きわめる。拓く。創り出す。
長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習

工学基礎実験

評価方法

- **すべての講義に出席**
- 各週の課題の達成度（教員・TAに確認してもらう）
  - 基本課題：～A
  - 応用課題：S
  - 挑戦課題：基本課題、応用課題がすべて終わった人向け

※プログラミングは自分で考えないと出来るようにならない！

※プログラムをコピーしない！

※プログラムをコピーさせない！

※わからないことがあったらTAを呼ぶ（答えではなくヒントをもらう）

ガイダンス

- 第1週
  - 組み込みプログラミングの導入
  - シリアル通信
- 第2週
  - デジタルIO（Output）
  - デジタルIO（Input）
- 第3週
  - PWM制御
  - Analog Digital変換

この資料について

- 目次
  - 各講義でやる内容を案内
- 解説
  - 教員が要素技術などを説明
- 例題
  - 教員と一緒に試す課題
- 演習（基本）
  - 各自が最低限達成するレベルの課題
- 演習（応用）
  - 基本が終わった人が取り組む課題

きわめる。拓く。創り出す。
NAS 長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.1-1

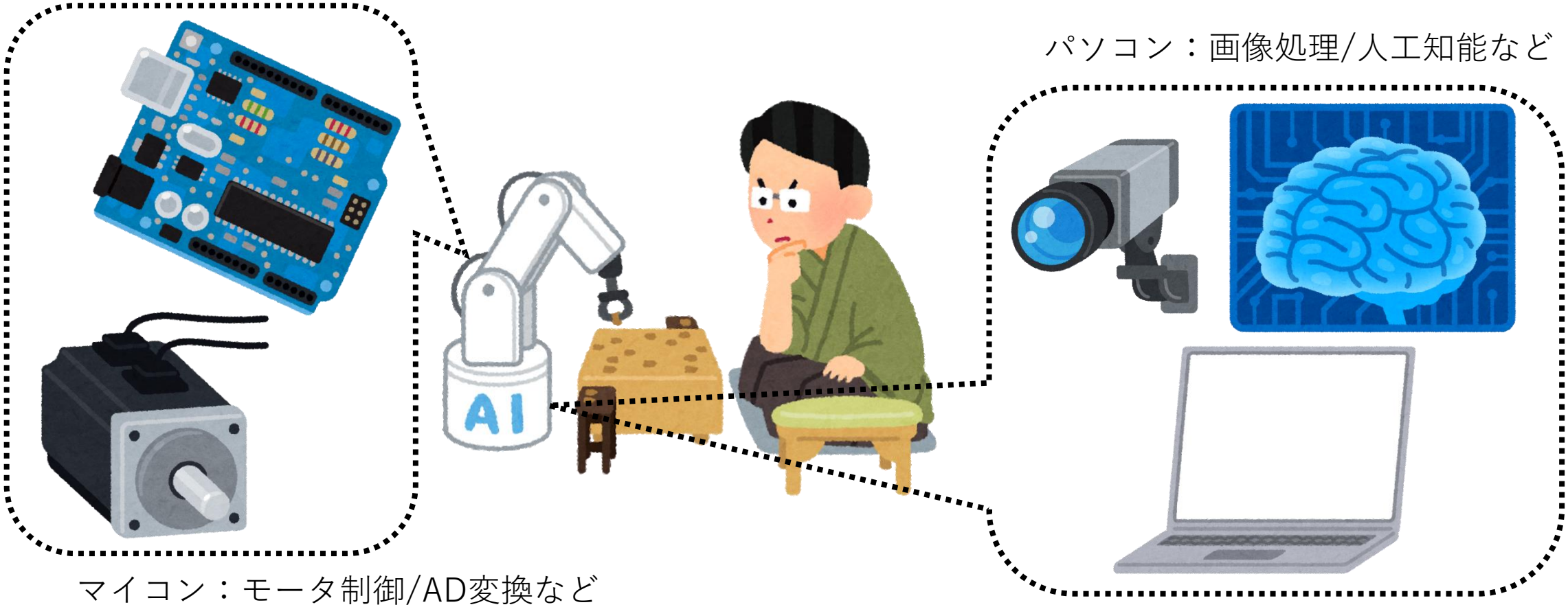
組み込みプログラミングの導入

【目次】組み込みプログラミングの導入

- 組み込みシステム、マイコンについて理解する
- 開発環境を整備する
  - Arduino IDE
- マイコンボード及び開発環境の動作確認
  - LEDピカピカ

【解説】組み込みシステムとは

- 特定の機能を実現するために家電製品や機械等に組み込まれるコンピュータシステムのこと（Wikipediaより）⇔パソコンは汎用的なシステム



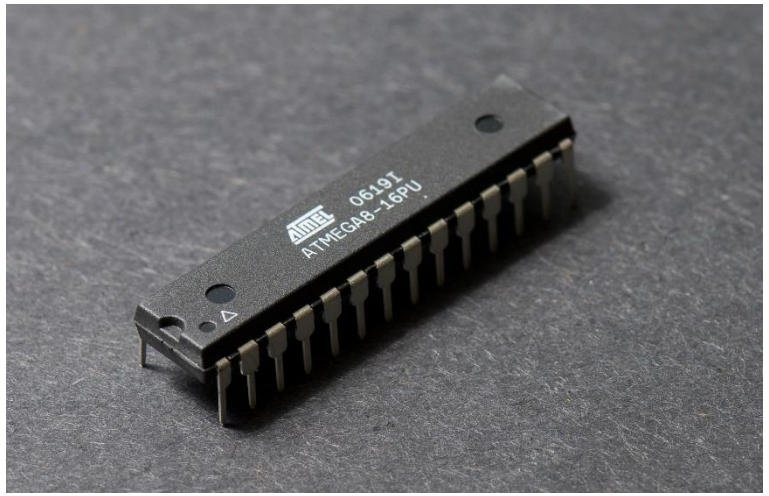
パソコン：画像処理/人工知能など

マイコン：モータ制御/AD変換など

【解説】マイコンとは

- マイクロコンピュータ
  - 1970～1980年代
  - 現在のパソコンに近い
    例：Commodore 64（右図）

- マイクロコントローラ
  - 2000年代～
  - 集積回路の一つ
  - CPU、メモリ、タイマー、入出力部
    例：Atmel AVR ATmega8（右図）



Wikipediaより

【解説】実験器具の確認

- パソコン（各自）
- Arduino学習キット基本版V1
  - 通称：ハコ

※機材は丁寧に扱うこと！！

機材の確認（Arduino学習キット基本版V1）

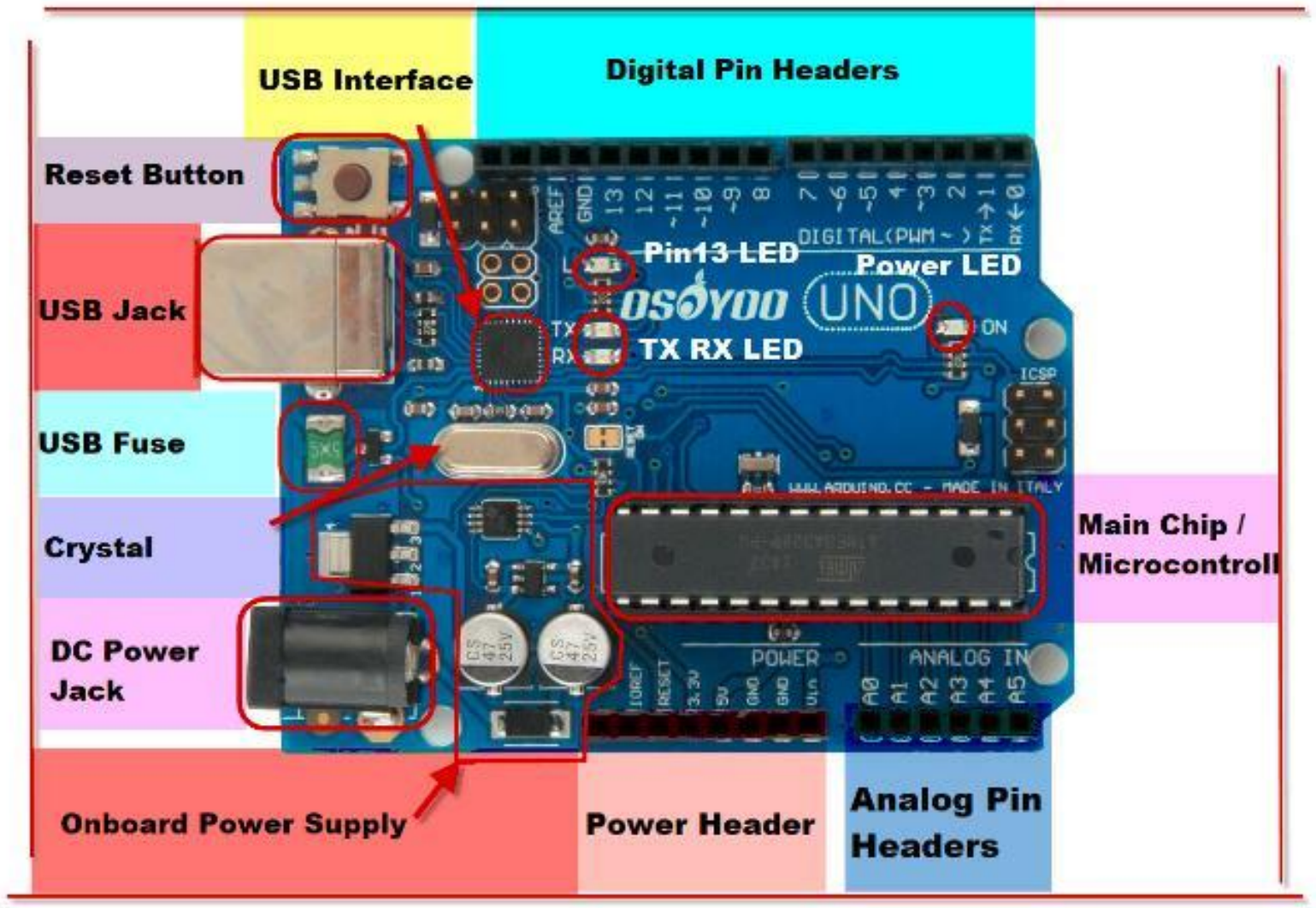
【解説】Arduinoとは

- Arduino（アルドゥイーノ）
  - 2005年にイタリアで誕生した、安価なマイコンボードの一種。
  - 技術的なハードルが低く、情報/機械系だけでなくデザイナ/クリエータ系など幅広く使われている

- 「プログラム」のことを「スケッチ（sketch）」と呼ぶ
  →講義では「プログラム」と呼ぶ

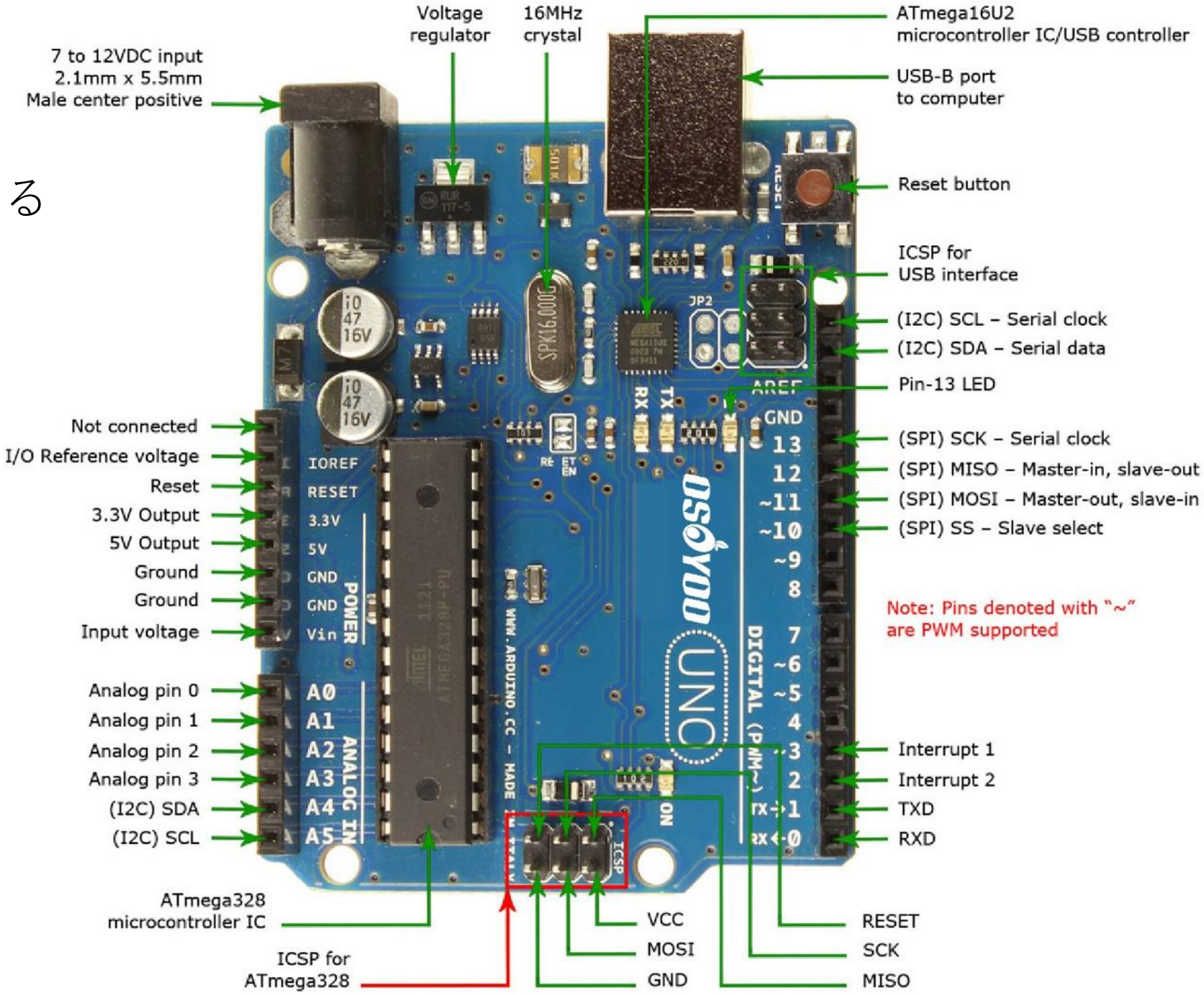
- 様々なシリーズ、互換機が存在
  - Arduino UNO、Arduino MEGA、Arduino Micro、Arduino Nanoなど
  - **Osoyoo**、ELEGOO Arduino用UNO、VKLSVAN UNO R3など
  →講義では「Arduino」と呼ぶ

【解説】Arduino互換機Osoyoo



【解説】ピン配置

- ピンには個別の機能がある
  - Reset
  - 3.3V Output
  - 5V Output
  - Ground
  - Analog
  - Digital
  - PWM「～」



【例題】開発環境の準備

- Arduino IDE
  - arduino-1.8.13-windows.zip
  - フリーソフト

※他の実験で「Arduino IDE」を
インストール済みの人は
ダウンロード不要！！

- ダウンロード先：

https://edison.elc.nias.ac.jp/eerc/experiment/fee/

※ネットワークに負荷がかかるので**必ずこのリンクからダウンロード**すること！
公式サイトにあるファイルと同じものです。
※Microsoft Storeから「Arduino」を検索してインストールもできますが、同じく
ネットワークに負荷がかかるので**今回はやらない！**

【例題】開発環境の準備

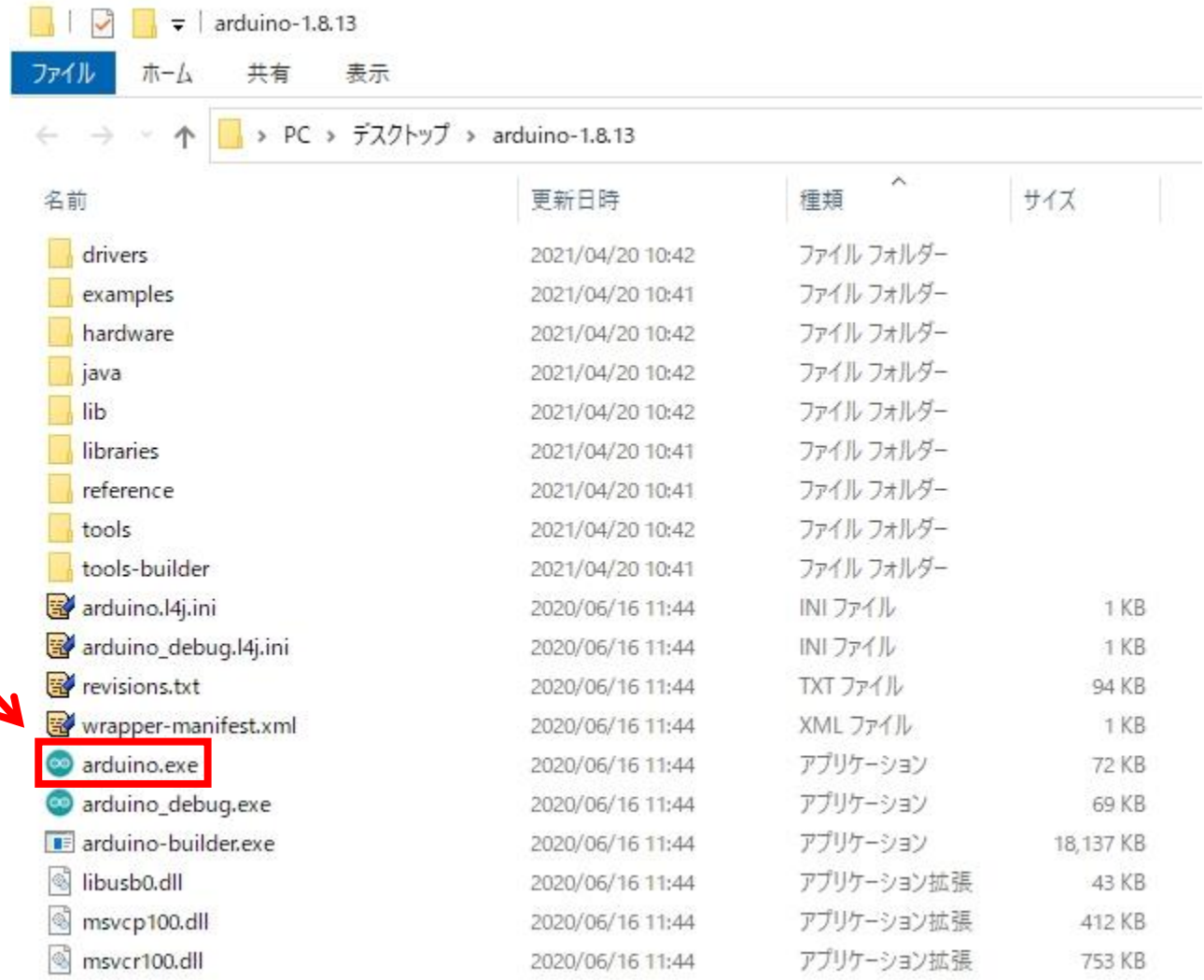
1. ダウンロードした「arduino-1.8.13-windows.zip」をデスクトップに移動

