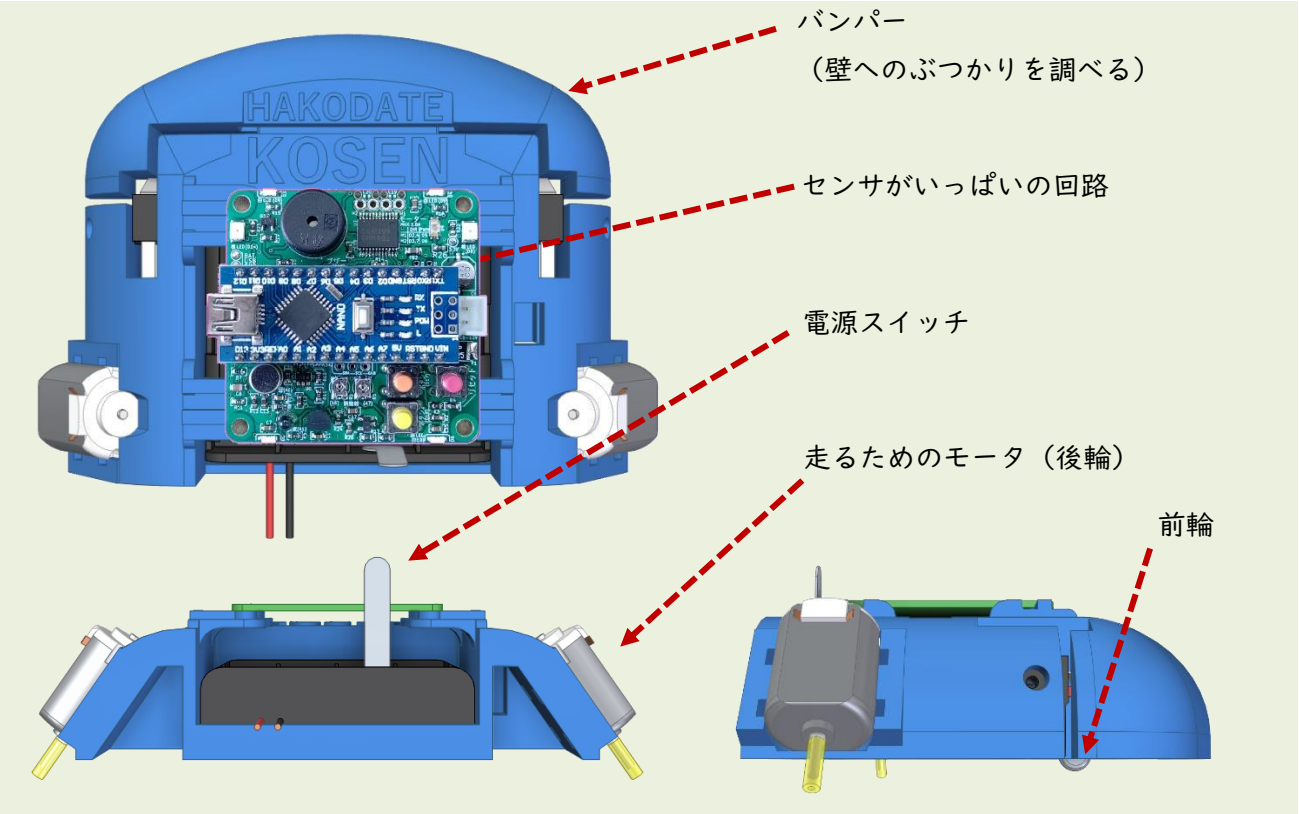
- https://www.switch-science.com/catalog/2554/
- http://akizukidenshi.com/catalog/g/gM-09059/

【互換品】

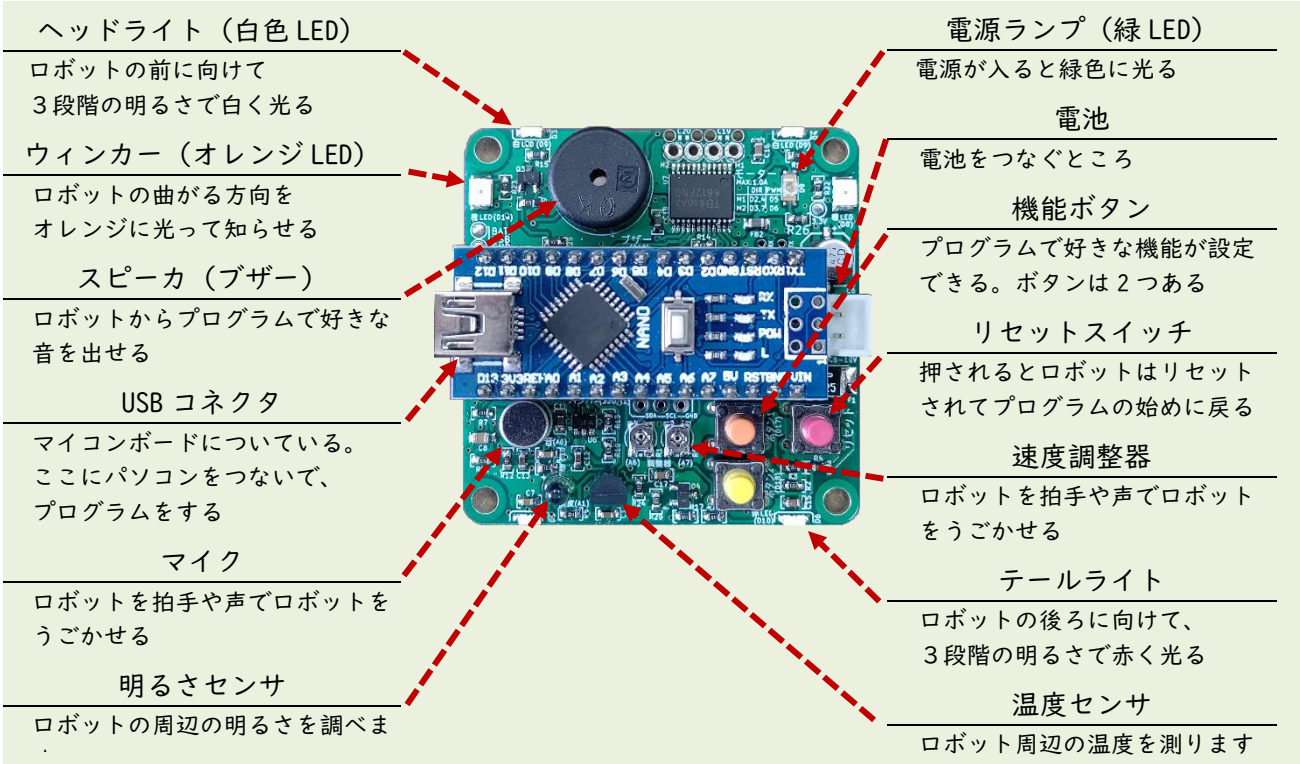
- https://www.switch-science.com/catalog/3524/
- http://akizukidenshi.com/catalog/g/gM-12936/
- https://www.amazon.co.jp/s?k=Arduino+Nano+16MHz&__mk_ja_JP=カタカナ&ref=nb_sb_noss