2. 「arduino-1.8.13-windows.zip」を展開（解凍）する

【例題】開発環境の準備

1. 「arduino-1.8.13」へ移動

2. 「arduino.exe」を起動

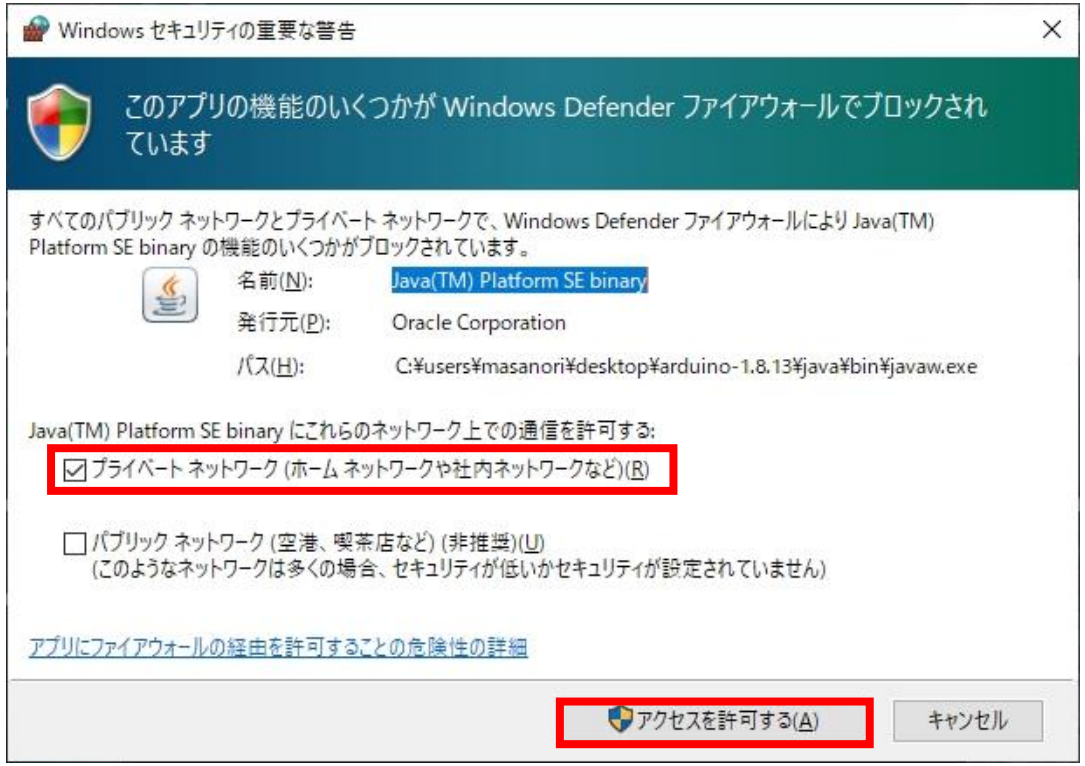


【例題】開発環境の準備

1. Windows セキュリティの重要な警告がでたら

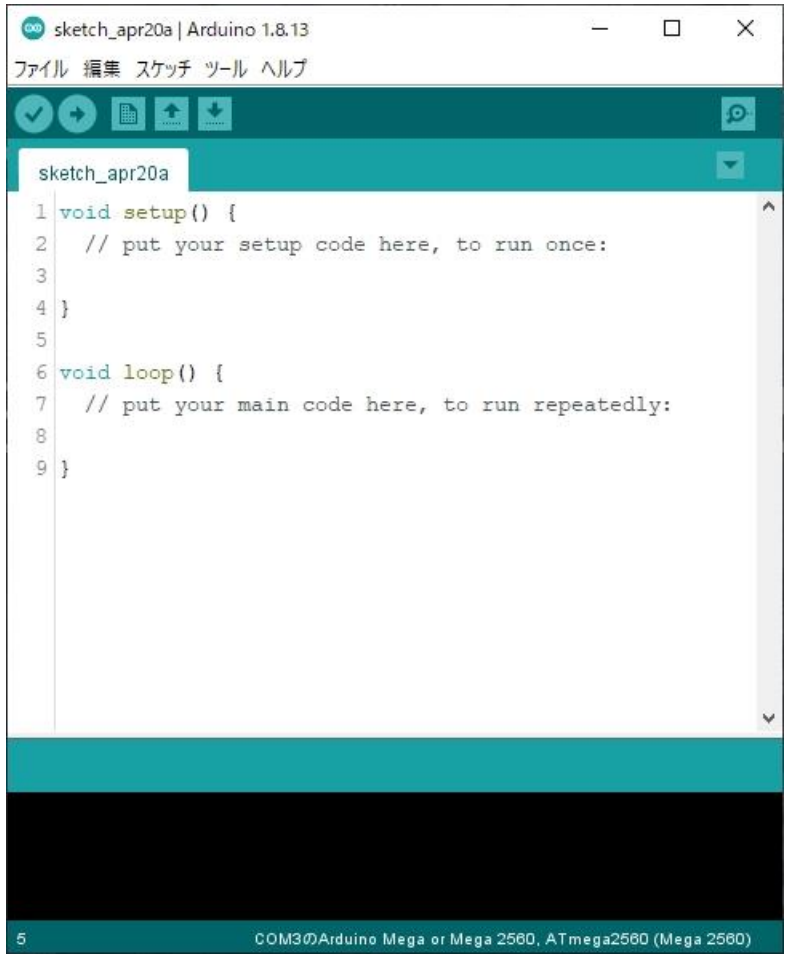
2. プライベートネットワーク（ホームネットワークや社内ネットワークなど）にチェック

3. アクセスを許可する



【例題】開発環境の準備

1. Arduino IDEが起動

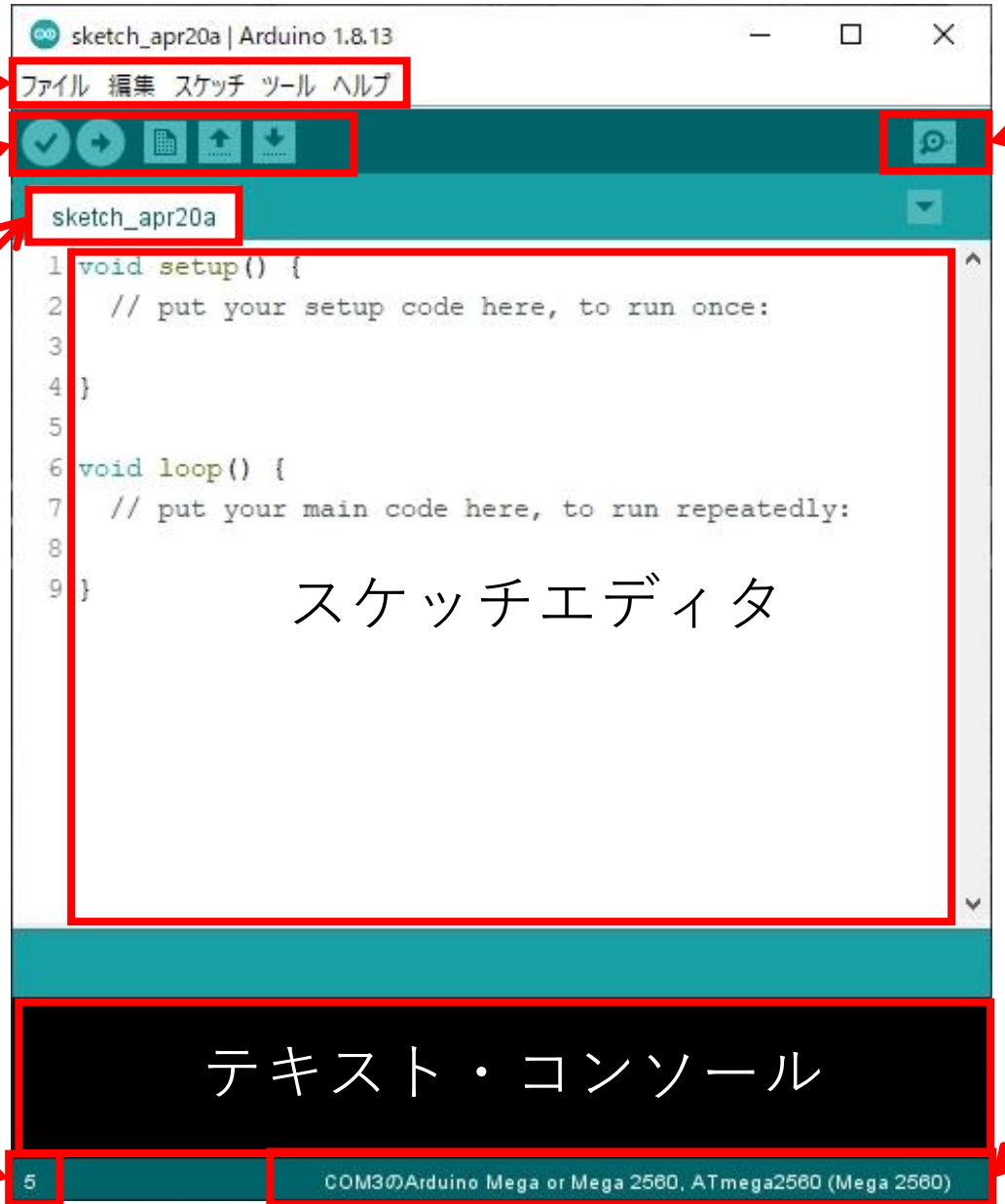


【例題】 Arduino IDE

メニューバー

ツールバー

スケッチ名



シリアルモニタ

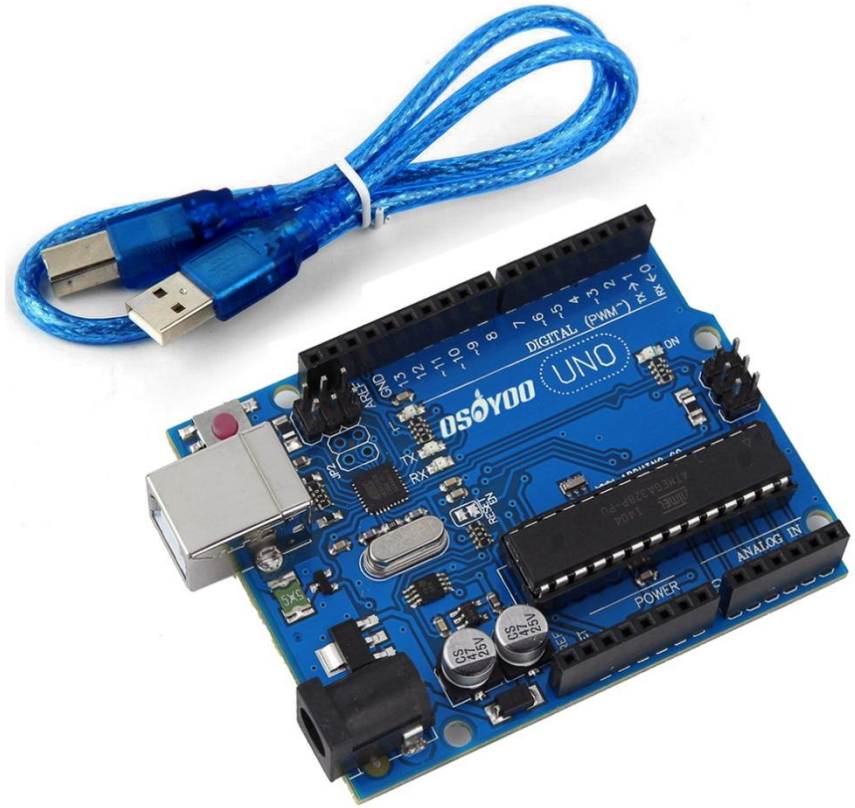
COMポート
と
接続Arduino

カーソル行

【例題】開発環境の準備

1. 「Arduino Uno」と「USBケーブル」を準備

2. コネクタを壊さないように**丁寧に接続**
   - 新品なので少しコネクタが固い（？）



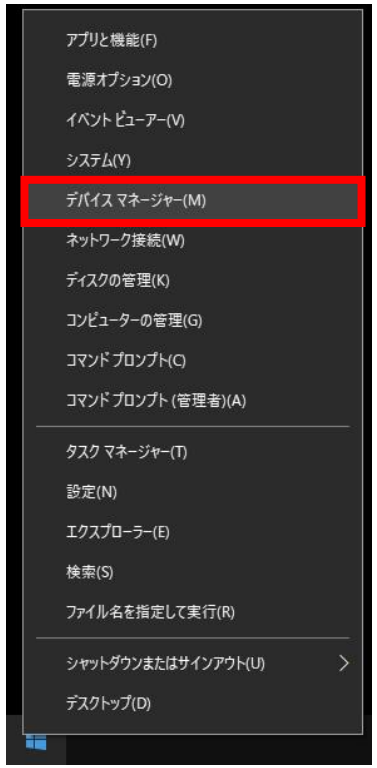
【例題】開発環境の準備

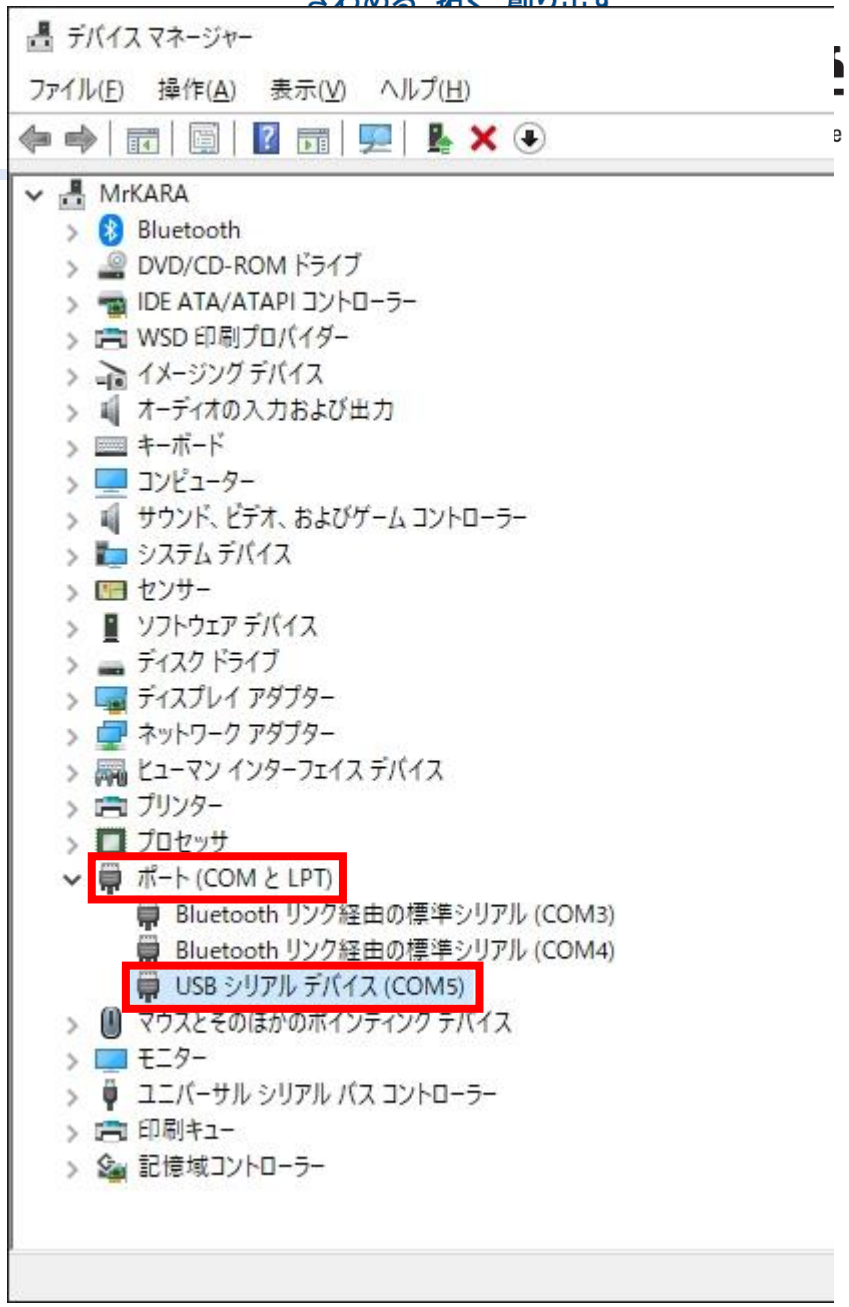
1. Windowsのタスクバーの左側にある「スタートメニュー」を右クリック

2. 「デバイスマネージャー」を選択

3. 「ポート（COMとLPT）」を展開

4. USBシリアルデバイス（COM?）
   - COM?の数字は環境に依存
   - この資料ではCOM5
   - **メモしておく**





【例題】開発環境の準備

使うAruduinoのシリーズを選択する

1. Arduino IDEの

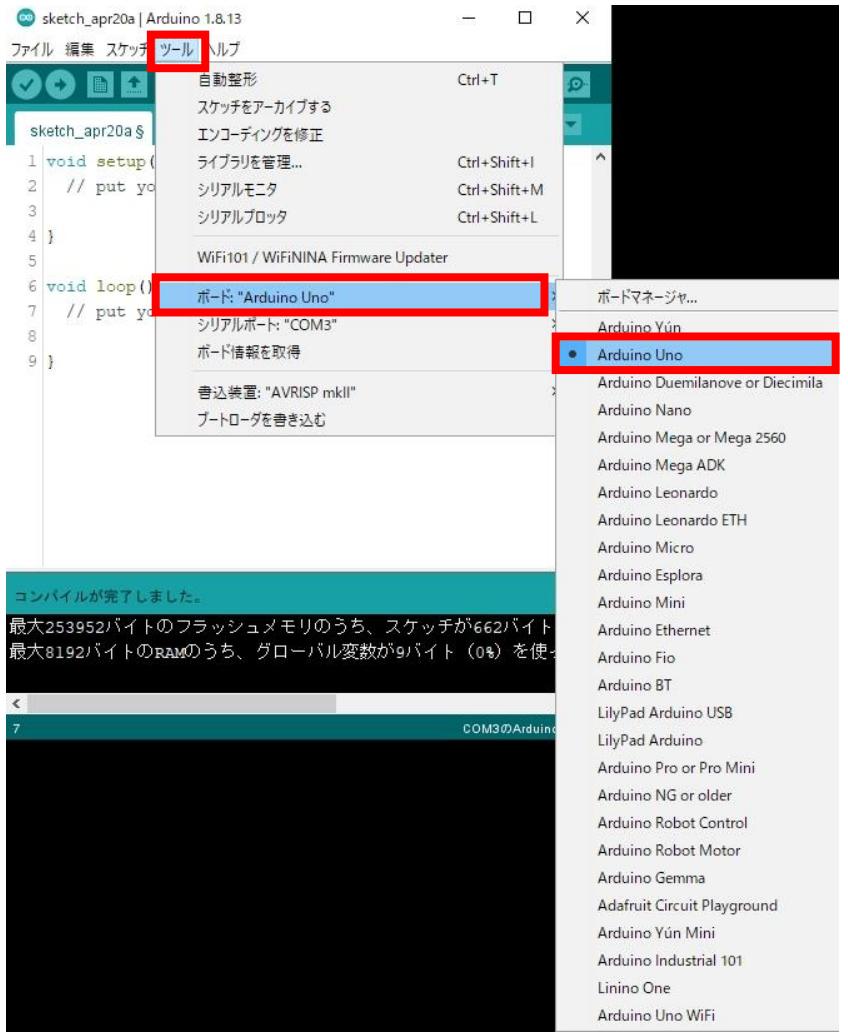
「ツール」

↓

「ボード」

↓

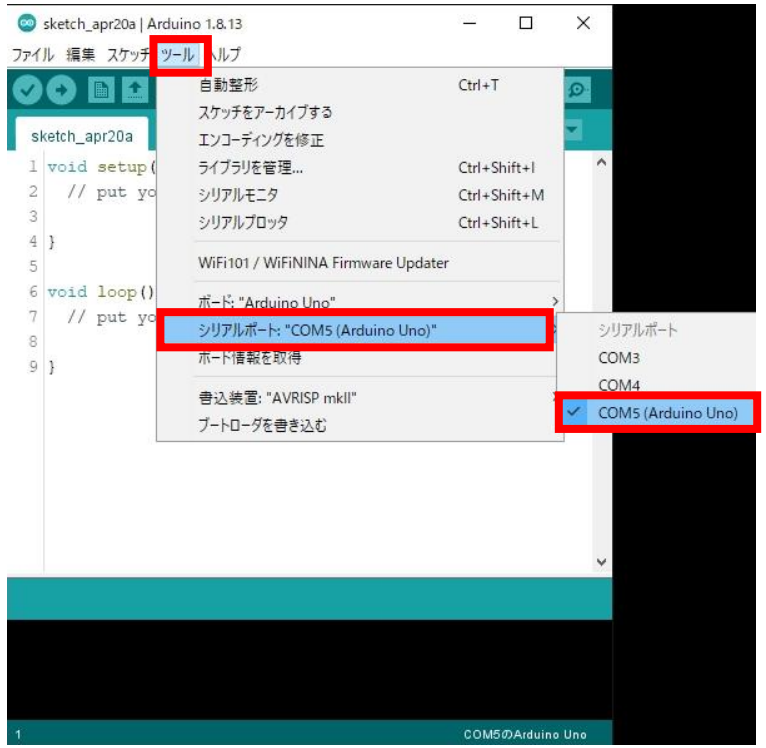
「Arduino UNO」



【例題】開発環境の準備

パソコンとArduinoをつなぐCOMポートを選択する

1. Arduino IDEの
   「ツール」
   ↓
   「シリアルポート」
   ↓
   「COM?(Arduino UNO)」

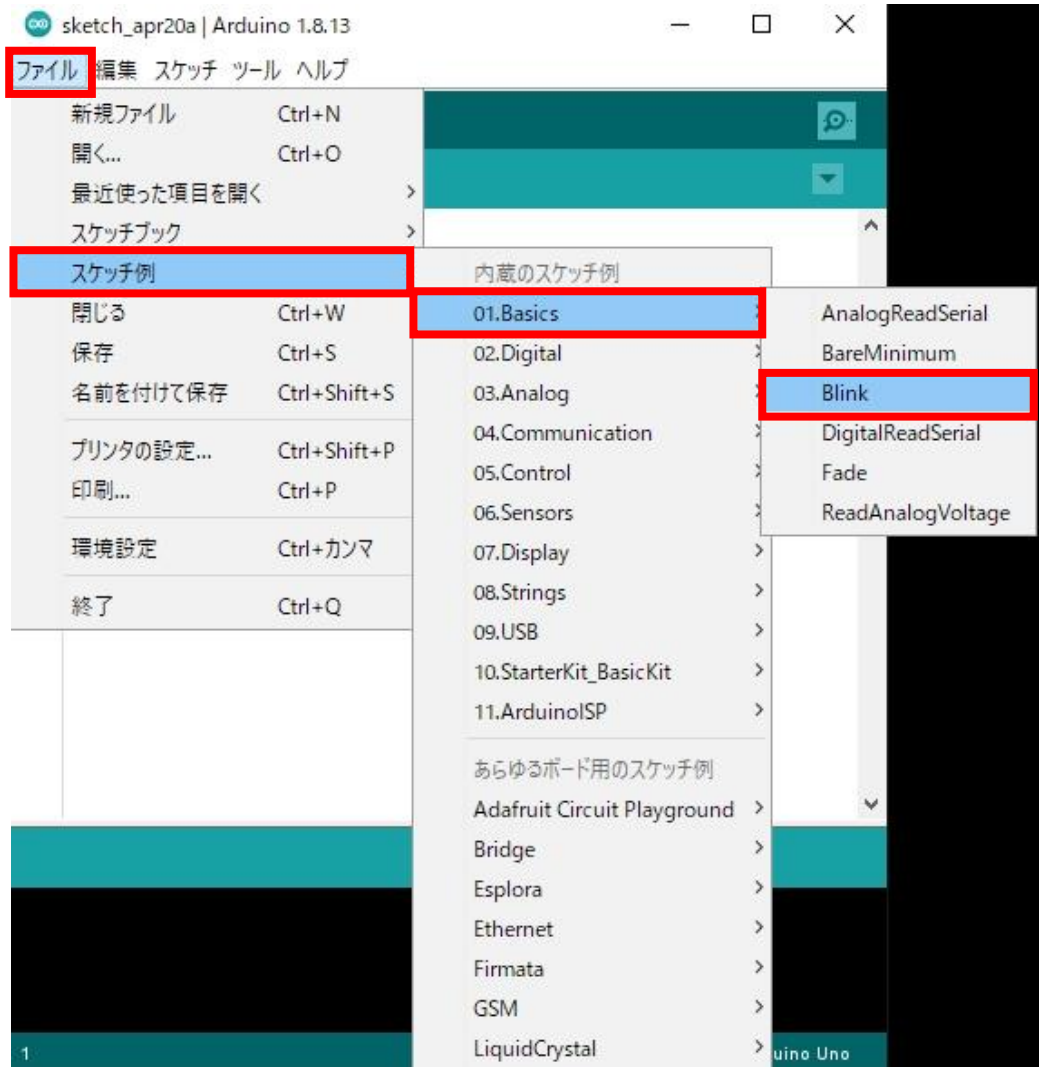


※COM？の数字は環境に依存
デバイスマネージャーで確認

【例題】動作確認

サンプルプログラムを選択する

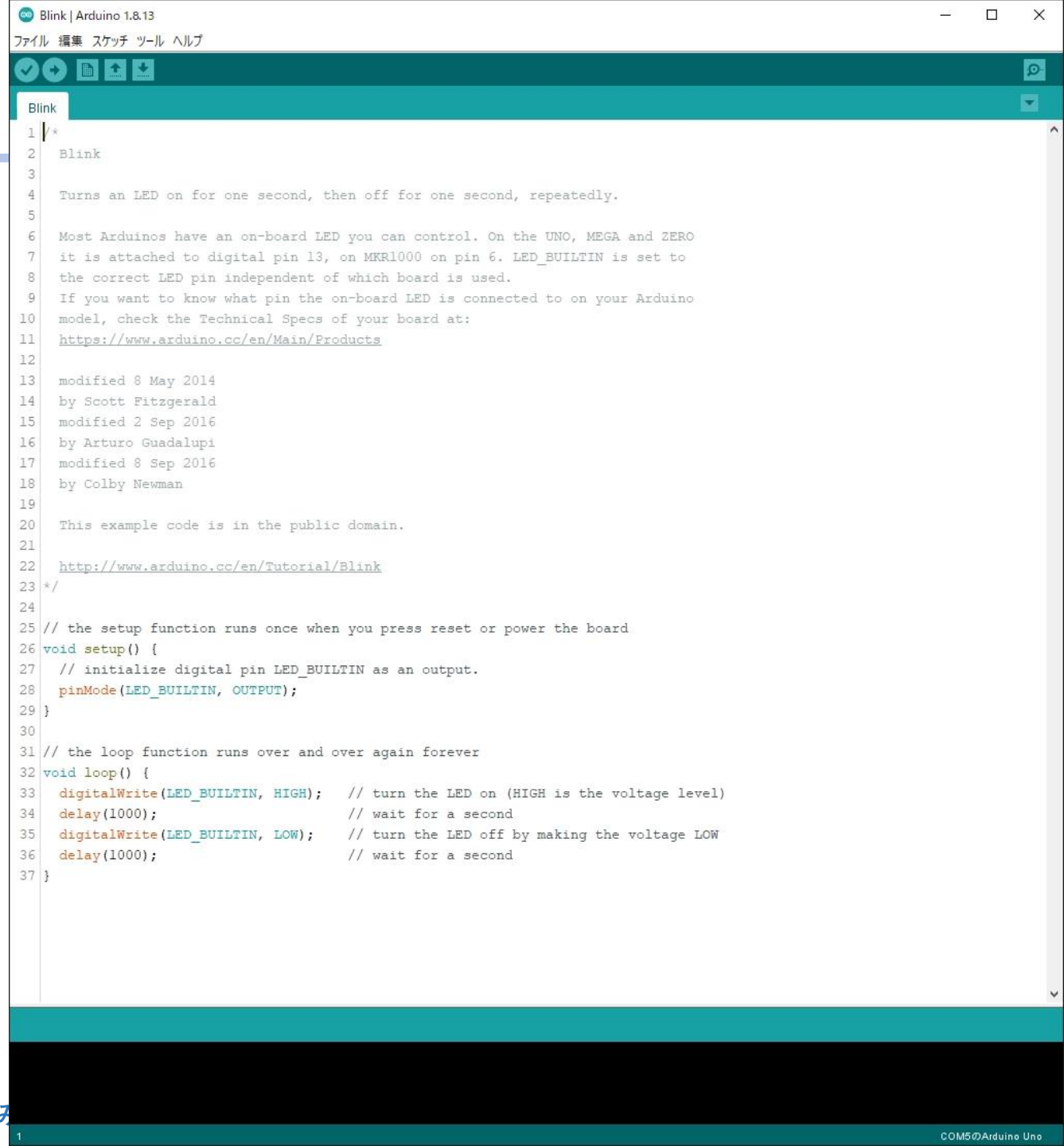
1. Arduino IDEの
   「ファイル」
   ↓
   「スケッチ例」
   ↓
   「01.Basics」
   ↓
   「Blink」



【例題】動作確認

1. 新しく「Blink」のスケッチが表示

- 1～23行目：コメント
- 25～29行目：初期設定
- 31～37行目：繰り返し
  - 33行目：LEDのピンをHigh
  - 34行目：1秒待機
  - 35行目：LEDのピンをLow
  - 36行目：1秒待機



【参考】Arduino日本語リファレンス

- delay(ms)
  - 一時停止する時間，単位はミリ秒

- pinMode(pin, mode)
  - pin：設定したいピンの番号
  - mode：INPUT，OUTPUT，INPUT_PULLUP
    - 設定したいピンを外部からの入力を受け付ける（INPUT）ようにする
    - 設定したいピンから外部へ出力（OUTPUT）するようにする
    - 設定したいピンを外部からの入力を受け付け、かつ内部でのプルアップ（INPUT_PULLUP）を有効にする

【参考】Arduino日本語リファレンス

- digitalWrite(pin, value)
  - pin：ピン番号
  - value：HIGH（5V）かLOW（0V）
  - 設定したピンからマイコンの外部へHIGHまたはLOWを出力する

- digitalRead(pin)
  - pin：読みたいピンの番号
  - 戻り値：HIGHまたはLOW
  - 設定したピンに接続された信号がHIGHかLOWか読みとる（HIGHまたはLOWをリターンする）

【参考】Arduino日本語リファレンス

- Serial.begin(speed)
  - speed：転送レート（例：9600なら9600bps）
- Serial.print(data, format)

文字列のときは””で囲う　例：Serial.print(”Hello”)

  - 例1：Serial.print(78)　　「78」が出力
  - 例2：Serial.print(78, BIN)　「1001110」が出力
- Serial.println(data, format)
  - 改行コード付きのSerial.print()
- Serial.write(val)
  - val：送信する値（1バイト）
  - str：文字列（複数バイト）

【参考】Arduino日本語リファレンス

- analogWrite(pin, value)
  - pin：設定したいピンの番号
  - value：デューティー比（0から255）
  - ArduinoUNO互換機は、デジタルピンのうち3、5、6、9、10、11が利用可能
    （目印として基板のピン番号の横に「～」マーク）
  - 設定したピンからマイコンの外部へアナログ電圧（PWM）を出力する

- analogRead(pin)
  - pin：設定したいピンの番号
  - ArduinoUNO互換機は、アナログピン（A0からA5）が利用可能
  - 0から1023までの整数値が戻り値（AD変換した値）
  - 設定したピンに接続されたアナログ電圧をAD変換した値をリターンする

【解説】一般的なマイコンと比較

一般的なマイコンの場合

```
int main(void)
{
        /// マイコンの初期設定

        /// プログラムの初期設定

        /// 無限ループ
        while(1)
        {
                /// 繰り返す作業を書く
        }

        return (0);
}
```

Arduinoの場合

```
void setup()
{
        /// 初期設定を書く
}


void loop()
{
        /// 繰り返す作業を書く
}
```

【解説】一般的なマイコンと比較

Arduinoが裏で実行

```
int main(void)
{
        /// マイコンの初期設定
        init();
        /// プログラムの初期設定
        setup();
        /// 無限ループ
        while(1)
        {
                ///繰り返す作業を書く
                loop();
        }
        return (0);
}
```

ユーザーが記述する内容

```
void setup()
{
        /// 初期設定を書く
}

void loop()
{
        ///繰り返す作業を書く
}
```

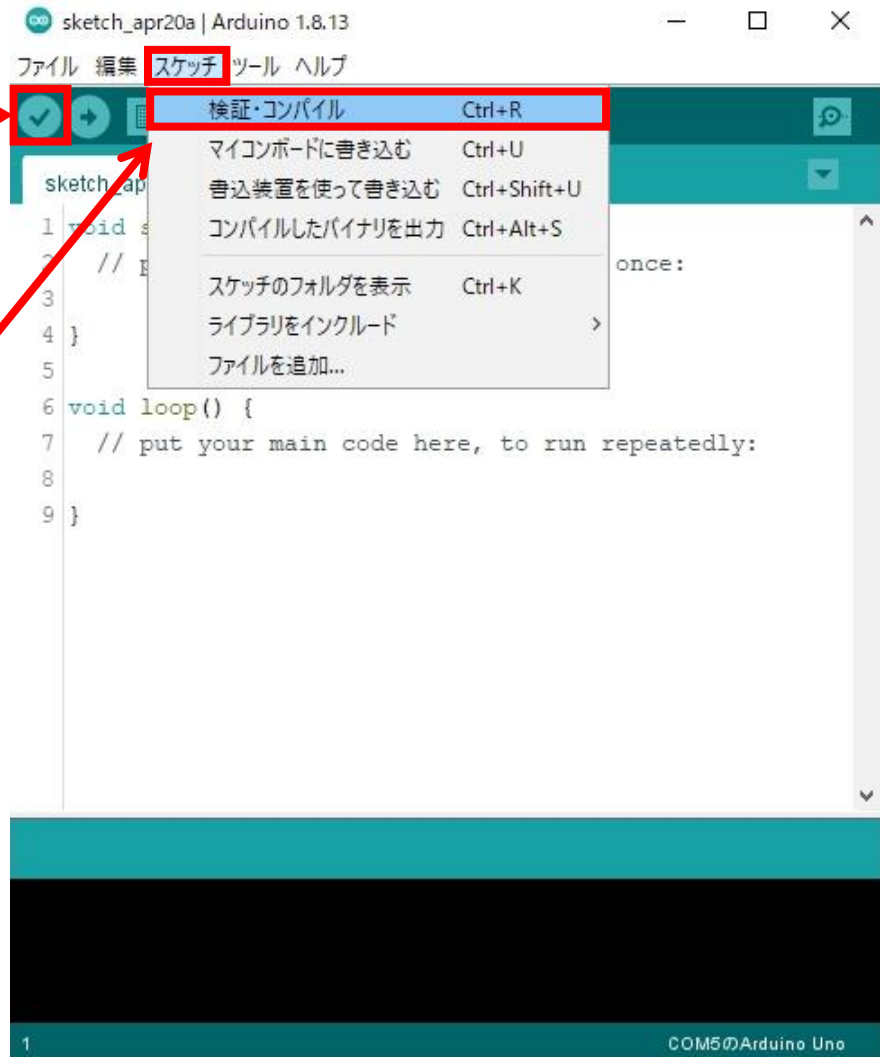
【例題】動作確認

プログラムをコンパイルする

- 「レ」ボタン

または

- 「Arduino IDEの
  「スケッチ」
  ↓
  「検証・コンパイル」



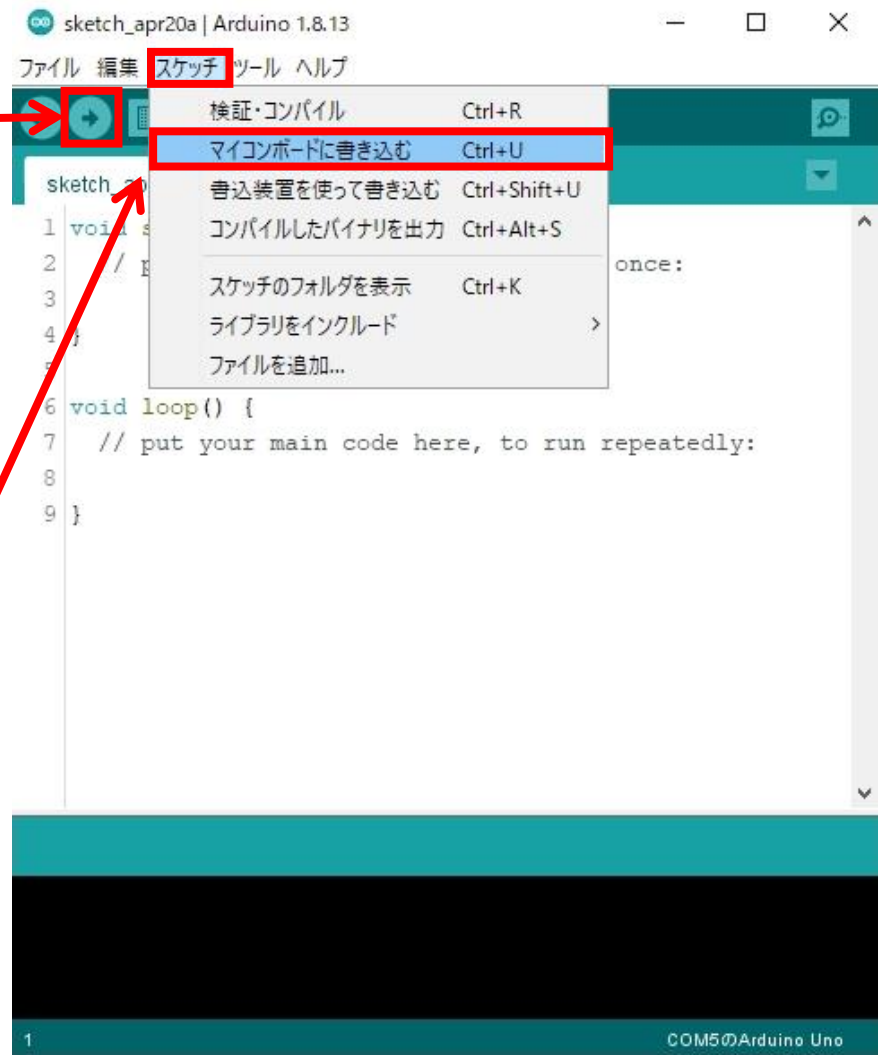
【例題】動作確認

プログラムを書き込む

- 「→」ボタン

または

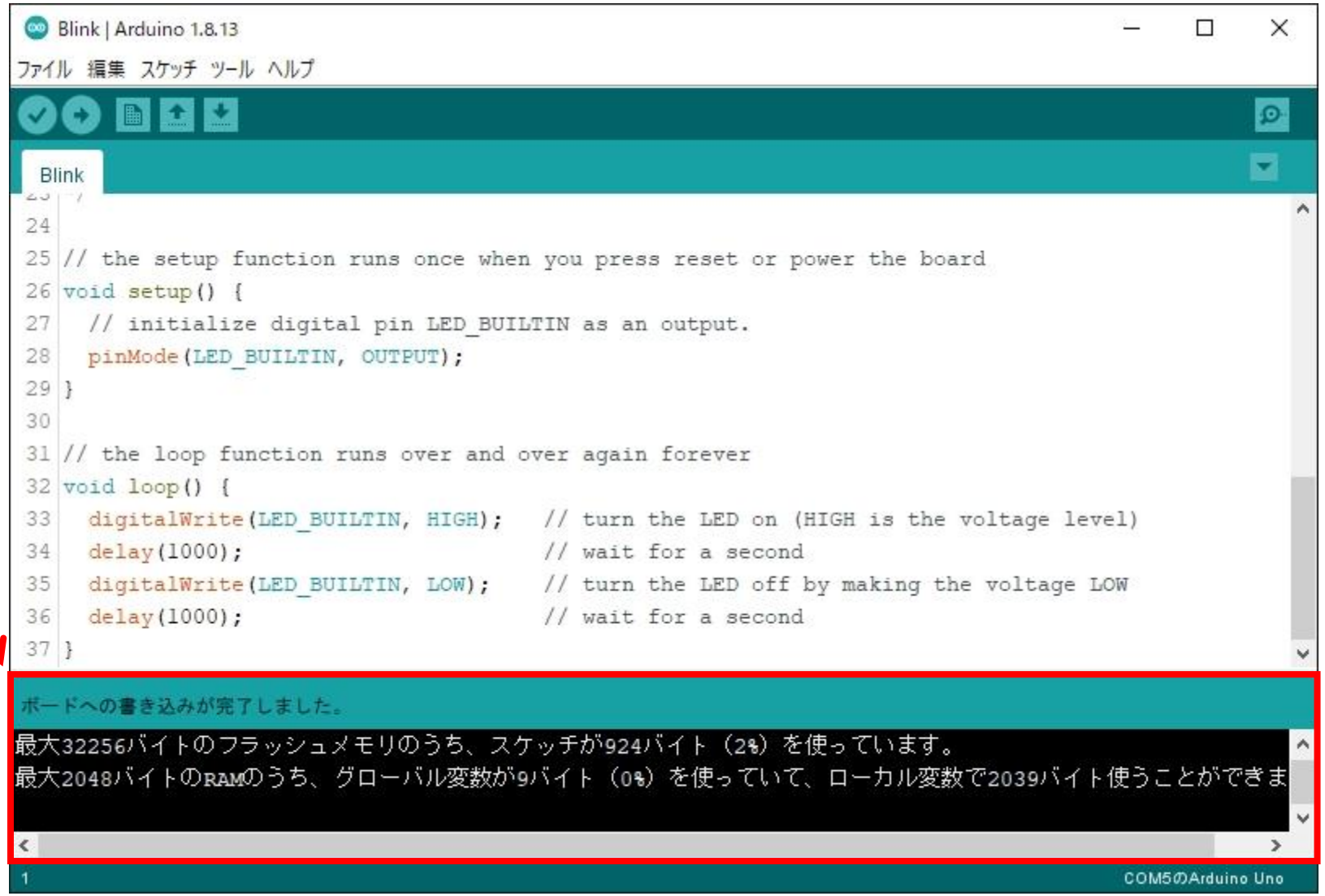
- 「Arduino IDEの
  「スケッチ」
  ↓
  「マイコンボードに書き込む」