２．ロボットの各部のなまえと機能

本体



基板（センサ等）



３．パソコンの準備

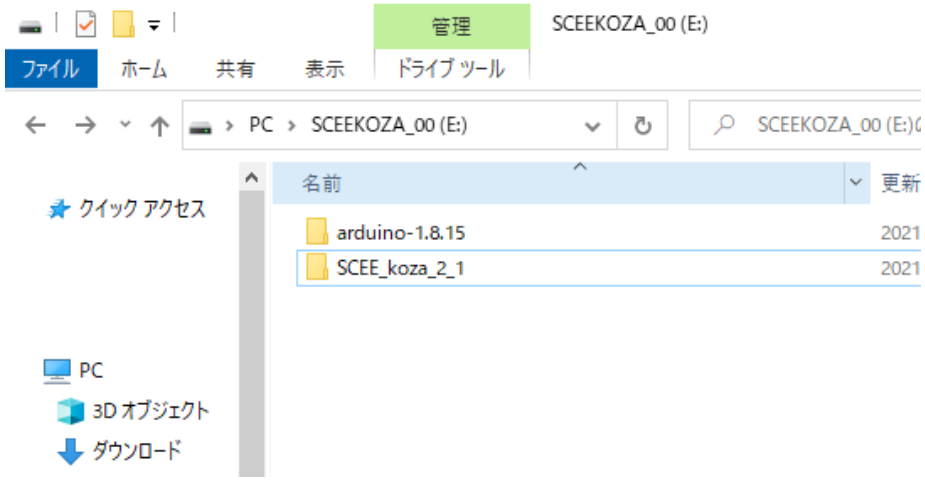
ロボットをプログラムするためのソフトは「Arduino IDE」と呼ばれます。今回はあらかじめ USB メモリに準備しているため、USB メモリをパソコンに接続することでインストールなどの操作は必要なく使用ができます（Windows のみ）。