【例題】動作確認

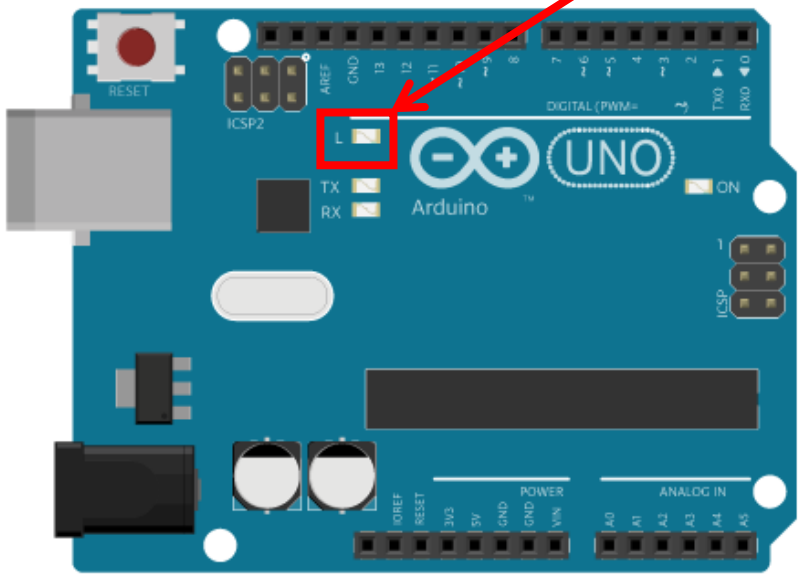
書き込み完了（成功）

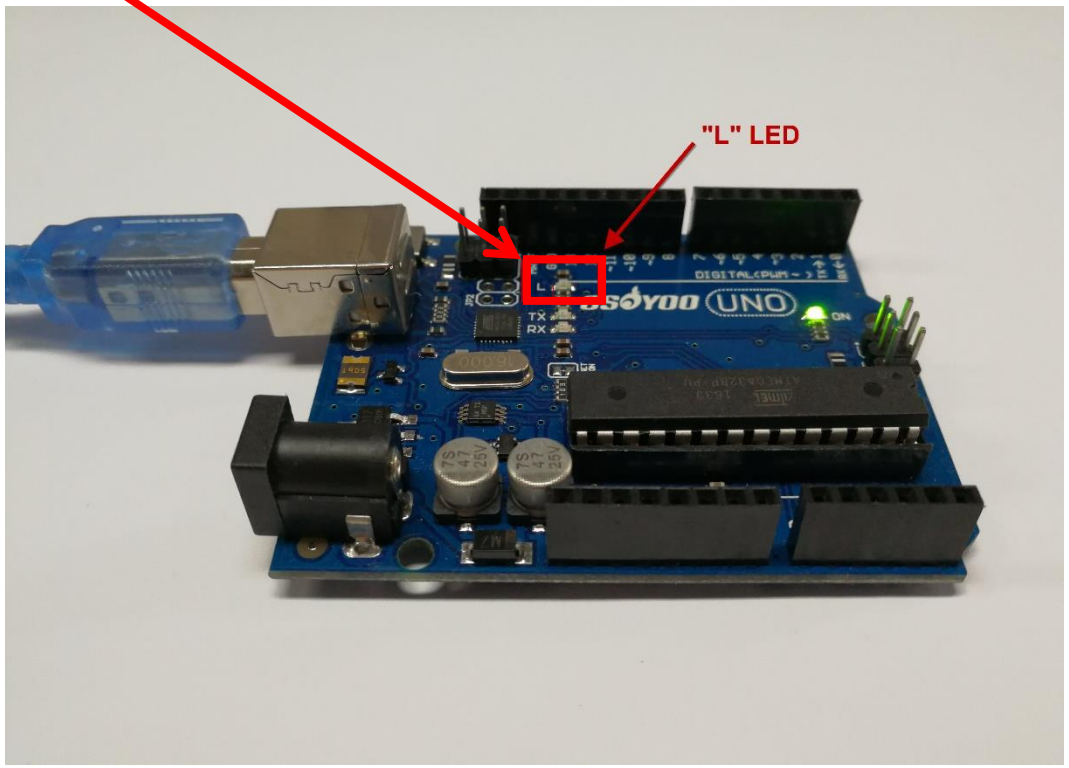
◘ 下段のメッセージで確認



【例題】動作確認

- ボードに搭載されたLEDが1秒間隔で点滅

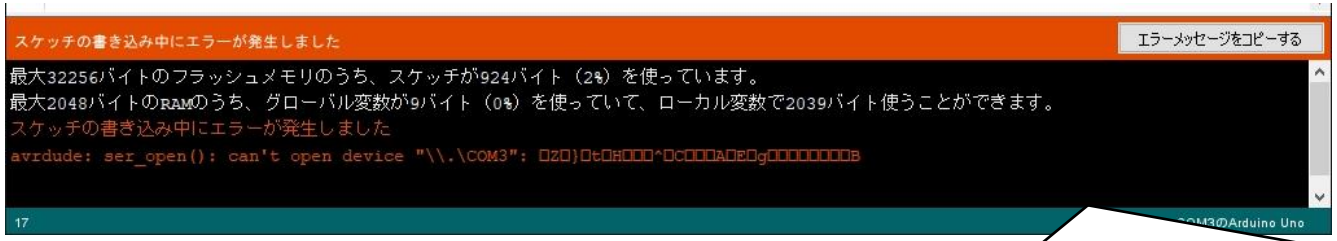




【例題】『動作不良』の動作確認

□ 「失敗あるある」を体験しよう　→　エラーから逃げないこと

1. COMポートの設定ミス
   - □ 「ツール」→「シリアルポート」で「わざと」正しくないポートにする



2. 接続していない
   - □ USBケーブルを外す

エラーを読んでおく
次に同じメッセージがでたら対処できるように！

【例題】C言語の復習（補足？蛇足？）

一緒に簡単なC言語のプログラムを書いてみよう（注：佐藤流）

1. メモ帳を開く、名前を付けて保存（名前は何でもよい）
2. 以下のプログラムを、佐藤流の書き方で一緒に書く

```
#include <stdio.h>
int main(void)
{
	int i = 0;
	printf("i = %d¥n", i);
	return(0);
}
```

【演習（基本）】

保存場所は自由

1. 新しくファイルを作ってみる
   - 「ファイル」→「新規ファイル」→「名前をつけて保存」
   - ファイル名は「LEDPikaPika」
2. 次のページのプログラムを記述
3. 点滅時間を変更して動作確認
   - 33行目：LEDのピンをHigh
   - **34行目：1秒待機**
   - 35行目：LEDのピンをLow
   - **36行目：1秒待機**

   - 「1000」を「100」へ変更
   - 「1000」を「2000」へ変更

【演習（基本）】LEDPikaPika


```
int LEDPIN = 13;
```

なぜ、わざわざ変数をつくるのか？

```
void setup()
{
        pinMode(LEDPIN, OUTPUT);
}

void loop()
{
        digitalWrite(LEDPIN, HIGH);
        delay(1000);
        digitalWrite(LEDPIN, LOW);
        delay(1000);
}
```

プログラミングのコツ

✓ 大枠から埋めていく、詳細は佐藤流で
（×）上から順に、左から順に書く
（〇）関数名を書く→カッコを閉じる
→中身を書く

✓ 書いたプログラムを変更する
（×）書いた文を消す
（〇）書いた文をコメントアウトする

【演習（基本）】

- 時間を「10」に変更
- 点滅が見える？？
- **USBケーブルに気を付けながら**、Arduinoを振ってみよう
  - LEDの点滅がどう見える？？

ーーーーーー　とか　- - - - -

※完成したらTAに確認依頼（課題1-1）

【演習（応用）】

◘ モールス信号のSOSを表現してみよう

・・・　ーーー　・・・

（短短短　長長長　短短短）

を繰り返す

（Arduinoを振るのではなく、Arduinoを机に置いてLEDが短く又は長く点灯する）

※完成したらTAに確認依頼（課題1-2）

きわめる。拓く。創り出す。
NAS 長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.1-2

シリアル通信

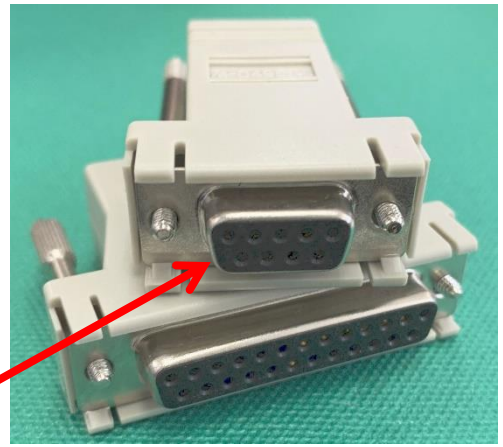
【目次】シリアル通信

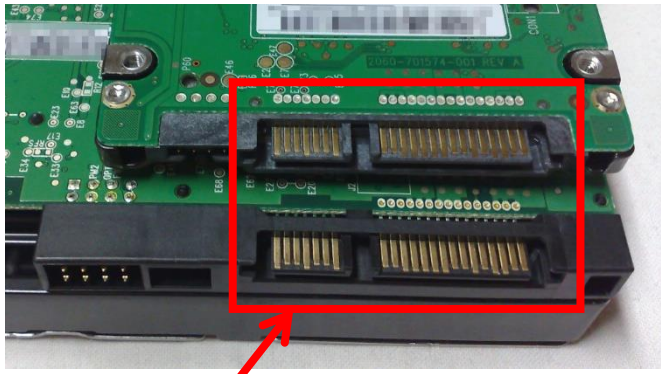
- シリアル通信を理解する
- シリアル通信を実装する
  - Hello Arduino
- C言語の復習（挑戦？）数値積分
  - 基本課題：長方形近似
  - 応用課題：台形近似

3週間の講義期間中、与えられた課題が終わったら
挑戦してみよう

【解説】シリアル通信とは

- シリアル通信（Serial Communication）
  ⇔パラレル通信





（例）

- RS232：Recommended Standard 232
  - パソコン同士、周辺機器などの接続
- USB：Universal Serial Bus
- Serial ATA（SATA）：SSDや光学ドライブを接続するインターフェース
- SPI、I^2C：シリアルバスの一種

Wikipediaより

【例題】Hello Arduino

1. 新しいスケッチを起動
2. 「HelloArduino」という名前で保存（**HelloとArduinoの間はつめる**）
3. 次のプログラムを書いて上書き保存
4. ダウンロード

（○）HelloArduino.ino

（×）Hello　Arduino.ino

【例題】Hello Arduino

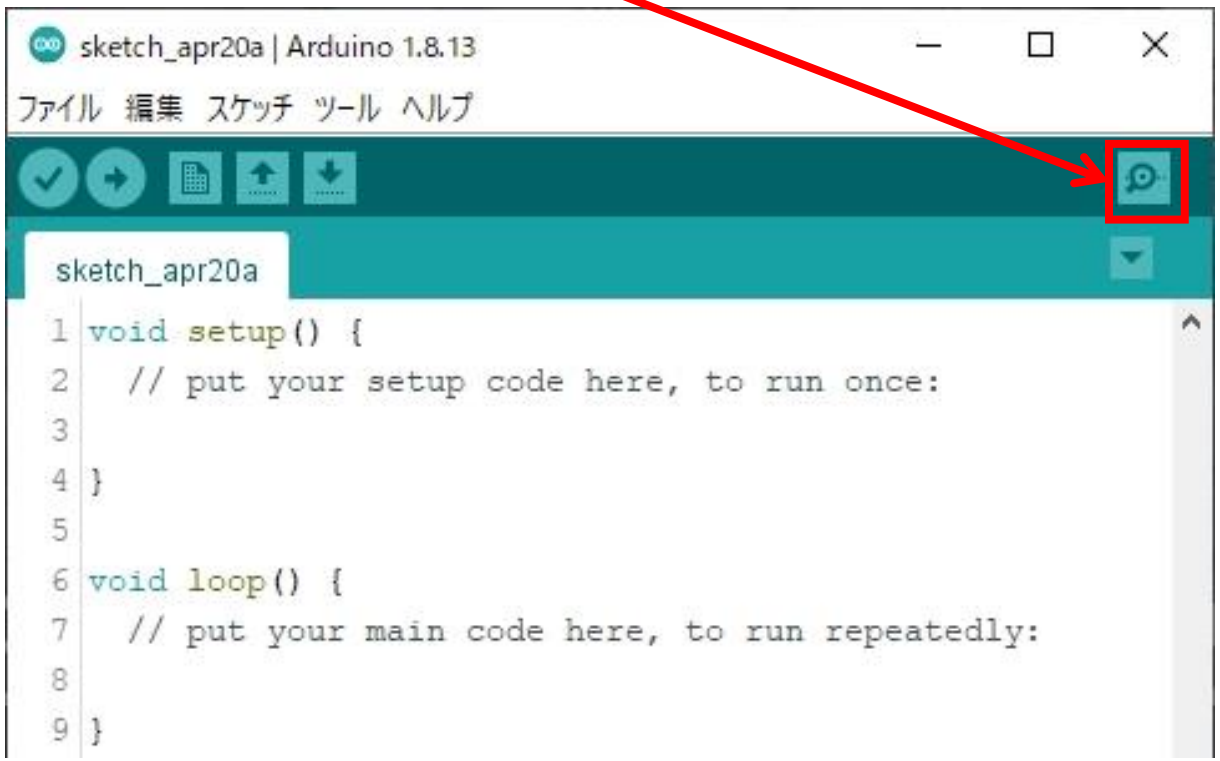

```
void setup()
{
    Serial.begin(9600);
}

void loop()
{
    Serial.println("Hello Arduino");
    delay(1000);
}
```

シリアル通信の初期設定、通信速度

文字列を出力するために“”で囲う

【例題】Hello Arduino

5. シリアルモニタを起動



【演習（基本）】

- 例題『Hello Arduino』に倣って、自分の学籍番号と氏名をシリアルモニタ上に表示

※表示出来たらTAに確認依頼（課題1-3）

【演習（基本）】

□ 関数$f(x) = x^2 + 5$について、$x = 1.2$であるときの値を表示

```
float x = 1.2;
float y = 0.0;

void setup() {
  Serial.begin(9600);
}

void loop() {

  delay(1000);
}
```

※あくまで参考

※計算結果を表示出来たらTAに確認依頼（課題1-4）

【演習（基本）】

- 1から10までの自然数の和を求める
  - 条件：forループを使う
  - 計算結果のSを表示する（55になるはず）

$$S = \sum_{i=1}^{10} x_i = x_1 + x_2 + \cdots + x_{10}$$
$$x_i = 1,2,3,\cdots,10$$

※計算結果を表示出来たらTAに確認依頼（課題1-５）

【解説】C言語のforループ


```
for(①初期化; ②条件式; ④変化式)
{
        ③実行する処理;
}
```

（例）

```
int i = 0;
for( i = 0; i < 5; i++)
{
        printf("i = %d¥n", i);
}
```

実行の順序

①初期化
　変数を初期化

②条件式
　条件式を判定し、満たす場合は③へ、
　満たさない場合は {} の外へ

③実行する処理
　処理を実行

④変化式
　変数のカウントアップなど

②へ戻る

K7：組み込み演習（挑戦）

（待ち時間で）プログラミング演習

【解説】プログラミング演習

- 各週の課題が終わって時間があるときに挑戦してみよう

課題：数値積分

- 微分積分はとても重要で、講義等で学んでいる例として
  - 電気：「微分要素（コイル）」「積分要素（コンデンサ）」
  - 機械：「位置」⇔「速度」⇔「加速度」

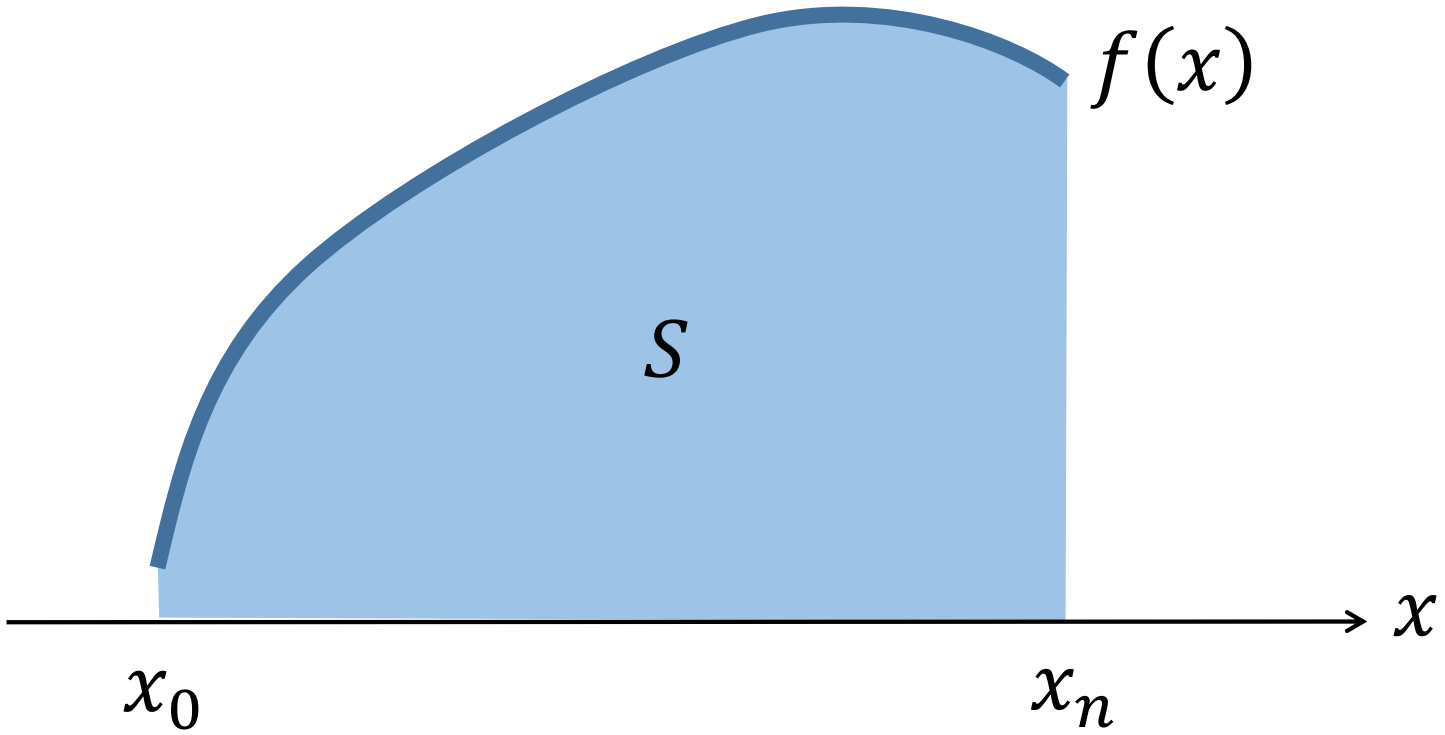
  があります。物理シミュレーション（ゲーム）などでも使います。

- 皆さんは「微分積分学I」「微分積分学II」などを学んでいますが、基本的に「連続系」で考えています。コンピュータで数値積分をするためには、「連続系」を「離散系」で考えます。簡単な例として、積分によって面積を出す場合に小さな短冊（又は台形）に分割して総和を求めます。

【解説】数値積分

◘ 連続時間における積分

$$S = \int_{x_0}^{x_n} f(x)dx$$

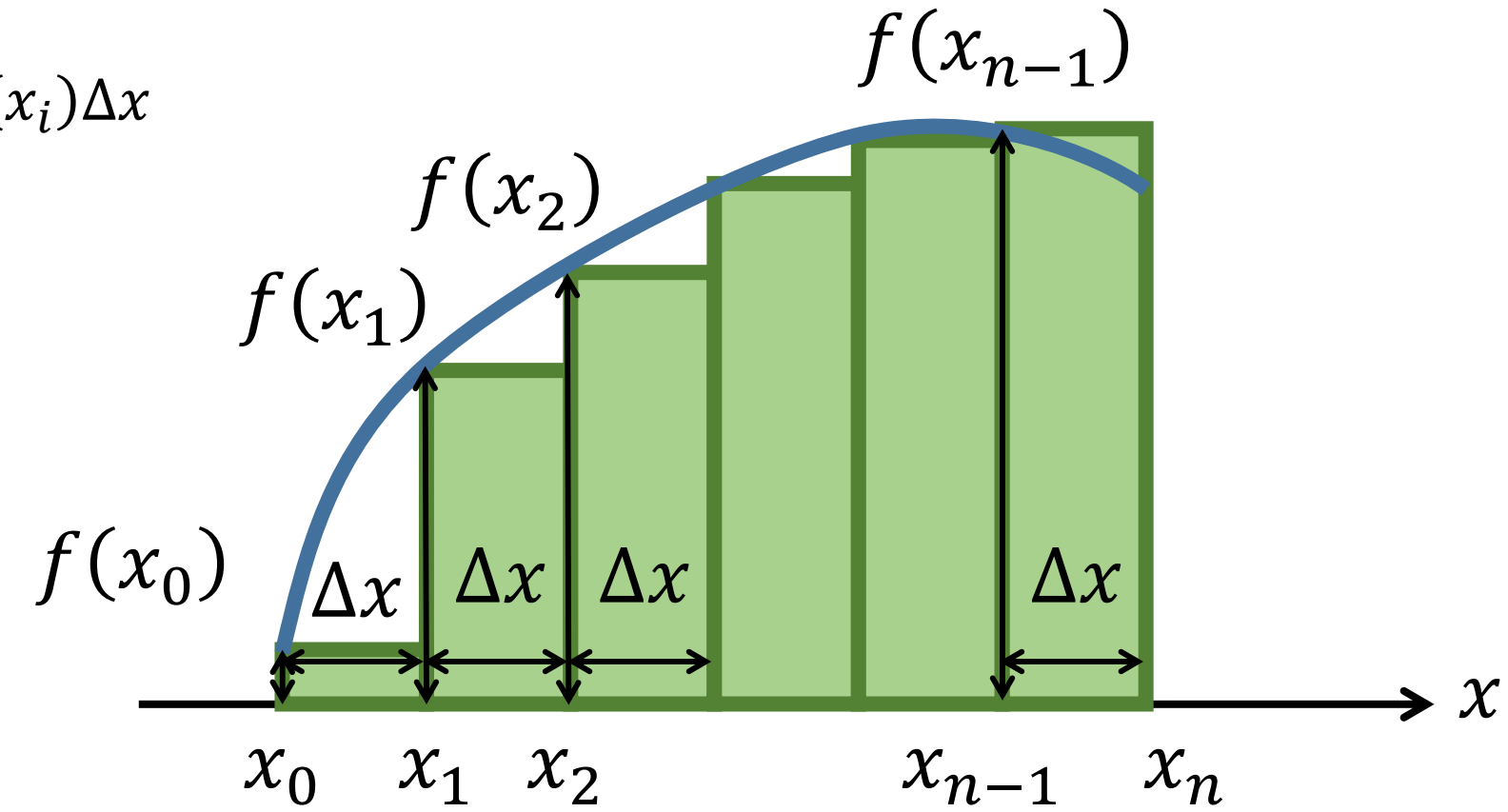


【解説】数値積分（長方形近似）

$n-1$に注意

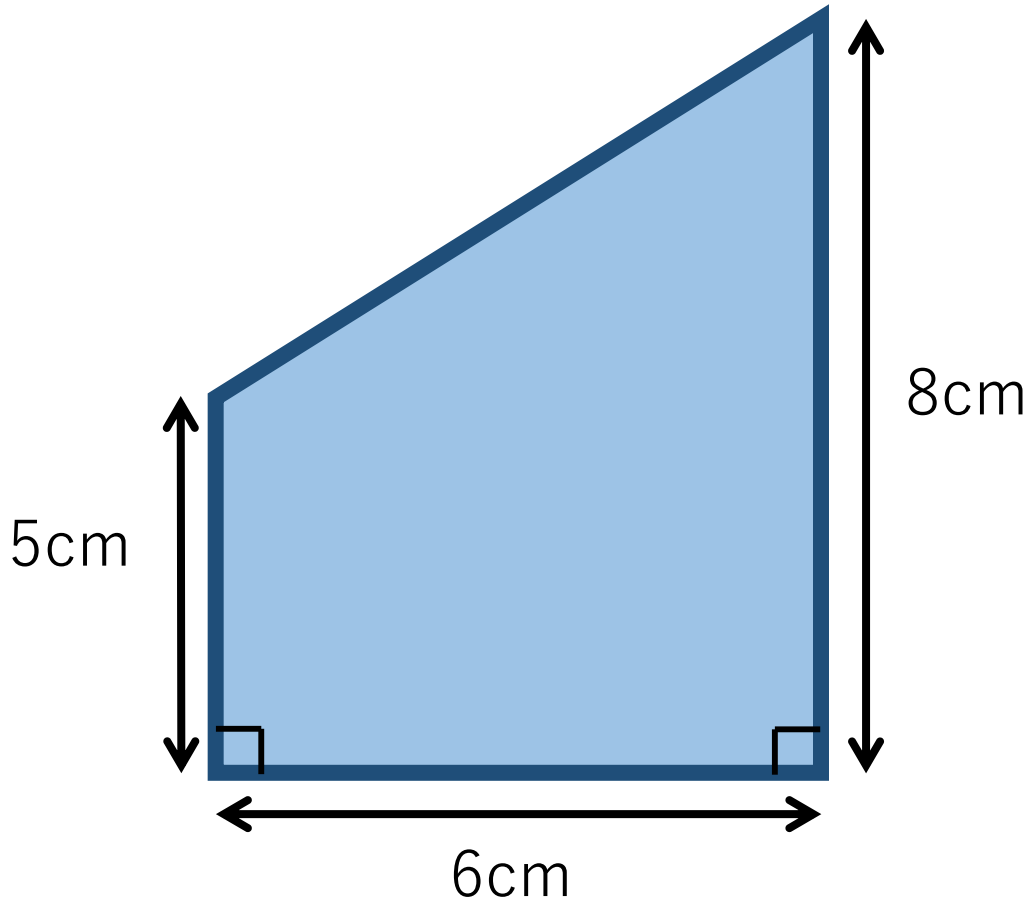
$$S = f(x_0) \times \Delta x + f(x_1) \times \Delta x + \cdots + f(x_{n-1}) \times \Delta x$$

$$= \sum_{i=0}^{n-1} f(x_i)\Delta x$$



【解説】数値積分

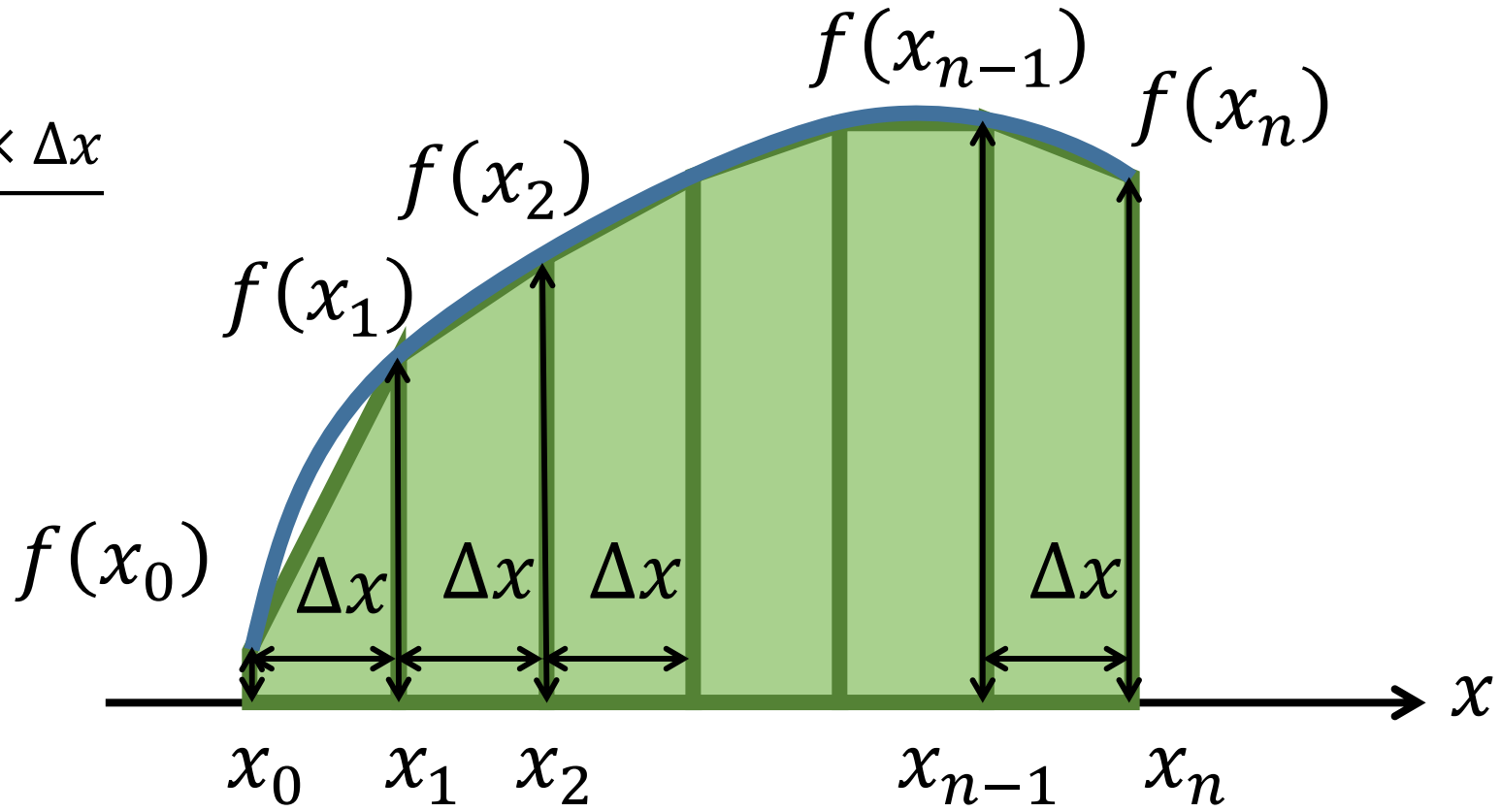
- 次の図形の面積は？



【解説】数値積分（台形近似）

$$S = \frac{\{f(x_0) + f(x_1)\} \times \Delta x}{2} + \frac{\{f(x_1) + f(x_2)\} \times \Delta x}{2} + \cdots + \frac{\{f(x_{n-1}) + f(x_n)\} \times \Delta x}{2}$$

$$= \sum_{i=0}^{n-1} \frac{\{f(x_i) + f(x_{i+1})\} \times \Delta x}{2}$$



【演習（応用）】数値積分（長方形近似）

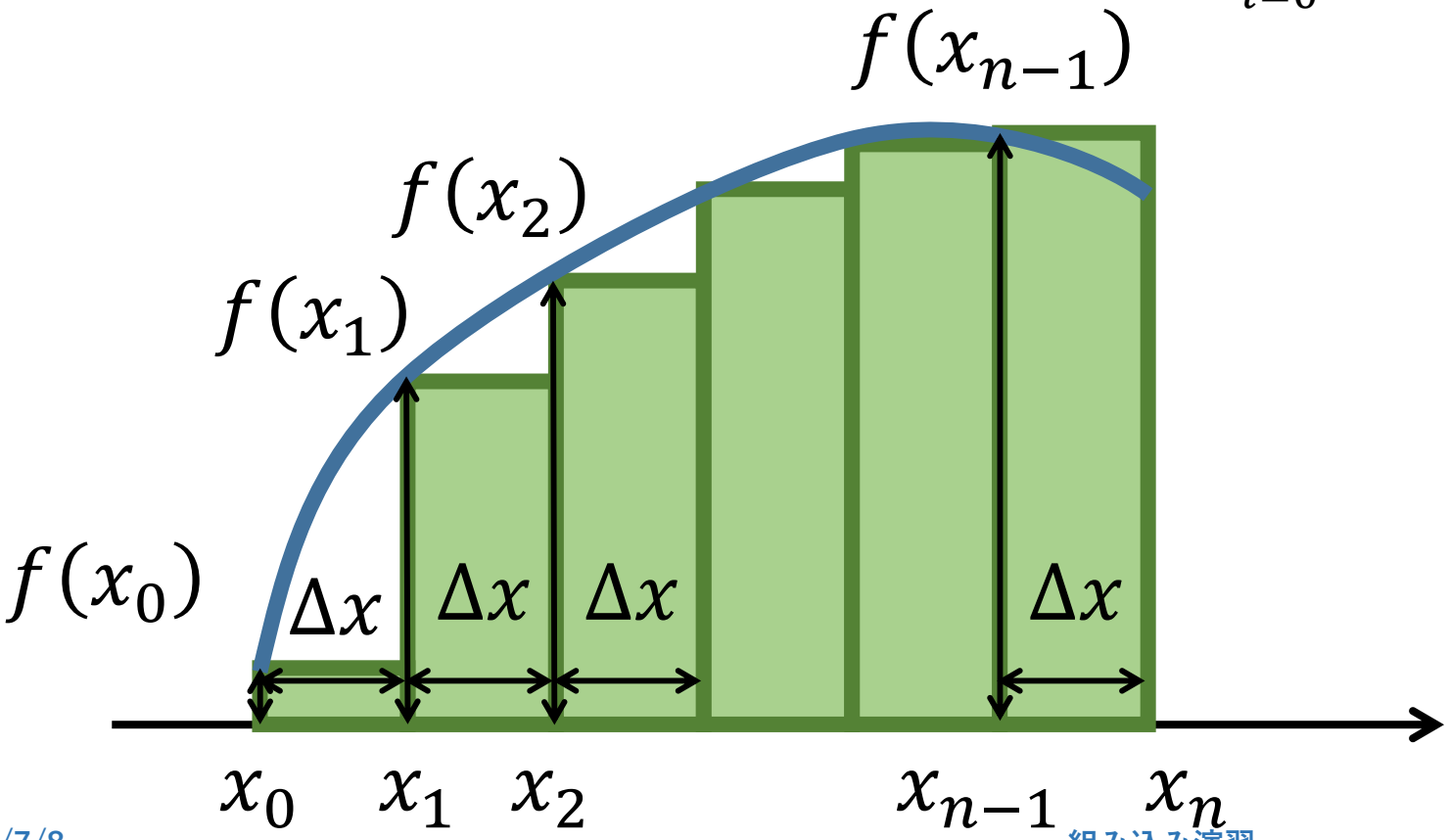
- 関数$f(x) = x^2 + 5$について、積分区間[1,5]で数値積分（長方形近似）

※計算結果を表示出来たらTAに確認依頼（挑戦1）

【演習（応用）】数値積分（長方形近似）

□ 関数$f(x) = x^2 + 5$について、積分区間[1,5]で数値積分（長方形近似）

$$S = \int_1^5 f(x)dx \fallingdotseq \sum_{i=0}^{n-1} f(x_i)\,\Delta x$$



$$x_1 = x_0 + 1 \times \Delta x$$
$$x_2 = x_0 + 2 \times \Delta x$$
$$\vdots$$
$$x_i = x_0 + i \times \Delta x$$
$$\vdots$$
$$x_{n-1} = x_0 + (n-1) \times \Delta x$$

【演習（応用）】数値積分（長方形近似）

◘ プログラムの流れ

1）積分区間の1と5は固定でよい

（例）`float x0 = 1.0, xn = 5.0;`

2）積分区間を分割する『分割数』numを変数として導入する

（例）`int num = 10;`

※計算結果を確認したら、num=20、10000など試す事

3）微小な幅を変数で用意し、積分区間と分割数から求める

（例）`float dx = (xn - x0)/num;`

4）面積を求める下記の式（シグマ）をforループで実装する

$$S = \sum_{i=0}^{num-1} f(x_i) \times \Delta x = \sum_{i=0}^{num-1} f(x_0 + i \times \Delta x\ ) \times \Delta x$$

【演習（応用）】数値積分（台形近似）

- 関数$f(x) = x^2 + 5$について、積分区間[1,5]で数値積分（台形近似）

※計算結果を表示出来たらTAに確認依頼（挑戦2）

きわめる。拓く。創り出す。
長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.2-1

デジタルIO（Output）

【目次】デジタルIO（Output）

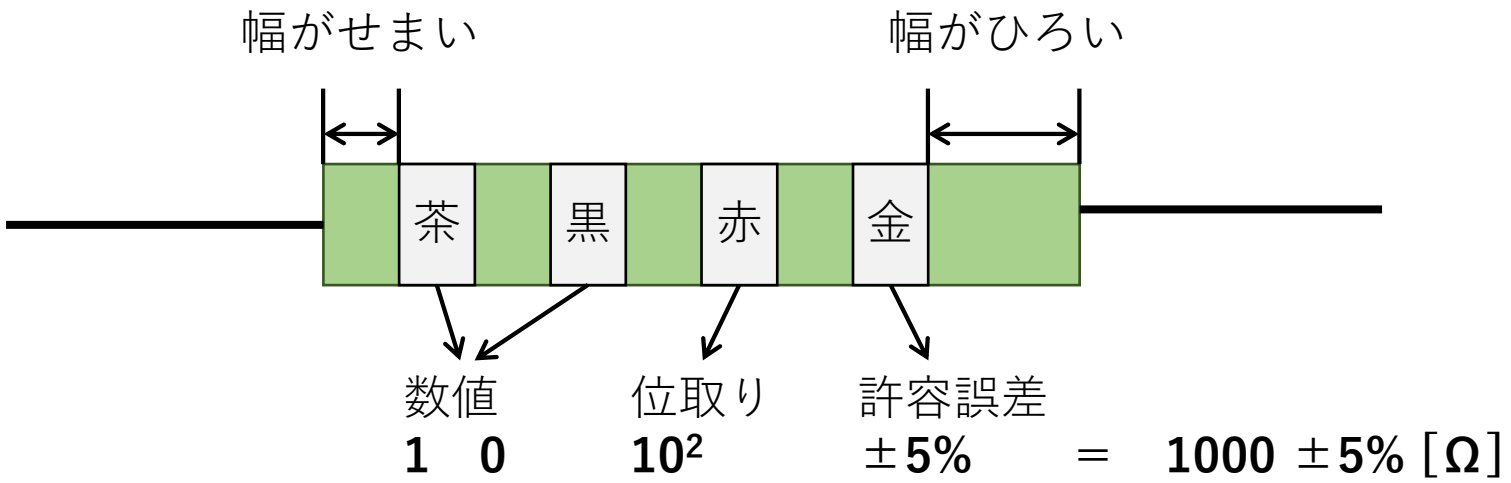
- 抵抗のカラーコードを理解する
- プルアップ抵抗、プルダウン抵抗を理解する
- 外部LEDを光らせるために必要な抵抗の値を求める
- 外部LEDを点滅させる
  - digitalWrite()

【解説】抵抗のカラーコード

- 抵抗はカラーコードで誤差を含む抵抗値を表示
  - 「抵抗値」と「許容誤差」を示す

- カラーコードの読み方
  - カラーコードが**左寄り**になるように読む
  - 左から順に、「数値」「位取り」「許容誤差」

※許容誤差は金色が多く、黄色は少ない



【例題】抵抗値を読む

- 例では「茶・黒・赤・金」のカラーコードなので，$10 \times 10^2 \pm 5\%[\Omega]$と分かる

  $10 \times 10^2 \pm 5\%[\Omega]$の場合

  $1000[\Omega]$の1%は＿＿＿＿＿＿＿＿ $[\Omega]$

  $1000[\Omega]$の5%は＿＿＿＿＿＿＿＿ $[\Omega]$

- したがって，誤差を含む抵抗値の範囲は

  ＿＿＿＿＿＿＿＿ ～ ＿＿＿＿＿＿＿＿ $[\Omega]$

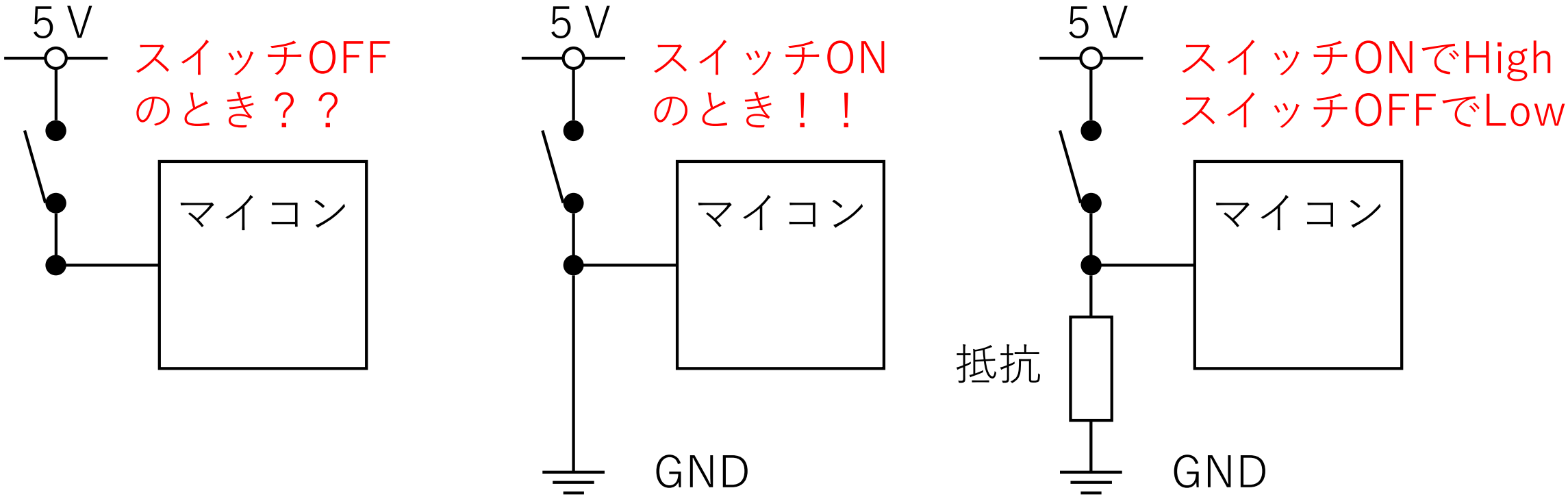
【参考】抵抗のカラーコード表

色	数値	位取り	誤差
黒	0	10^0	$\pm 1\%$
茶	1	10^1	$\pm 2\%$
赤	2	10^2	
橙	3	10^3	
黄	4	10^4	$\pm 0.5\%$
緑	5	10^5	$\pm 0.25\%$
青	6	10^6	$\pm 0.10\%$
紫	7	10^7	$\pm 0.05\%$
灰	8	10^8	
白	9	10^9	
金		10^{-1}	$\pm 5\%$
銀		10^{-2}	$\pm 10\%$