以下の URL から最新版の Arduino IDE をダウンロードができます。

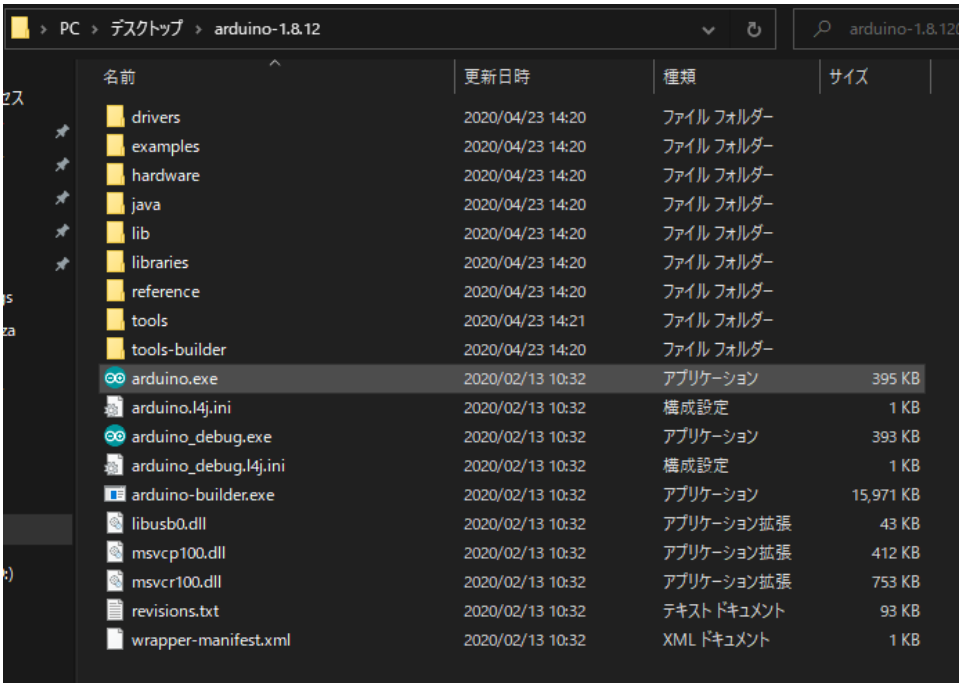
（MacOS X や Linux の場合もこちらからダウンロードが可能です。）

https://www.arduino.cc/en/Main/Software

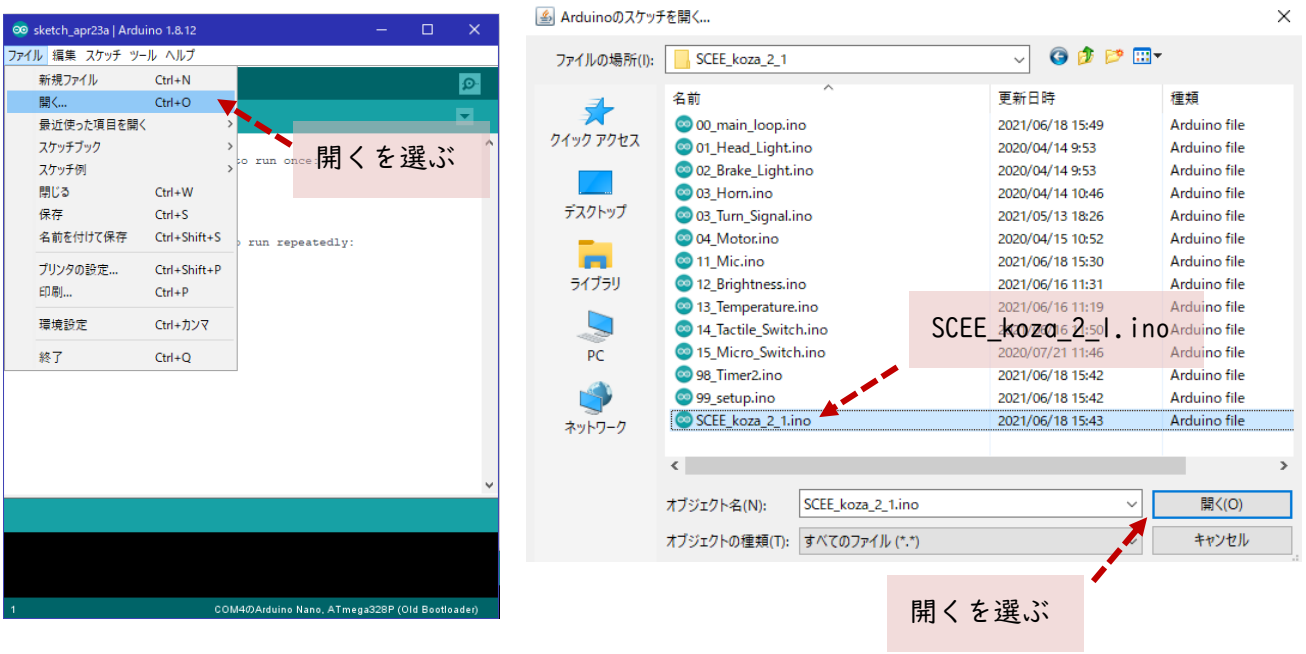
① USB メモリを PC 接続して中身のデータを開きます。



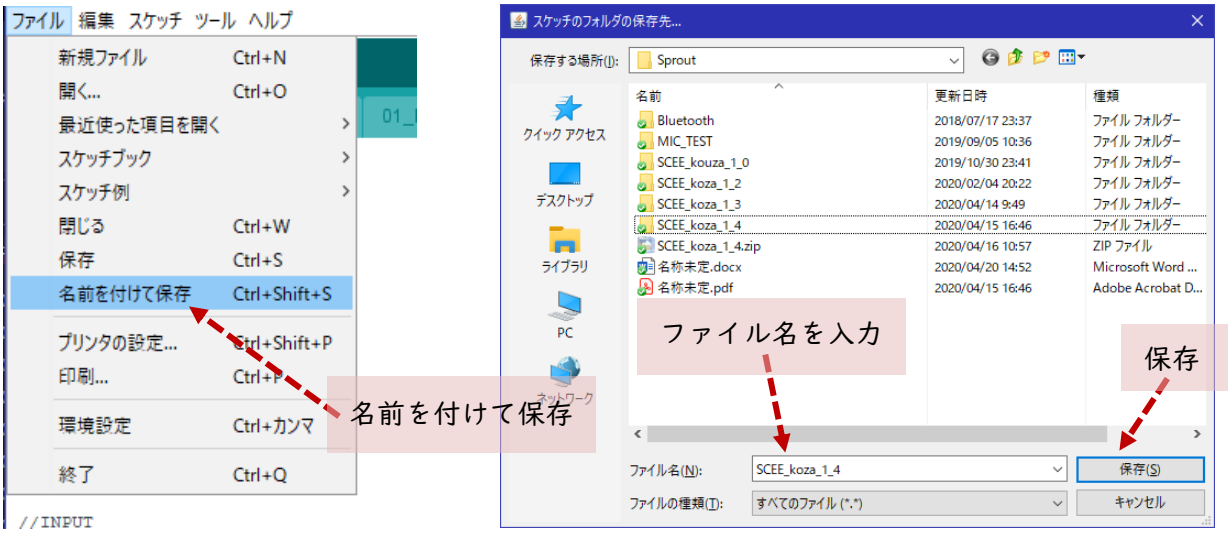
② 「arduino-1.8.15」フォルダを開き、中にある「arduino.exe」を起動します。



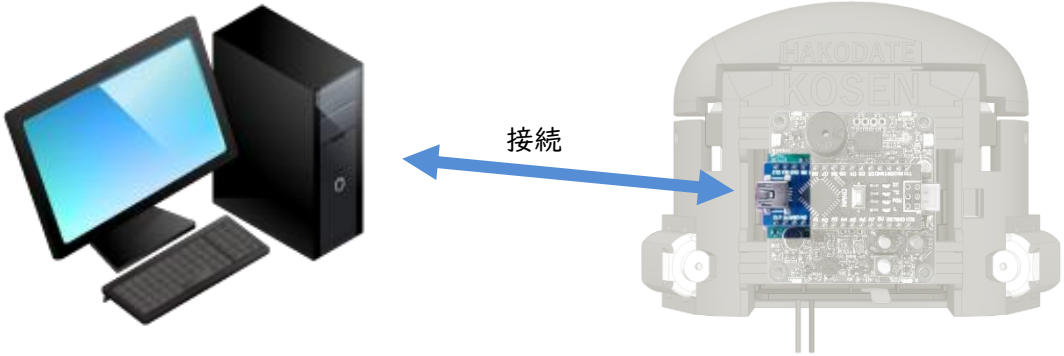
③ 左上の「ファイル」から USB メモリに入っている、「SCEE_koza_2_l」内のロボット用のプログラム「SCEE_koza_2_l.ino」を選びます。



④ これから作るプログラムに名前を付けて保存します。左上の「ファイル」から「名前を付けて保存」を選び、表示される「ファイル名」に英語と数字で好きな名前を付けて保存します。



⑤ ロボットの USB コネクタと PC を USB ケーブルで接続します。



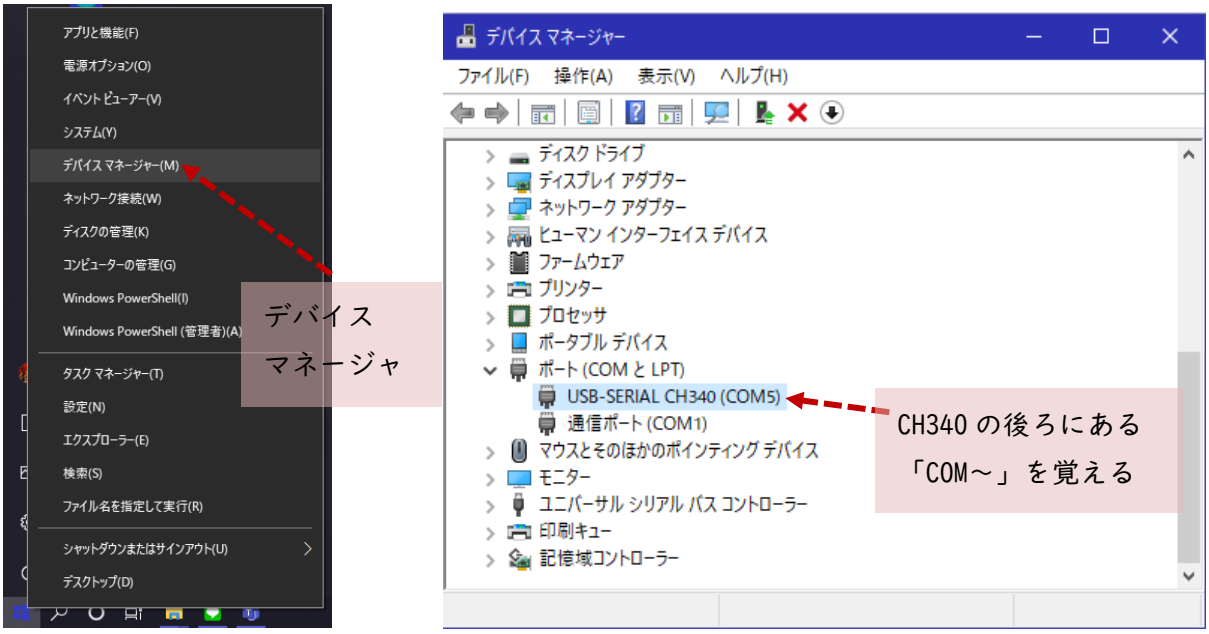
⑥ ロボットが接続されたポートを調べます。

パソコンの左下に表示されている「 Windous」アイコンを、マウスの右ボタンでクリックします。表示されるメニューから「デバイスマネージャー」を選択します。

表示された項目の中から「ポート（COM と LPT)」を選択し表示される「USB-SERIAL CH340（COM～)」と表示される（）内に表示された「COM[番号]」を覚える。