【解説】プルアップ抵抗、プルダウン抵抗

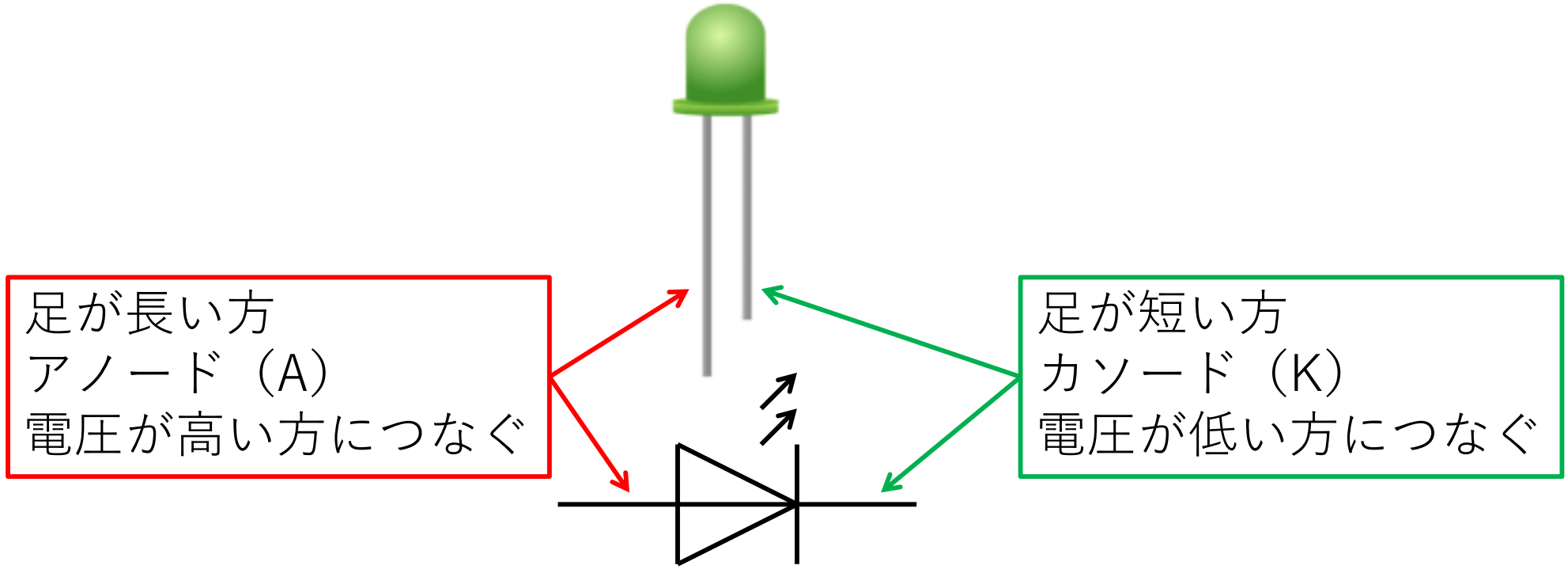
- 電子回路において信号線への入力が切り離されている状態は不安定
  - Lowなのか、Highなのか決まらない

- スイッチがOFFの時にLowかHighに確定するための抵抗



【解説】LEDについて

- LED：Light Emitting Diode（発光ダイオード）
  - 低消費電力（10～30mA程度）　⇔　豆電球は300mA程度
  - 長寿命（数万～10万時間＝連続で数年～15年）
  - 安価（1個数円～数十円）

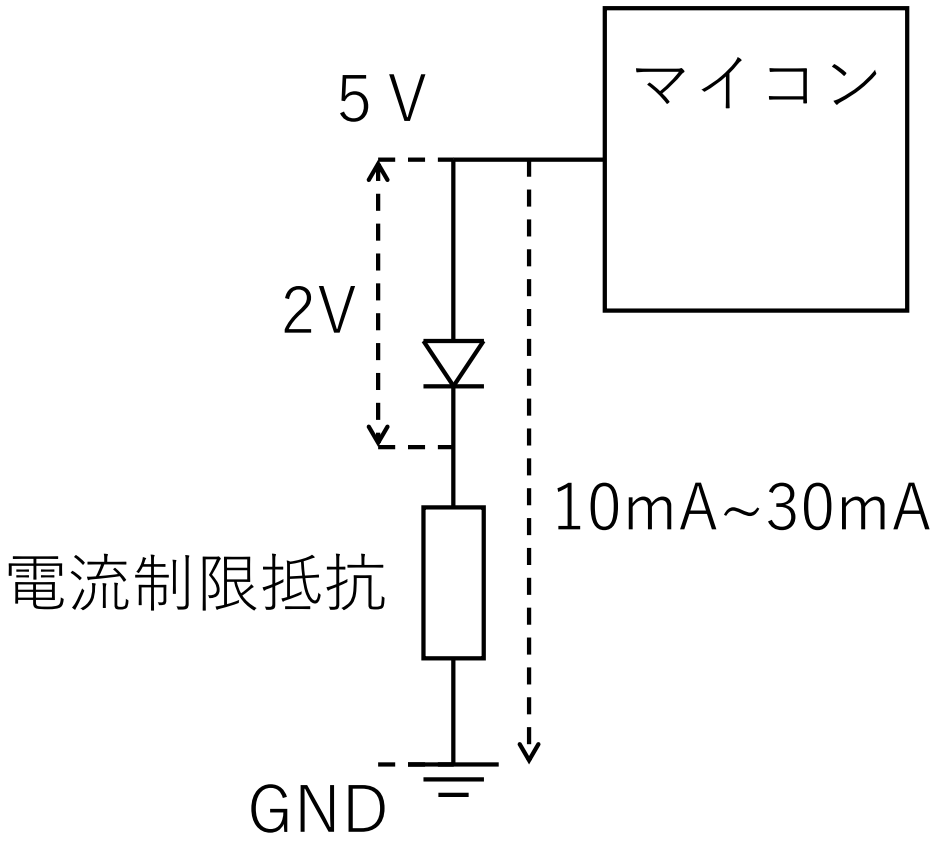


【解説】LEDについて

（注意）LEDと電源は直接つないではいけない
- LEDに過大電流が流れ、LEDが壊れる

- （大雑把な）目安
  - 電圧：LEDに2V程度
  - 電流：10ｍA～30mA程度

※製品や色によって異なるので
データシートを見ること



【例題】電流制限抵抗の計算

- マイコンが5V駆動（High=5V、Low=0V）とする。
- LEDに2Vの電圧がかかると仮定する。
- 電流制限抵抗にはかかる電圧は＿＿＿＿＿＿[V]である。

- LEDに10mAの電流が流れると仮定する。
- オームの法則より、電流制限抵抗の抵抗値は＿＿＿＿＿＿[Ω]である。

- 同様にLEDに30mAの電流が流れると仮定する。
- オームの法則より、電流制限抵抗の抵抗値は＿＿＿＿＿＿[Ω]である。

【例題】200[Ω]の抵抗を探す

- $200[\Omega]$の抵抗なので、$20 \times 10^1 \pm 5\%[\Omega]$を探す

- カラーコード表を参考に
  - 2・・・＿＿＿＿色
  - 0・・・＿＿＿＿色
  - 1・・・＿＿＿＿色
  - 5%・・＿＿＿＿色

【演習（基本）】Arduinoの動作確認

- 確認事項
  - USBケーブルは正しく接続されている？
  - COMポートの設定は正しい？
  - Arduino IDEで正しいボード（Arduino UNO）を選択している？

- 次ページのプログラムをコンパイルして動作確認

※動作確認が終わったらUSBケーブルを抜く

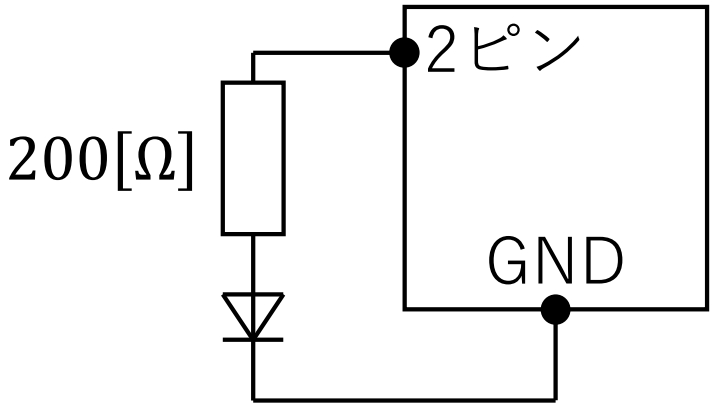
【演習（基本）】LED Pika Pika

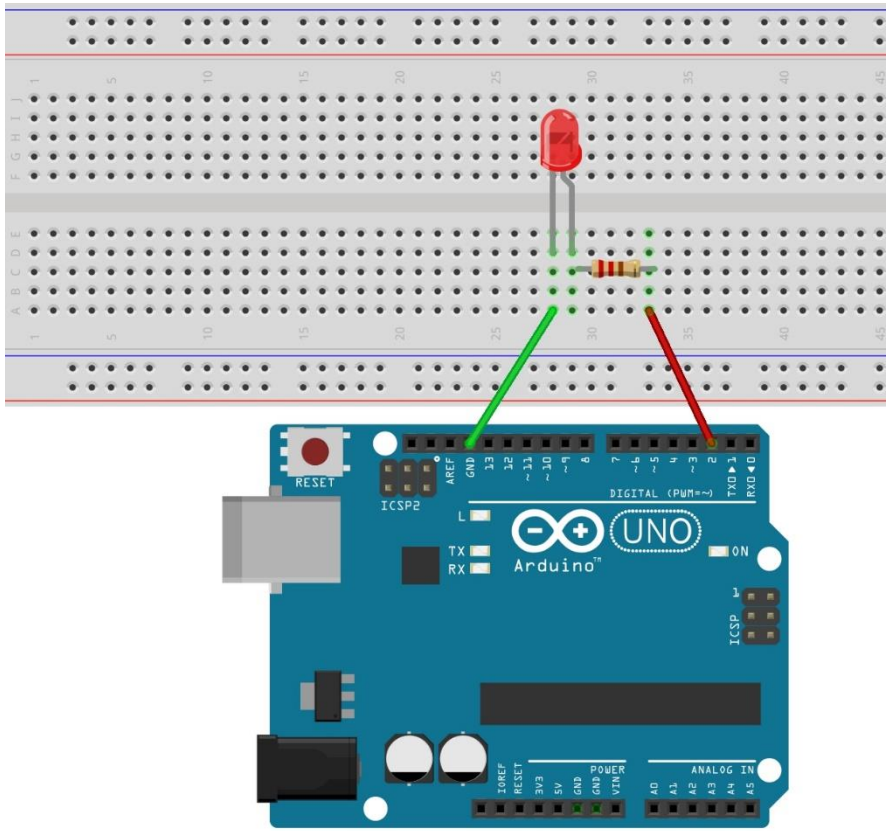

```
int LEDPIN = 13;

void setup()
{
        pinMode(LEDPIN, OUTPUT);
}

void loop()
{
        digitalWrite(LEDPIN, HIGH);
        delay(200);
        digitalWrite(LEDPIN, LOW);
        delay(200);
}
```

【演習（基本）】配線

※PCからUSBケーブルを外してから作業すること

- LED、200[Ω]の抵抗を用意
- Arduinoの2番ピンとGNDを確認
- 回路図を参考に配線をする





※配線ができたらTAに確認依頼（課題2-1）

【演習（基本）】外部LEDを点滅

- 2番ピンを使って外部LEDを点滅させる
  - LED Pika Pikaを参考に**13ピンを2ピンに変更**
  - プログラムのdelay()の数値に応じてLEDが点滅することを確認

※外部LEDが点滅するプログラムができたらTAに確認依頼（課題2-2）

【演習（基本）】外部LEDを点滅

□ 5回点滅したら消灯するプログラムに変更

（やり方の例）

□ loop()を5回繰り返して、以降は消灯

または

□ loop()1回で5回点滅して、以降は消灯

などなど

※ loop()は無限に実行されるため「以降は消灯」がポイント

※5回点滅後に消灯するプログラムができたらTAに確認依頼（課題2-3）

確認がとれたら他の方法を考えて実装してみよう（TAの確認は不要）

きわめる。拓く。創り出す。
長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.2-2

デジタルIO（Input）

【目次】デジタルIO（Input）

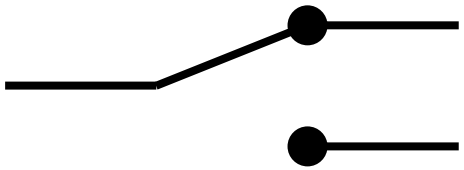
- スイッチについて理解する
- スイッチを使ってLEDを点滅させる

【解説】スイッチの種類

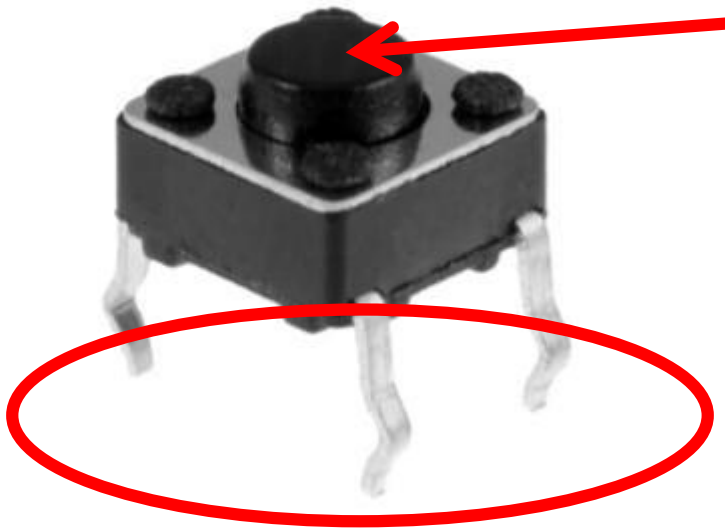
- スイッチには3種類があります
- A接点（m**A**ke）：通常OFF、NO（ノーマリーオープン）

- B接点（**B**reak）：通常ON、NC（ノーマリークローズ）

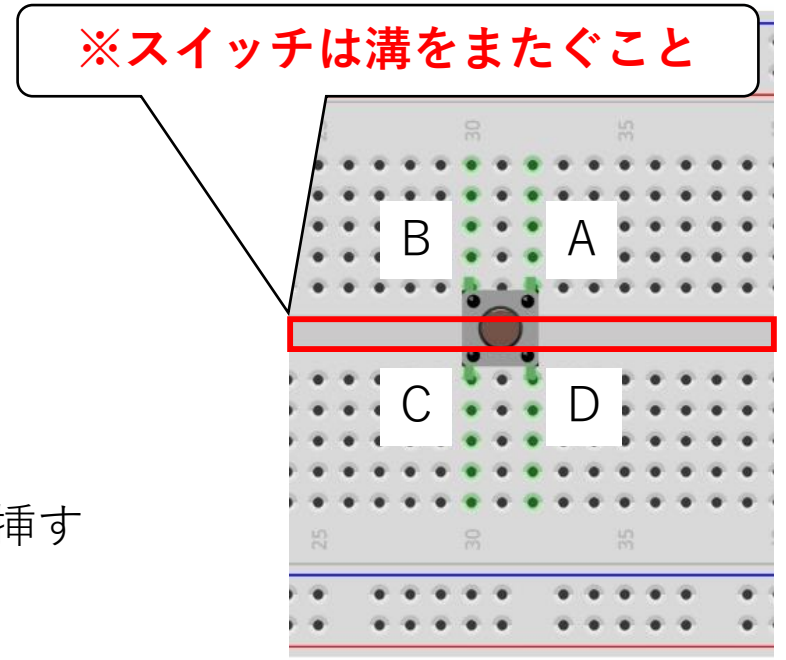
- C接点（**C**hange）：A/B接点切替

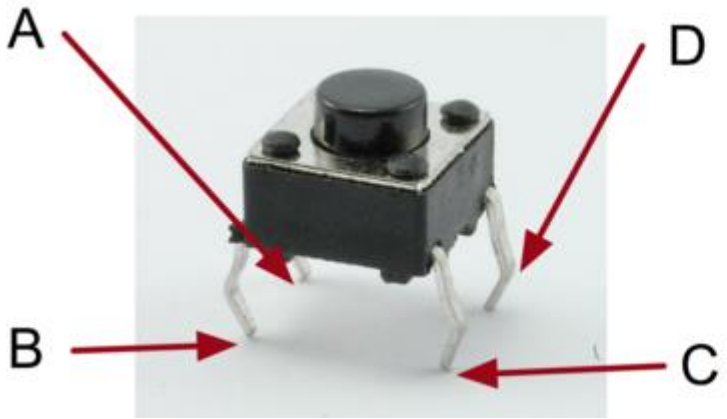


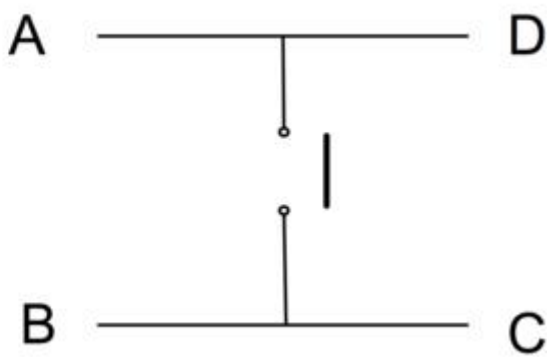
【解説】スイッチ（押しボタン）



ブレッドボードや基板に挿す
※足の向きに注意！！



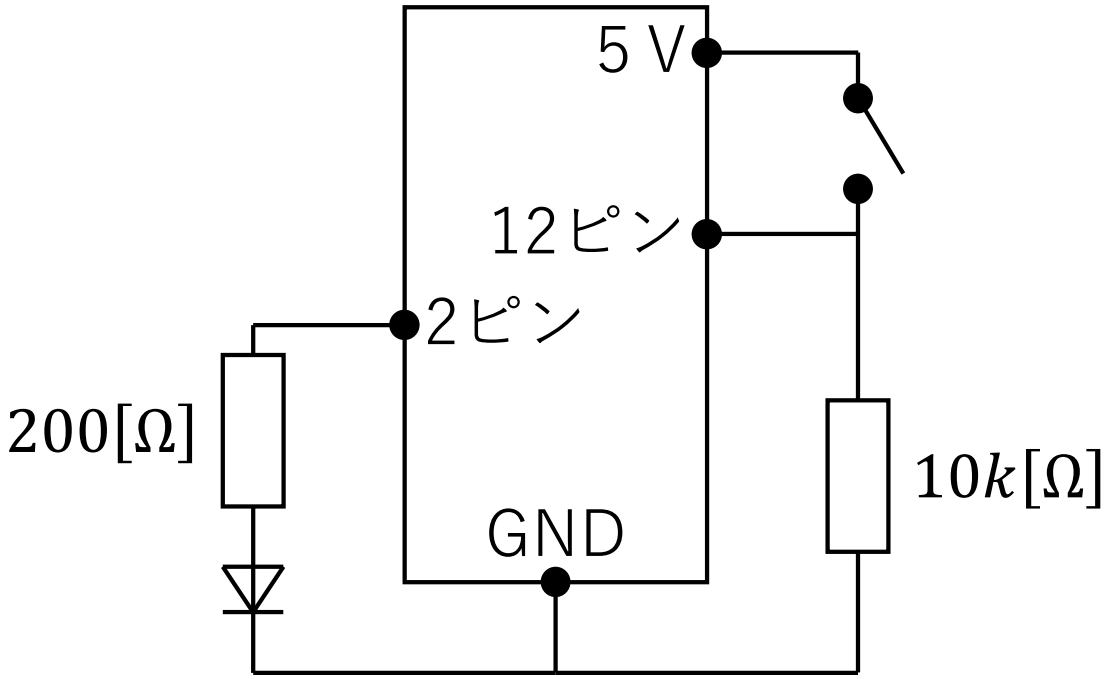


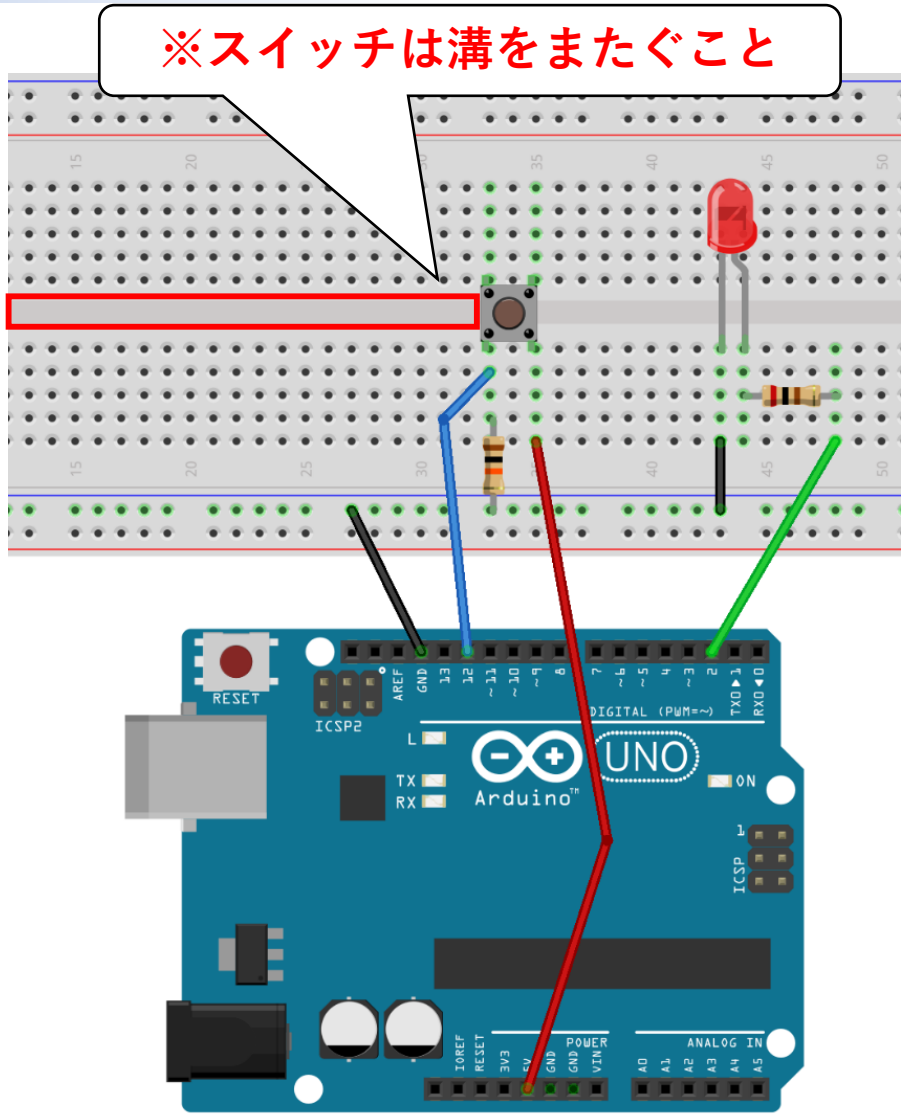


【演習（基本）配線

※PCからUSBケーブルを外してから作業すること

- LED、**$10k[\Omega]$の抵抗**、スイッチを用意
- Arduinoの2番ピンと12番ピンとGNDを確認
- 回路図を参考に配線をする





※配線ができたらTAに確認依頼（課題2-4）

【演習（基本）】外部LEDを点滅

- スイッチを押している間LEDが点灯、離したら消灯する
  - digitalRead()がHIGHなら。。。LOWなら。。。

（例：digitalRead()の使い方）

（変数宣言）

```
int SWITCHPIN = 12;    // スイッチを接続するピンの番号
int sw = 0;            // スイッチの状態を記録する変数
```