正常にここで COM［番号］が表示されない場合には「⑦ドライバのインストール」を実行してください。正常に表示された場合は⑦を飛ばし⑧に進んでください。

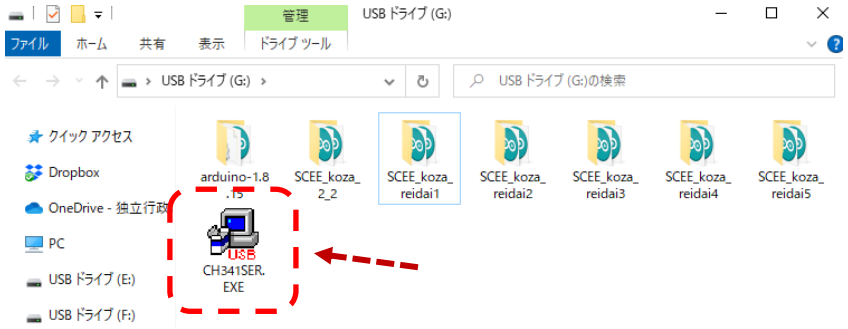
（忘れそうなときはここにメモ：確認した番号は COM__）



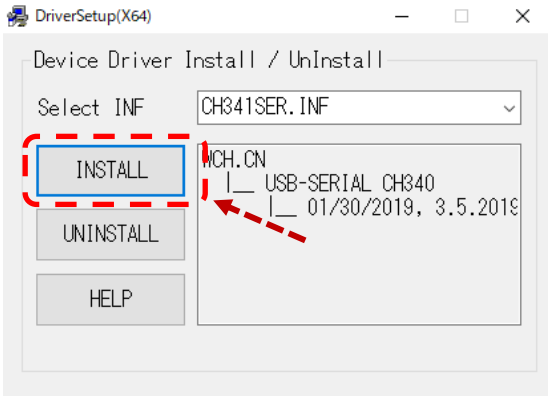
⑦ ドライバのインストール（難しい場合があるので大人にも相談してみたください。）

COM［番号］が⑥の手順で表示されない場合、パソコンにロボットを認識し使用するためのソフト（ドライバ）をインストールする必要があります。

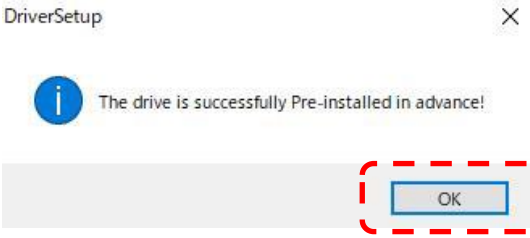
I. ドライバをインストールするには USB に入っている「CH341SER.EXE」を実行してください。実行後変更を許可するかきく画面が表示される場合は「許可」もしくは「はい」を選択してください。



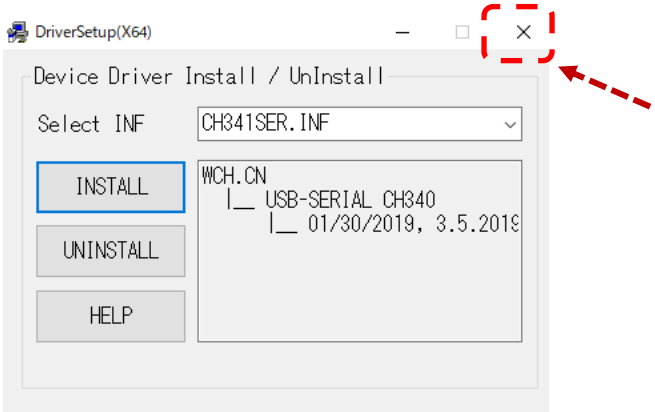
II. ソフトが起動した画面に「INSTALL」と表示されたボタンが表示されるので選択します。



III.インストール完了画面が表示されるので「OK」で通知を閉じます。



IV. 右上の「×」でソフトを終了します。



V. 手順⑥に戻り COM［番号］をもう一度確認してみたください。

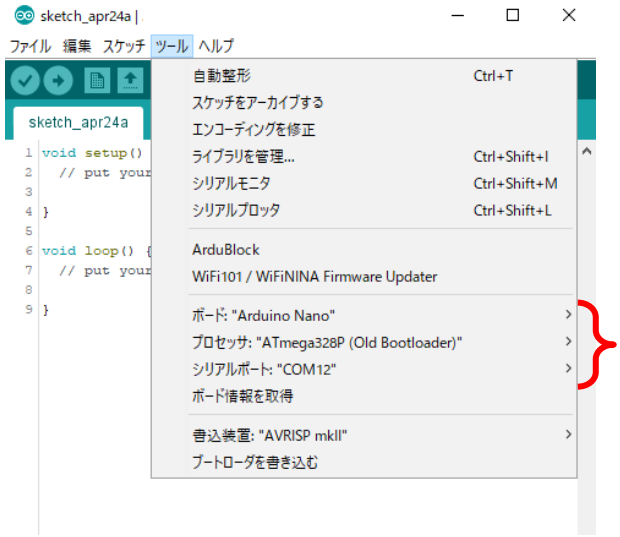
⑧ Arduino IDE に調べたポートとロボットに使用されているマイコンの設定をします。

Arduino IDE のメニューから「ツール」をえらび、次のように設定を変更します。

ボード：　　Arduino Nano

プロセッサ：　ATmega328P　※Old Bootloader ではない

ポート：　　⑥で調べた番号を選ぶ。



設定する

※プロセッサは基板上の青いマイコンを故障などの際に交換して使用する場合、購入時期により「ATmega328P（Old　Bootloader)」の場合もあります。

⑨ ロボットへのプログラムの書き込み

Arduino IDE の上部にある「マイコンボードへの書き込み」によりロボットにプログラムを書き込みます。

【実行・書き込み】
ロボットにプログラムを書き込みます。

【新規作成】
新たに作る際に使用しますが、今回のロボットでは使用しません。

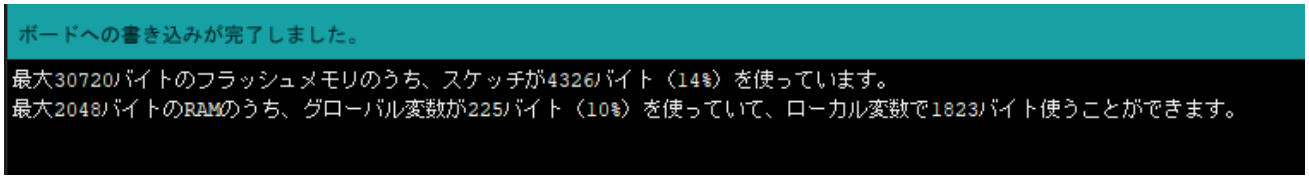
【開く】
保存したプログラムを開くときに使用します。

【検証】
プログラムのミスなどをチェックします。

【保存】
プログラムを保存して中断できます。

⑩ ロボットとパソコンの設定は完了です。

以下のように成功の表示がされればロボットとパソコンの接続は完了です。



もし、その他の表示がされてしまう場合には、下に示すエラーの例を確認してください。

・USB ポートエラー

⑦で設定した内容（マイコン、COM など）に間違いがある可能性があります。

・プログラムコードエラー

プログラムの記述に誤りがあるか、手順⑥以降でファイルが正常に開けていない可能性があります。オレンジの帯に書かれている命令が間違っていないか、③で開いたファイルは正しいかなどを確認してください。よく「；」を忘れている場合があります。

[KS-2020C]プログラミングマニュアル（別冊）　-大人用（仮）-

このマニュアルについて

本マニュアルは函館高専技術教育支援センター主催の公開講座「プログラミング体験！～ロボットカーを動かしてみよう～」に参加され、プログラミングロボット「KS-2020C」をお持ち帰りいただいた参加者の方がご自宅でも楽しむために制作されています。

本マニュアルでは公開講座の中では時間の都合で使用しなかった機能や、説明を省略した応用的な機能について記載されています。より詳しく様々なことを可能とするため、講座の内容よりも専門的な部分、難しい表現や処理が含まれます。小さなお子様が使用される際には保護者の皆様に横から支えていただけますと幸いです。

モータの動作（走る）以外では USB を介した電力で動作可能ですので電池のスイッチが OFF でも試すことが可能です。モータを制御するプログラムが含まれる場合にはスイッチを ON にしてください。

もしも、誤って講座で配布された初期プログラムを改変・削除してしまった場合には函館高専技術教育支援センターHP（https://www.hakodate-ct.ac.jp/~w-scee/）公開講座のページにて再配布しております。再度ダウンロードしてご利用ください。

技術教育支援センターHP

必要となる命令「繰り返し」

プログラムは自分が入力した順番に上から実行されます。基本的にはプログラムは始めから最後まで 1 度だけ実行され終了します。ロボットは同じ動作を繰り返すことで走り続けたり、ライトをつけたり消したりを行います。そのため、1 度で終わらずに繰り返し実行される必要があります。

・基本となる形

```
void main_loop{
        プログラムの内容
                ①
}
```

①プログラムの内容：「{ 」から「 }」までの間にロボットのプログラムを入力することでロボットはプログラムを電源が切るまで、もしくは動作停止命令を実行するまで繰り返し動き続けます。

・基本となる形

```
while( 繰り返す条件   ){
          ①
        プログラムの内容


}
```

① 繰り返す条件：ここに指示した繰り返しの条件を満たす間、{ } 内に記述したプログラムを繰り返し実行します

必要となる命令「待機」

プログラムは入力順に実行されますがロボットに搭載されたマイコンは 1 秒間に約 100 万回以上の命令を実行可能です。したがって、人間による操作や目が認識できる速さに合わせるためには「待機」させることで動きを合わせる必要があります。

・基本となる形

```
delay( 待つ時間の指示   );
          ①
```

① 待つ時間の指示：ロボットに待たせる時間を入力します。
待ち時間は 1/1000 秒（0.001 秒）単位で入力します。
入力する待機時間は**「1（0.001 秒）」～「32000（32 秒）」**までの間で指示が可能です。

必要となる命令「判断」

ロボットには様々なセンサが搭載されています。これらのセンサは周りの状況に応じて様々な変化を数値に変換して取得します。これらの数値を用いて「一定以上の数値であれば走る」、「一定以上の温度になったら音を鳴らす」等の動作が可能となります。ロボットはその際に「一定以上」かどうかを判断する必要があります。

判断には主に 2 種類の方法がありますが、今回のロボットでは「if 文」による判断のみで十分に楽しめますので、「if 文」に絞り記載します。

if 文は「もし～ならば」といった判断をします。

例）　もしスイッチが押されたら・・・
　　もし A 以上なら命令 a、B 以上なら命令 b、C 以上なら命令 c を行う　　等・・・

1 個目の条件の時は「if」と入力しますが、1 個目を満たさない場合で更に条件判断を行う場合には「else if」と入力します。また、どの条件にも一致しない場合を「else」と入力します。2 分岐の場合は「else if」を用いずに「if」と「else」のみを用います。

・基本となる形

```
if( 条件 A  ){
    ②      条件 A を満たしたときの命令（走る、光るなど）
}
else if( 条件 B  ){
        ① 条件 B を満たしたときの命令（走る、光るなど）
}
else{
            条件 A,B を満たさないときの命令
}
```

① 条件：判断の条件を入力します。（講座で用意された命令にはそのまま条件に使用可能なものがあります。）
条件には以下の比較条件が使用可能です。

条件	条件式
A と B が等しい	A == B
A は B より大きい	A > B
A は B より小さい	A < B

条件	条件式
A と B は異なる	A != B
A は B 以上	A >= B
A は B 以下	A <= B

機能１「ヘッドライト（白色 LED）」

ヘッドライトはロボットの前方向に向けて 2 つの白色 LED が点灯します。ヘッドライトは明るさを 3 段階（+消灯）で調整ができます。

一度、点灯したヘッドライトは消すまで点灯します。

・基本となる形

```
raito( 明るさの指示 );
       ①
```

① 明るさの指示：ヘッドライトの明るさを以下から選択し指定します。

明るさ	明るさの指示①部分に入るプログラム
最大（ハイビーム）	ON_3
中くらい（前照灯）	ON_2
最小（幅灯）	ON_1
消す	OFF

・ヘッドライトのプログラムの例

① 最大で光らせる
② 1 秒（1000 ミリ秒）待つ
③ 中間で光らせる
④ 1 秒（1000 ミリ秒）待つ
⑤ 最小で光らせる
⑥ 1 秒（1000 ミリ秒）待つ
⑦ ヘッドライトを消す
⑧ 1 秒（1000 ミリ秒）待つ
⑨ 繰り返す

サンプル：ヘッドライト

```
void main_loop(){
    raito(ON_3);
    delay(1000);
    raito(ON_2);
    delay(1000);
    raito(ON_1);
    delay(1000);
    raito(ON_1);
    delay(1000);
}
```