（setup関数内）

```
pinMode(SWITCHPIN, INPUT); // ピンの設定
```

（loop関数内）

```
sw = digitalRead(SWITCHPIN);      // ピンの状態をswへ代入
```

※できたらTAに確認依頼（課題2-5）

【演習（応用）】外部LEDを点滅

- スイッチを押す毎にLEDの点滅と消灯を切り替える
  実装したい内容は
  - 最初は消灯した状態
  - スイッチを1度押すとLEDが継続的に点滅する（On-Offを繰り返す）
  - もう一度スイッチを押すとLEDが消灯する

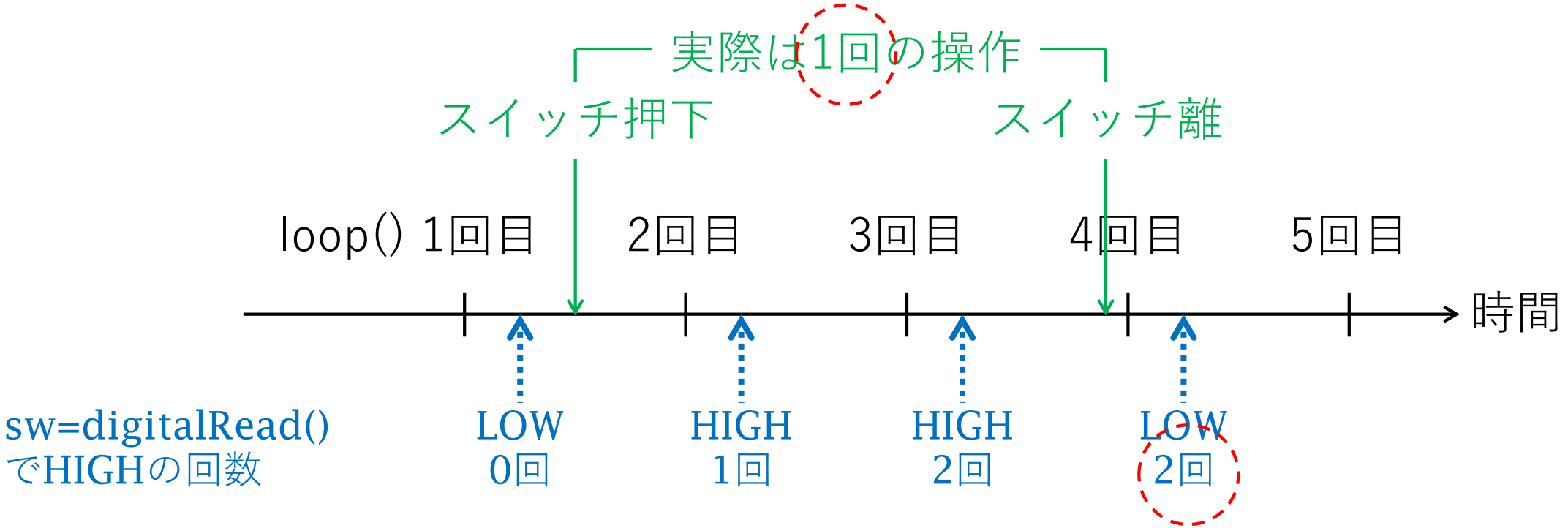
※「スイッチを押した回数が奇数なら点滅、偶数なら消灯」はダメ！！
もう一工夫が必要

※できたらTAに確認依頼（課題2-6）

【解説】スイッチで点滅と消灯の切り替え

※プログラムのポイント（押した回数の奇数・偶数でダメな理由）

- プログラムは高速でループしている（500kHz）
  - スイッチのOn-Offの判定が難しい（下図）
  - delay()は「何もしない状態」なのでdigitalRead()もしなくなる



【解説】スイッチで点滅と消灯の切り替え

※「スイッチを押した回数が奇数なら点滅、偶数なら消灯」はダメ！！
　もう一工夫が必要

どんな機能が必要か？

- 「スイッチを押し下げて，スイッチを離した回数」をカウントする機能
  - 高速でループするloop()がdigitalRead()でスイッチの状態を読み込む
    ⇔delay()で高速ループを遅くすることは**不可**
  - 今回digitalRead()が「1(HIGH)」だったとき，
    - 初めて押し下げた？
    - それとも，前回から押し下げ続けた状態の続き？

    ⇒**今回と前回の状態を把握**
  - **「押し下げて離した回数」をカウント**
    - **カウント値が奇数＝LEDを点滅、偶数＝LEDを消灯**

【解説】スイッチで点滅と消灯の切り替え

■ スイッチのOn-Offについて今回と前回の状態を場合分けすると

今回	前回	状況	判断・行動
0	0	スイッチOff	スイッチを離した
0	1	スイッチOff	
1	0	スイッチOn (押し下げ)	初めてスイッチを押した
1	1	スイッチOn (押し下げ続き)	何もしない

きわめる。拓く。創り出す。
長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.3-1

PWM制御

【目次】PWM制御

- PWM制御（アナログ出力）について理解する

- PWM制御でLEDを明滅させる（点滅ではなく明滅）
  - 点滅：点灯と消灯を繰り返す
  - 明滅：徐々に明るくなる、徐々に暗くなる

- サーボモータの位置制御について理解する

【解説】PWMとは

- パルス幅変調

  PWM：Pulse Width Modulation

- スイッチを高速でOn-Offし、出力される電力を制御する
  - 実際のスイッチは半導体

（参考）

- 周波数変調

  FM：Frequency Modulation

- 振幅変調

  AM：Amplitude Modulation

【解説】PWM制御のイメージ

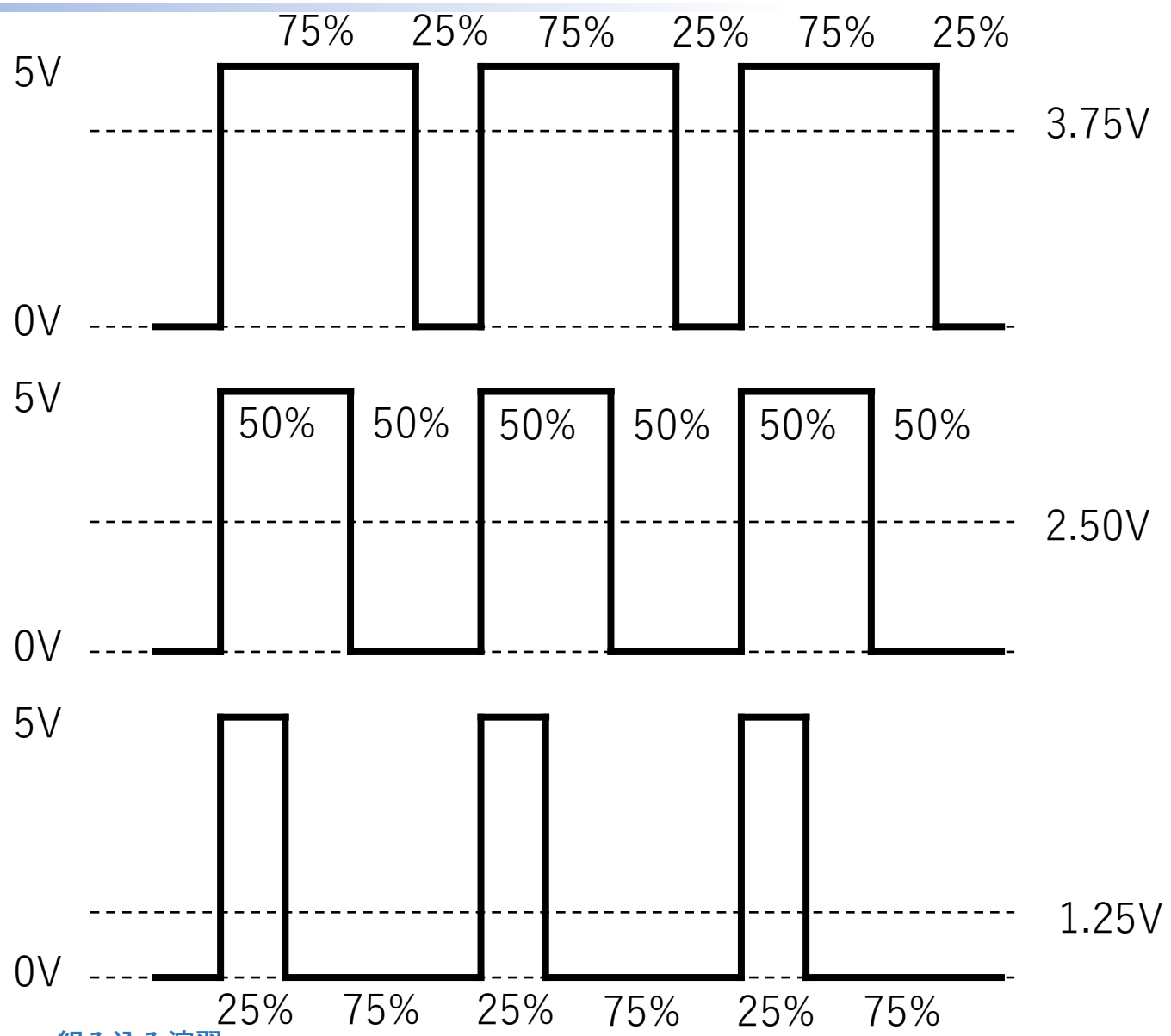
あくまでPWM制御のイメージ（佐藤流）

- もしも自転車をこぐときに
  - A）100%全力で立ちこぎ
  - B）全く何もしない

  の2通りしかできないとする。
- 目的地に着くために、「A）100%全力で立ちこぎ」しかしない場合はとても疲れる。これはエネルギー（回路なら電力）をすごく消費している。
- 「A）100%全力で立ちこぎ」をして、自転車に勢いがついたら「B）全く何もしない」で惰性で進み、またしばらくしたら「A）100%全力で立ちこぎ」して・・・を繰り返した方がエネルギーの消費が少ない。
- モーターを回す際にスイッチを高速でOnとOffを繰り返すと、見かけ上電圧を下げることができ、回転数を下げて消費電力を下げることができる。

【解説】PWM制御とは

- 半導体のスイッチで高速にスイッチをOn-Offする
- Onにする時間とOffにする時間の合計は一定にする
- Onにする時間を長く（Offにする時間を短く）すると電圧は高くなる
- Onにする時間を短く（Offにする時間を長く）すると電圧は低くなる



【解説】PWM制御とは

- デューティー比：1周期に対するONの時間の割合

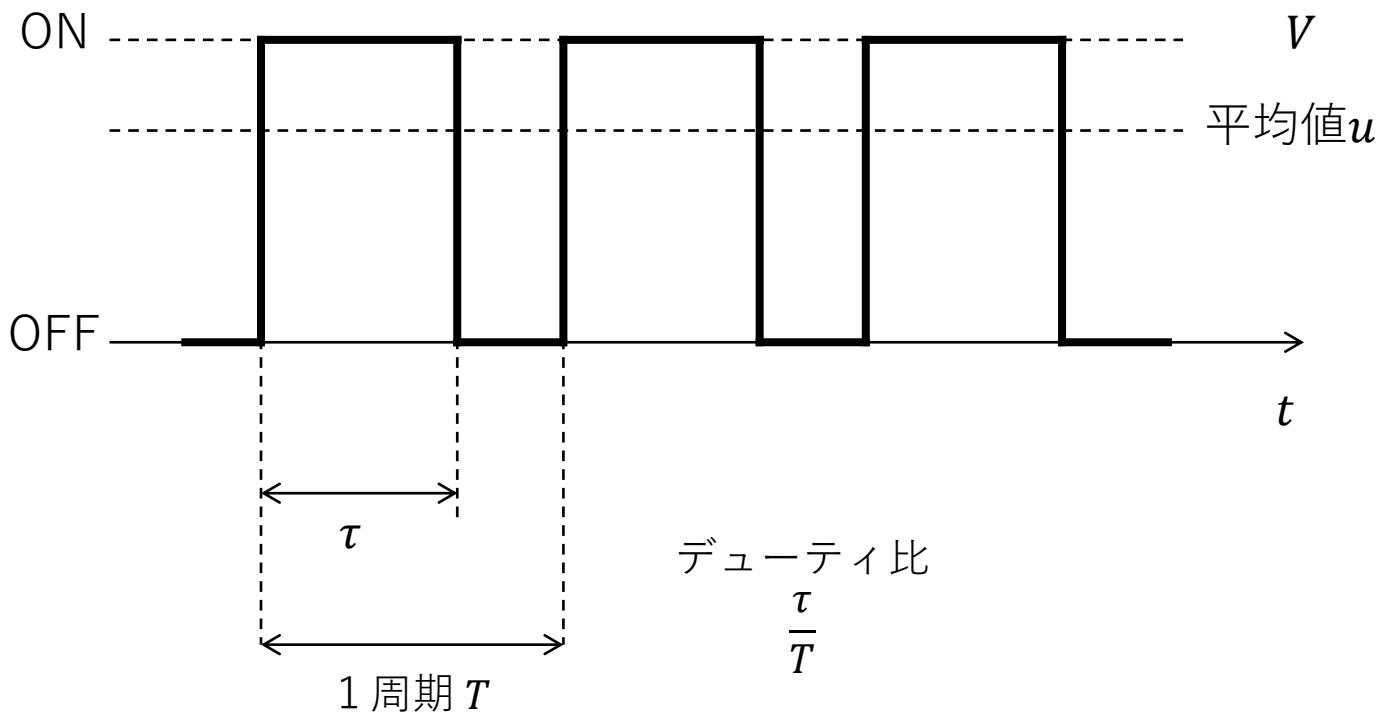
（例）

1周期Tに対してOnの時間τの場合

$$\frac{\tau}{T}$$

- 出力される見かけの電圧（平均値u）は

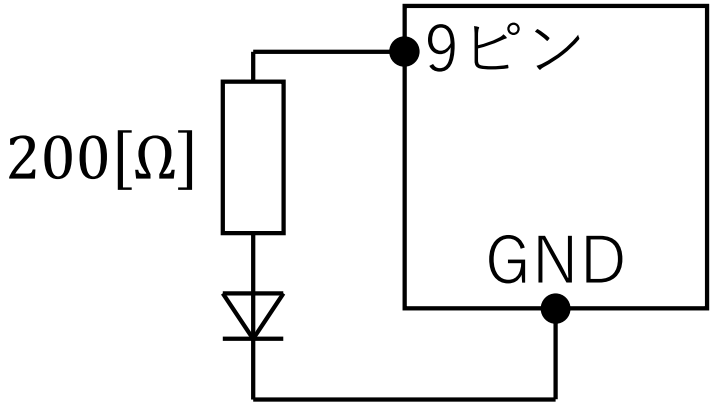
$$u = V \times \frac{\tau}{T}$$

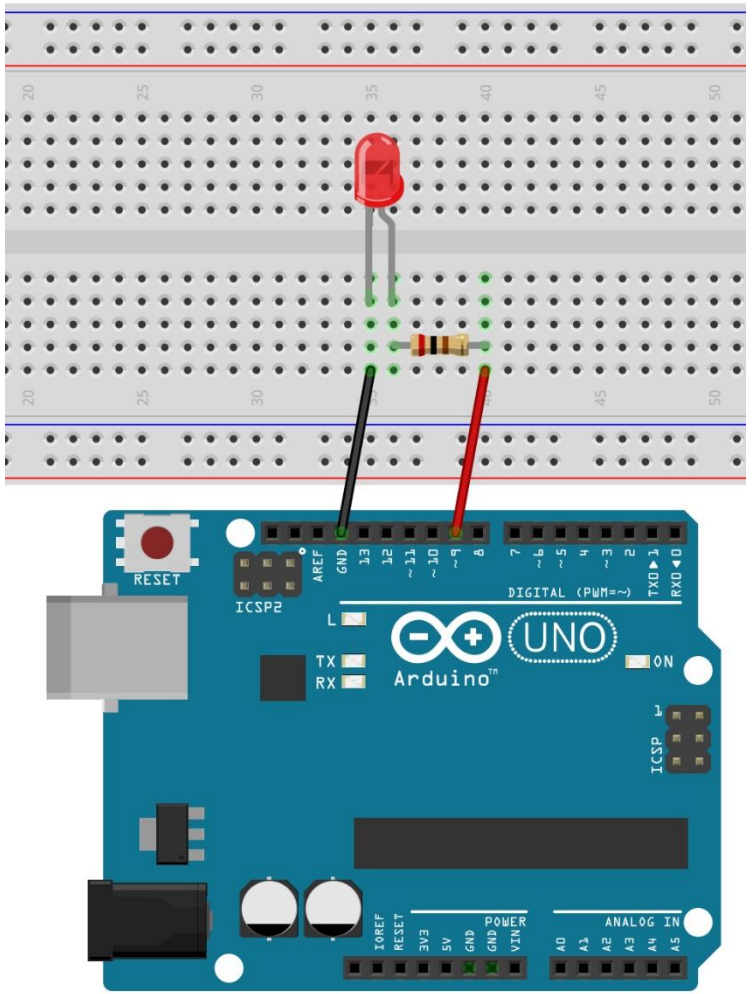


【演習（基本）】回路を組む

※PCからUSBケーブルを外してから作業すること

- LED、200[Ω]の抵抗を用意
- Arduinoの9番ピンとGNDを確認
- 回路図を参考に配線をする





※配線ができたらTAに確認依頼（課題3-1）

【課題（基本）】外部LEDを明滅させる

- （動作確認１）PWM制御を使って外部LEDを点灯させる
  - analogWrite()を使う
  - デューティー比（analogWrite()の引数、下の例では255）を変化させてLEDの明るさが変化することを確認する

```
int LEDPIN = 9;
analogWrite(LEDPIN, 255);
```