機能２「テールライト（赤色 LED)」

テールライトはロボットの後ろ方向に向けて 2 つの赤色 LED が点灯します。テールライトは明るさを 3 段階（+消灯）で調整ができます。

一度、点灯したテールライトは消すまで点灯します。

・基本となる形

```
raito_aka( 明るさの指示 );
           ①
```

① 明るさの指示：テールライトの明るさを以下から選択し指定します。

明るさ	明るさの指示①部分に入るプログラム
最大（ブレーキランプ）	ON_3
中くらい（尾灯）	ON_2
最小（尾灯）	ON_1
消す	OFF

・テールライトのプログラムの例

① 最大で光らせる
② 1 秒（1000 ミリ秒）待つ
③ 中間で光らせる
④ 1 秒（1000 ミリ秒）待つ
⑤ 最小で光らせる
⑥ 1 秒（1000 ミリ秒）待つ
⑦ テールライトを消す

サンプル：テールライト

```
void main_loop(){
    raito_aka(ON_3);
    delay(1000);
    raito_aka (ON_2);
    delay(1000);
    raito_aka (ON_1);
    delay(1000);
    raito_aka (ON_1);
    delay(1000);
}
```

機能３「スピーカ（電子ブザー）」

スピーカからはメッセージカード等に使用されている電子音を鳴らすことが可能です。プログラムを試す際にはロボットに搭載されたスピーカの上面の穴をテープ等でふさぐと音が小さくなります。

※音は 2 オクターブ分、用意されていますがマイコンの個体差により多少音程のずれが生じる可能性があります。

※モータと一部マイコンの機能を共有するため、モータにより音階が変化する可能性があります。

・基本となる形

```
oto( 音階の指示,  鳴らす時間 );
       ①          ②
```

① 音階の指示：鳴らしたい音階を以下の表から選択し指定します。

音階（低音）	ド	レ	ミ	ファ	ソ	ラ	シ
①に入るプログラム	do_1	re_1	mi_1	fa_1	so_1	ra_1	si_1
音階（高音）	ド	レ	ミ	ファ	ソ	ラ	シ
①に入るプログラム	do_2	re_2	mi_2	fa_	so_2	ra_2	si_2

② 鳴らす時間：音を鳴らす時間を 1/1000 秒（0.001 秒）単位で入力します。
入力する時間は**「1（0.001 秒）」～「32000（32 秒）」**までの間で指示が可能です。

・スピーカのプログラムの例

ドレミファソラシドを順番に
各 0.5 秒間ならす。

ポイント！

②部分で指定した時間音を鳴らそうと動作しますが、次の命令は②で指示した時間を無視してすぐに実行されます。

そのため「delay(時間)」で鳴っている時間待機させる必要があります。

例）

```
oto( do_1, 500);
delay(500);←鳴らす時間と同じ時間待機させる。
```

サンプル：スピーカ

```
void main_loop(){
    oto( do_1, 500);
    delay(500);
    oto( re_1, 500);
    delay(500);
    oto( mi_1, 500);
    delay(500);
    oto( fa_1, 500);
    delay(500);
    oto( so_1, 500);
    delay(500);
    oto( ra_1, 500);
    delay(500);
    oto( si_1, 500);
    delay(500);
    oto( do_2, 500);
    delay(500);
}
```

機能４「ウィンカー（橙色 LED）」

ウィンカーは大型の橙色 LED で構成されています。ウィンカーはヘッドライトやテールライトとは異なり、明るさの調整はできません。代わりにタイマーと呼ばれる機能を用いて他の機能の使用中でも自動で点滅することが可能となっています。
※モータと一部機能を共有しているため点滅速度に変化が起こる場合があります。

・基本となる形

```
uinka ＝ 左右の指示 ;
         ①
```

① 左右の指示：点滅させたい方向を選択します。

方向	明るさの指示①部分に入るプログラム
両方（ハザードランプ）	RL
右（ウィンカー）	R
左（ウィンカー）	L
消す	OFF

・ウィンカーのプログラムの例

① ハザードランプを点ける
② スイッチ I がおされたら OFF
③ 押されていなかったら ON のまま
④ 1 秒（1000 ミリ秒）待つ

サンプル：ウィンカー

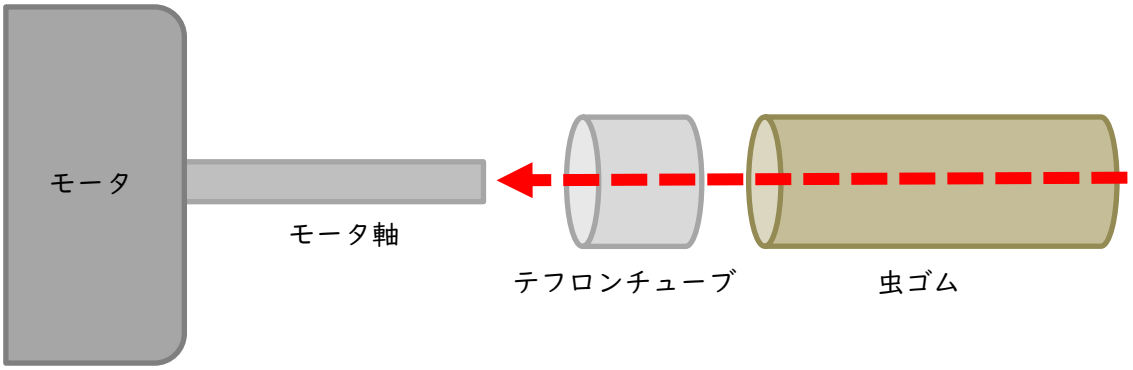
```
void main_loop(){
    if( botann(1) ){
        uinka = OFF;
    }
    else{
        uinka = RL
    }
    delay(1000);
}
```

機能５「走る（モータ）」

ロボットには 2 つのモータが搭載されていて平らな面を走行することが可能です。モータはロボットのエネルギーを多く使用するため動作させるためには電池のスイッチを ON にする必要があります。モータは左右で別々の動作が可能となっており様々な動きが可能です。

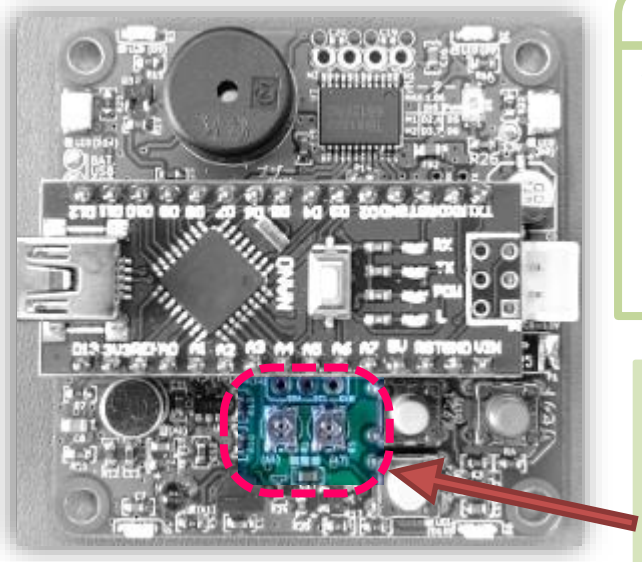
・交換部品について

モータには自転車用の虫ゴムを滑り止めに取り付けてあります。摩耗した場合や紛失した際には交換してください。交換の際には虫ゴムのほかにモータとの摩擦を減らすためにテフロンチューブと呼ばれる半透明の部品が付いています。交換の際には紛失しないように注意してください。



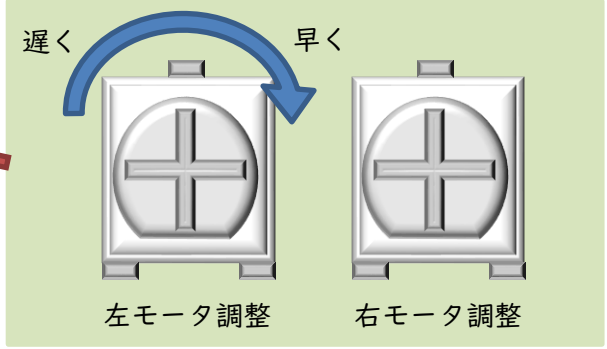
・モータの速度調整

モータの回転速度は基板上の部品を＋ドライバーで回すことで調整が可能です。以下のプログラムをロボットにプログラムすることで 1 秒ごとに前進、停止を繰り返します。速度調整並びに、直進するように左右のバランスを整えてください。



サンプル：速度調整

```
void main_loop(){
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( sayuu, teisi);
    delay(1000);
}
```



・基本となる形

```
hasiru( 回転させるモータの指示,  回転方向の指示 );
          ①                      ②
```

① 回転させるモータの指示：回転させる側のモータを指定します。

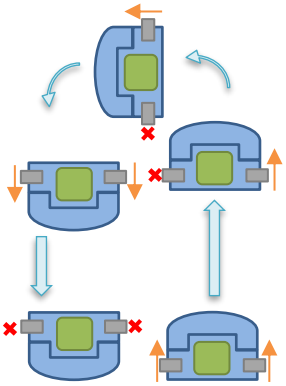
回転させたいモータ	①部分に入力するプログラム
両方（左右）のモータ	sayuu
右のモータ	migi
左のモータ	hidari

② 回転方向の指示：モータに対する回転を指示します。

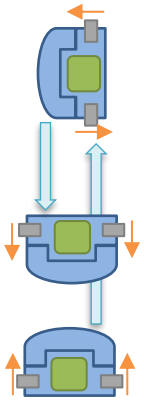
回転（動作）方向	②部分に入力するプログラム
前に進む	mae
後ろに下がる	usiro
停止（ブレーキ）	teisi
空転（ニュートラル）	OFF

・モータのプログラムの例

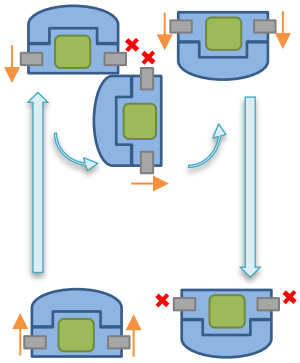
例）左おお回り旋回



例）その場で左旋回



例）右後方に切り返し旋回



サンプル：左おお回り旋回

```
void main_loop(){
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( migi, mae);
    delay(800);
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( sayuu, teisi);
    delay(1000);
}
```

サンプル：左その場旋回

```
void main_loop(){
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( migi, mae);
    hasiru( hidari, usiro);
    delay(500);
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( sayuu, teisi);
    delay(1000);
}
```

サンプル：右切り返し旋回

```
void main_loop(){
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( hidari, usiro);
    delay(800);
    hasiru( sayuu, mae);
    delay(1000);
    hasiru( sayuu, teisi);
    delay(1000);
}
```

便利な機能の紹介

ここから先で紹介する機能はロボットが周りの状態を知るためのセンサーとなります。ロボットに搭載された、いくつかのセンサーを使用することでロボットは複雑な動作や環境に合わせた判断が可能となります。

ただし、センサーの結果は電気の信号となっていて人間には目で見てわかりません。そこでロボットをコンピュータに接続することでその時のロボットの状態を 1 秒ごとに表示する機能を搭載しています。ここでは表示機能について、これからセンサーの紹介をする前に説明します。

・基本となる形

```
jyouhou ＝ 機能の ON/OFF 指示;
           ①
```

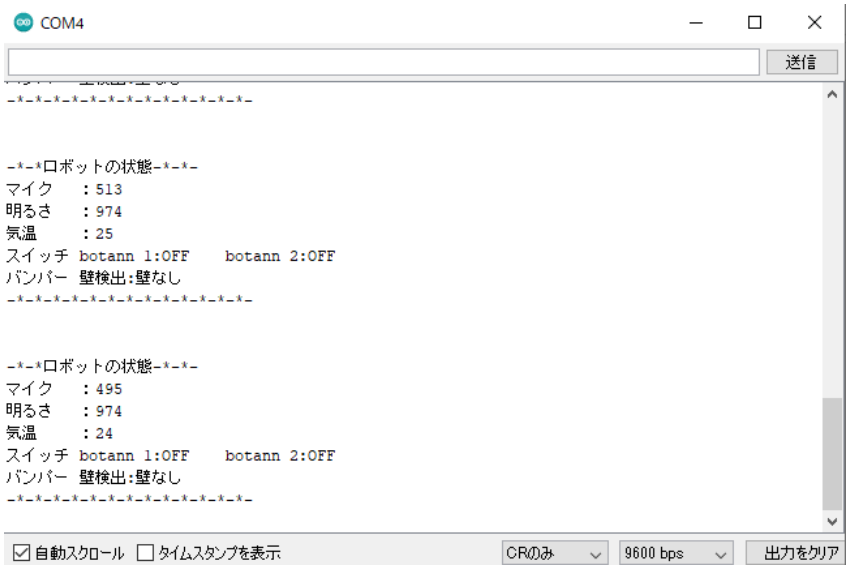
① ロボットの状態を表示させるかどうかを指示します。

機能の ON/OFF	①部分に入力するプログラム
有効：ON	ON
無効：OFF	OFF

00_main_loop で「ON」に設定することでコンピュータへ情報を送信するようになります。この機能は若干の動作に遅延（0.001 秒程度）を発生させるため初期では OFF となっています。

プログラム内で有効化し、開発ソフト「Arduino IDE」の右上にある虫眼鏡の形をしたアイコン「シリアルモニター  」を**ロボットを接続した状態**でクリックします。新たに COM～と表示された枠が現れ表示を開始します。

シリアルモニターのアイコン



ロボットからの状態の表示

機能６「音に反応（マイク）」

ロボットにはマイクが搭載されて周囲の音（大きな声や拍手）に反応して動くことができます。周囲の音の大きさを 0～1023 の数値で聞き取ります。
※ロボットが走る、音を鳴らすなどにより雑音が発生するため反応する感度に余裕を持っておく必要があります。

・基本となる形

```
kiku( 反応する感度 );
```
①

② 反応する感度：０～1023 までの幅で任意の反応させたい音量を選びます。
設定した値よりも測定した値が大きいときは「１」小さいときは「０」となります。
※通常の状態でも常に音を拾っており上限、下限付近はほぼ使用しません。
※ロボットの企画段階では 450 くらいで息を吹きかけたときに反応が得られました。

・マイクのプログラムの例①

マイクが反応するかチェックする。通常時にはブレーキランプを点灯させておき、マイクの反応があった時のみヘッドランプを 1 秒点滅させる。
備考：if 文による判断を使用。

サンプル：マイクチェック

```
void main_loop(){
    if(kiku(450)){
        raito(ON_2);
        raito_aka(OFF);
        delay(1000);
    }
    else{
        raito(OFF);
        raito_aka(ON_2);
    }
}
```

・マイクのプログラムの例②

マイクが反応するまで待機して、マイクに音が入り反応したらロボットが走り出す。

サンプル：マイクチェック

```
void main_loop(){
    static int ugoki = 0;
    if(kiku(450)){
        ugoki = 1;
    }
    else{;
        ugoki = 0;
    }
    While(ugoki == 1 ){
        hasiru(sayuu, mae);
    }
}
```

機能７「明るさ検出（照度センサ）」

搭載された照度センサにより周辺の明るさを測定することができます。周囲の明るさを 1～1023 の数値で測定します。センサの変化は人間の目と近い感度を持っています。

・基本となる形

```
akarusa( 反応する感度 );
```
①

① 反応する感度：1（暗）～1023（明）までの幅で任意の反応させたい明るさを選びます。
設定した明るさよりも暗い場合には「１」明るい場合には「０」が得られます。

※通常の状態でも常に微小な明るさがあるので上限、下限付近はほぼ使用しません。

・明るさ検出のプログラムの例①

センサ反応する幅の中央（500）よりも周囲が暗くなったらヘッドランプとテールランプを点灯する。
備考：if 文による判断を使用。

サンプル：明るさ判断

```
void main_loop(){
    if(akarusa(500)){
        raito(ON_2);
        raito_aka(ON_2);
    }
    else{
        raito(OFF);
        raito_aka(OFF);
        delay(50);
    }
}
```

・明るさ検出のプログラムの例②

現在の部屋の明るさを PC で確認する。
現在の明るさを USB 経由で PC に転送します。
値を確認することで、その環境における明るさの動作値を決める際などに使用できます。

プログラムを書き込み後、開発ソフトの右上にある「シリアルモニター  」をクリックします。
※マイクでも同様に PC で確認できます。

サンプル：明るさを調べる

```
void main_loop(){
    akarusa(500, 'T');
}
```

使用する予定の明るさの値を入れておくとそれよりも明るければ「0」暗ければ「I」で表示もされます。

機能８「温度計（温度センサ）」

搭載された温度センサで室内の温度を測定します。
測定される温度は大まかな温度となっており、ロボット全体の使用する電気の量で 3℃程度誤差があります。

・基本となる形

```
ondo(反応する気温);
     ①
```

① 反応する温度：5℃～40℃までの幅で任意の反応させたい気温を選びます。
設定した気温よりも暖かい場合には「１」寒い場合には「０」が得られます。

・温度測定のプログラムの例①

指でセンサーをつまんで 0.5 秒ごとに温度を測定し 28℃を超えたらスピーカーから音を出す。

サンプル：温度測定

```
void main_loop(){
    if( ondo(28) ){
         oto( do_1, 500);
         delay(500);
    }
    else{
         delay(500);
    }
}
```

・温度測定のプログラムの例②

現在のロボットの測定した温度を表示する。
現在の温度と気温を USB 経由で PC に転送します。
値を確認することで、その環境における明るさの動作値を決める際などに使用できます。
プログラムを書き込み後、開発ソフトの右上にある「シリアルモニター  」をクリックします。
※マイクでも同様に PC で確認できます。
温度：ロボットが認識する値
気温：人間にわかるように℃に直した値

サンプル：温度測定

```
void main_loop(){

    ondo('T');

}
```

機能９「機能ボタン（押しボタンスイッチ）」

ロボットには自由に機能を割り当て可能なボタンが 2 つ搭載されています。押したらクラクションを鳴らす。走りだす。ライトをつけるなど利用方法は自由に使用可能です。

・基本となる形

```
botann( ボタン No. );
          ①
```

① ボタン番号：ボタンは 2 つ搭載されているので「１」もしくは「２」の押されているか調べたいボタンを選択します。押されていれば「１」押されていなければ「０」が返されます（botann(-)が「0」もしくは「1」に置き換わるイメージ)。
返される値を if 文等で比較して利用します。

・スイッチのプログラムの例①

ボタン１が押されたら 1 秒間音を鳴らし、
ボタン 2 が押されたら、押されている間
ヘッドランプを点灯する。
それ以外の時はライトを消し何もしない。

備考：if 文による 3 分岐を使用。

サンプル：ボタン機能

```
void main_loop(){
      if( botann(1) ){
         oto( do_1, 500);
      }
      else if( botann(2) ){
         raito( ON_2 );
      }
      else{
         raito( OFF );
      }
}
```

・スイッチのプログラムの例②

ボタン１を押すまで待機し、ボタン 1 が
押されたら走りだす。走っているときに
ボタン２が押されたら再び停止する。

サンプル：スイッチで動く

```
void main_loop(){
    static int ugoki = 0;
    if( botann(1) ){
         ugoki = 1;
    }
    else{;
         ugoki = 0;
    }
    While(ugoki == 1 ){
         hasiru(sayuu, mae);
         if( botann(2) ){
             ugoki = 0;
         }
         else{;
              ugoki = 1;
          }
    }
}
```

機能１０「バンパー（壁検出)」

ロボットの前面には 2 つのスイッチが付いたバンパーが搭載されており、壁に一定速度以上で衝突した際に、正面、左右どの方向が衝突したかを検出することが可能です。

・基本となる形

```
kabe();
```

実行すると壁にぶつかっている場合、表の値が返ってきます（数値に kabe()が置き換わるイメージ)。返される値を if 文等で比較して利用します。

状態	返される値
衝突なし	0
左側面検出	1
右側面検出	2
正面検出	3

・バンパーのプログラムの例
壁に当たったら逆の方向に動く。
正面⇒後ろに下がって U ターン
右側⇒後ろに下がって左前方に回避
左側⇒後ろに下がって右前方に回避

サンプル：バンパー

```
void main_loop(){
    if(kabe() == 3 ){
        hasiru(sayuu, usiro);
        delay(500);
        hasiru(migi, usiro);
        hasiru(hidari, mae);
        delay(500);
    }
    else if(kabe() == 2){
        hasiru(sayuu, usiro);
        delay(500);
        hasiru(migi, mae);
        delay(500);
    }
    else if(kabe() == 1){
        hasiru(sayuu, usiro);
        delay(500);
        hasiru(hidari, mae);
        delay(500);
    }
    else{
        hasiru(sayuu, mae);
    }
}
```