- （動作確認2）PWM制御を使って徐々に明るくなる

```
int pwm = 0;
analogWrite(LEDPIN, pwm);
```

PWMの値が

0, 1, 2, ・・・, 253, 254, 0, 1, 2, ・・・

と変化するとよい

※（動作確認2）ができたらTAに確認依頼（課題3-2）

【課題（応用）】外部LEDを明滅させる

- 課題（基本）はいきなり消灯している
- 課題（応用）は次第に暗くなる部分を追加で実装する

- PWMを使って徐々に明るくなる

PWMの値の変化

（前回・基本課題）

0, 1, 2, ・・・, 253, **254, 0,** 1, 2, ・・・

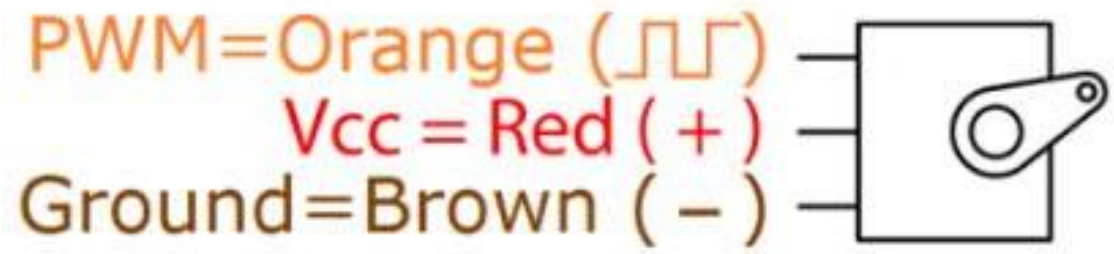
（今回・応用課題）

0, 1, 2, ・・・, 253, **254, 253,** 252, ・・・, 2, **1, 0, 1**, 2, ・・・

※動作確認ができたらTAに確認依頼（課題3-3）

【解説】サーボモーター

- サーボモーター（Servomotor）
  - サーボ機構によって位置や速度を制御可能なモーター
  - ホビー（ラジコン等）用途



※配線ミスをすると壊れるので注意！！

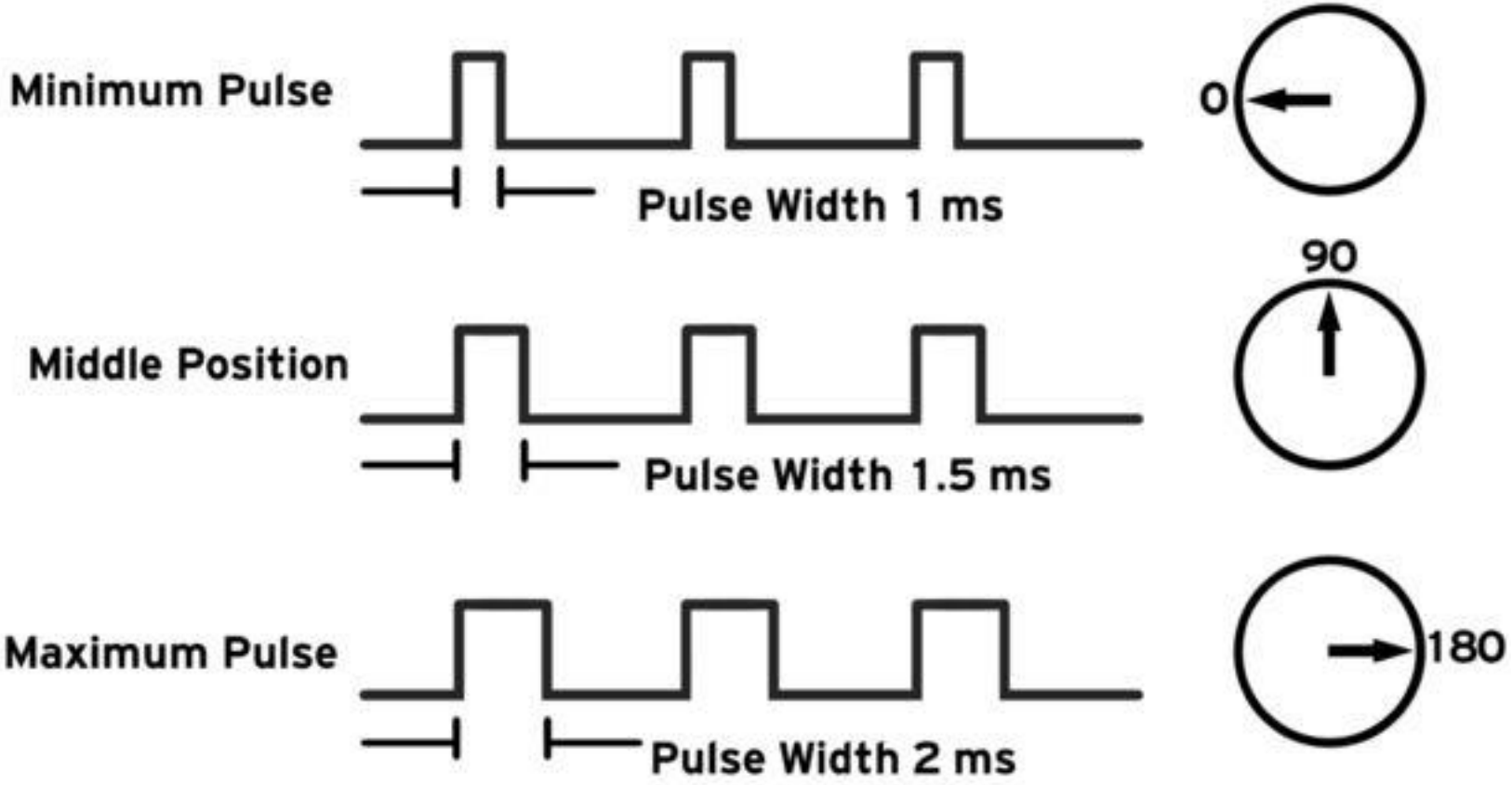
橙：信号線

赤：５V（電源）

茶：GND

【解説】サーボモーターとPWM制御

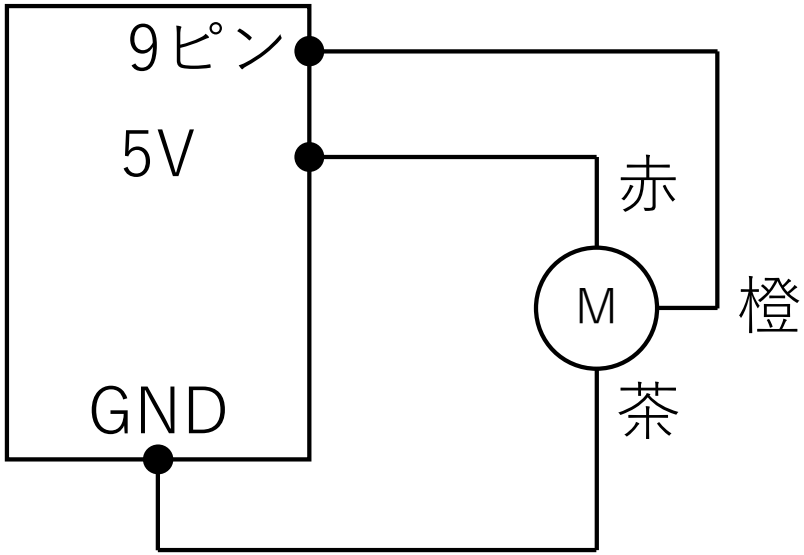
- 一般的なホビー用途のサーボモーターは、パルスのデューティー比で回転角度を制御する
  - パルス幅が1.0ms：
  - パルス幅が1.5ms：
  - パルス幅が2.0ms：

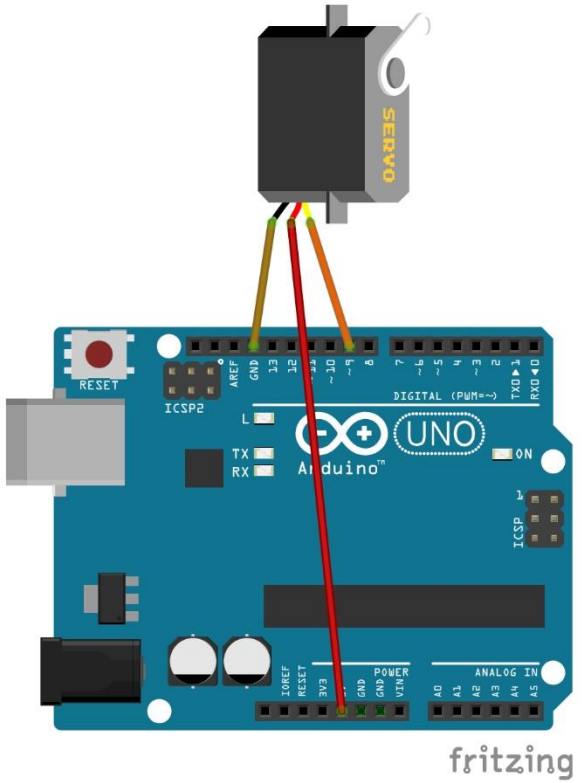


【演習（基本）】動作確認

※PCからUSBケーブルを外してから作業すること

- サーボモーター
  - 橙線→PWM（9ピン）
  - 赤線→５Ｖ（赤）
  - 茶線→GND（茶）





※配線ができたらTAに確認依頼（課題3-4）

【解説】Arduinoのライブラリを活用

- 今回はサーボモータを制御するためのライブラリを使用する
  - 標準的なサーボモータに対し、0度から180度の範囲でシャフトの位置（角度）を指定する。

- attach(pin)
  - pin：サーボを割り当てるピンの番号
- write(angle)
  - angle：サーボに与える値（0度から180度の範囲）

【参考】サーボの角度を指定する


```
#include <Servo.h>
Servo myservo;
void setup() {
      myservo.attach(9);     // 9ピンをサーボの信号ピンとする
      myservo.write(0);      // 0度に初期化
}

void loop() {
      myservo.write(180);
      delay(100);
}
```

【演習（基本）】サーボを左右に振る

- 『サーボの角度を指定する』プログラムを参考にし、サーボホーン（シャフト）を左右に振るプログラムを実装する

※角度を指定して動かす

※変数を使って徐々に動かすのは応用編

※できたらTAに確認依頼（課題3-5）

【演習（応用）】サーボを左右に振る

- 『サーボの角度を指定する』プログラムを参考にし、サーボホーン（シャフト）を左右に振るプログラムを実装する

※forループを使って、左から中、中から右、右から中、中から左へ『徐々に』『滑らかに』動くプログラムにすること

※できたらTAに確認依頼（課題3-6）

きわめる。拓く。創り出す。
長崎総合科学大学
Nagasaki Institute of Applied Science

K7：組み込み演習　No.3-2

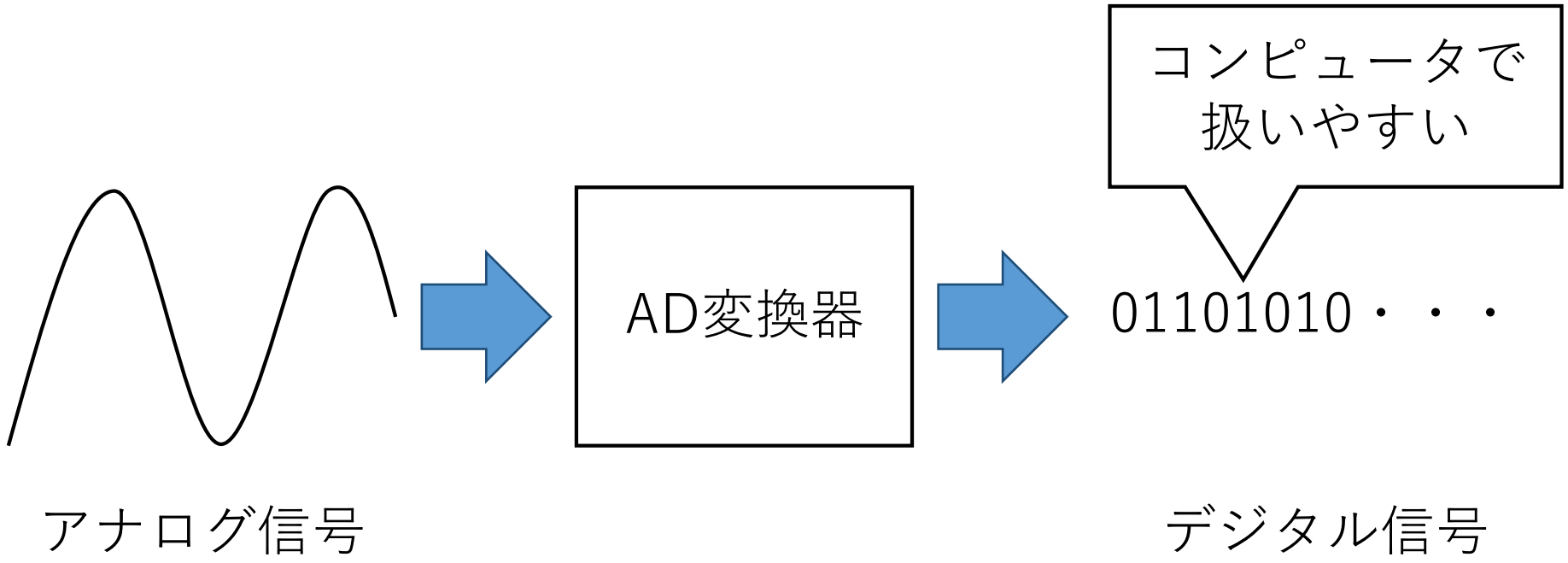
Analog Digital 変換

【目次】 Analog Digital変換

- Analog Digital変換について理解する
- 分解能について理解する
- 可変抵抗の値をシリアル通信で表示する
- 可変抵抗の値に応じて外部LEDを明滅
- 可変抵抗の値に応じてサーボモータを制御（PWM制御）

【解説】Analog-Digital変換（AD変換）

- アナログ（Analog）信号をデジタル（Digital）信号に変換
  - センサで測定した変位、圧力、温度、光、音などを変換



【解説】AD変換の手順

1. 標本化（sampling）
   - アナログ信号を一定周期ごとに取り出す

2. 量子化（digitizing）
   - 標本化したアナログ量をデジタル数値へ変換

3. 符号化（coding）
   - デジタル数値をnビットの2進数へ変換

【解説】AD変換器の分解能

- nビットのAD変換器の分解能は

$$\frac{V}{2^n - 1}$$

ただし、 Vはフルスケール電圧を表す

【例題】AD変換器の分解能

- フルスケール電圧Vが$5V$で、3ビットのAD変換器の場合は、

$$\frac{V}{2^n - 1} = \frac{5}{2^3 - 1} \approx$$

したがって、＿＿＿＿＿＿＿＿[V]刻みで8段階（2^3ビット）で変換される

- センサから出力される電圧をV_{in}としたとき

$0 \leq V_{in} < 0.714$　　変換されて0
$0.714 \leq V_{in} < 1.428$　　変換されて1
⋮
$4.284 \leq V_{in} < 4.998$　　変換されて7

【例題】AD変換の分解能

- フルスケール電圧Vが$5V$で、**8ビット**のAD変換器の場合は、

$$\frac{V}{2^n - 1} \approx$$

したがって、＿＿＿＿＿＿[V]刻みで変換される

- フルスケール電圧Vが$5V$で、**12ビット**のAD変換器の場合は、

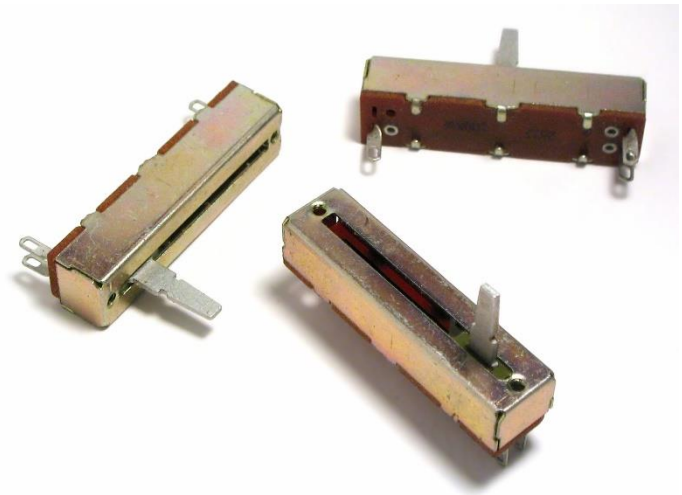
$$\frac{V}{2^n - 1} \approx$$

したがって、＿＿＿＿＿＿[V]刻みで変換される

【解説】ポテンショメーター

- ポテンショメーター（Potentiometer）、可変抵抗
  - 回転角度や移動量を電圧に変換する機器・素子
  - ロータリー型：回転角度を検出
  - リニア型：直線上の位置を検出

ロータリー型



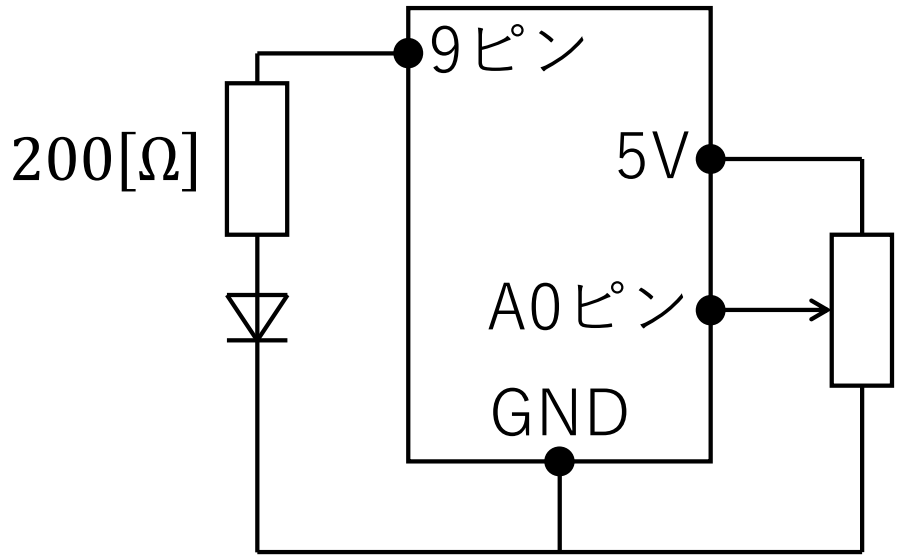
リニア型

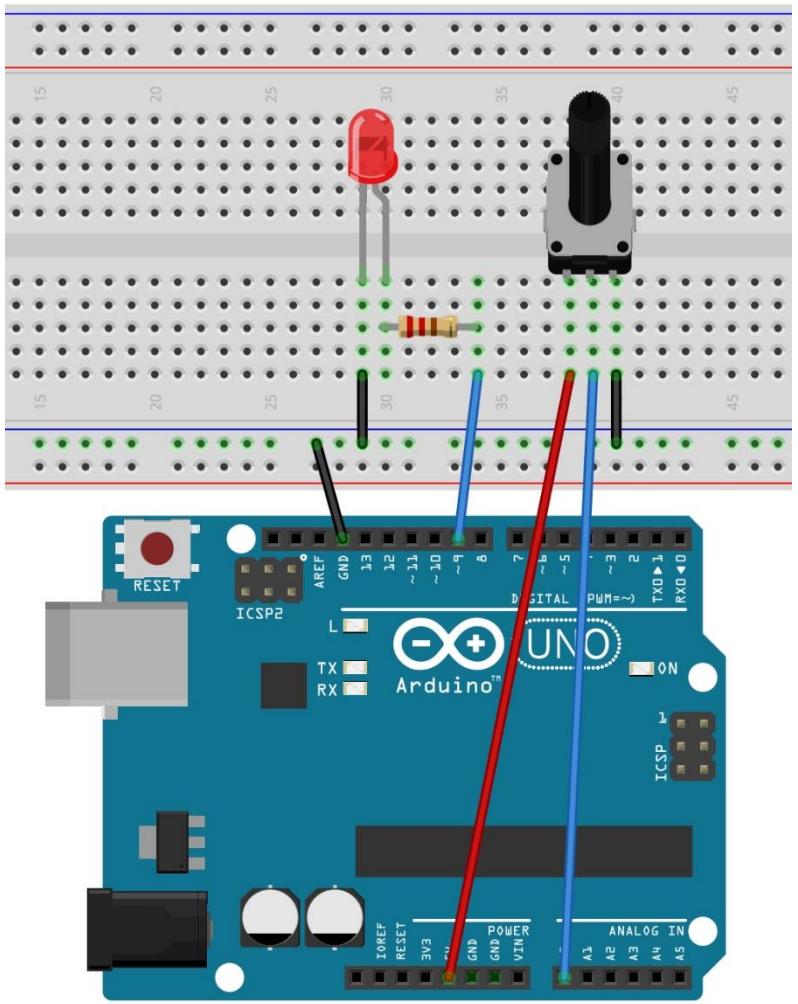
Wikipediaより

【演習（基礎）】配線

※PCからUSBケーブルを外してから作業すること

- ポテンショメーター（10kΩ）
  - 5V（赤）
  - A0（青）
  - GND（黒）
- 抵抗（200Ω）
  - 9ピン（青）
  - （LEDへ）
- LED
  - （抵抗へ）
  - GND（黒）





※配線ができたらTAに確認依頼（課題3-7）

【演習（基礎）】動作確認

- （動作確認1）外部LEDが点滅するか確認
  - 「LEDPikPika」の「13ピン」を「9ピン」に変更
  - 外部LEDを点滅させる

- （動作確認2）AD変換できているか確認
  - 「A0ピン」を「INPUT」に設定
  - analogRead(A0)の返り値をシリアルモニタで表示

```
int adval = 0;
adval = analogRead(A0);
Serial.print(adval);
```

  - ポテンショメーターを回すとシリアルモニタの値が変化することを確認

※（動作確認2）ができたらTAに確認依頼（課題3-8）

【演習（基礎）】動作確認

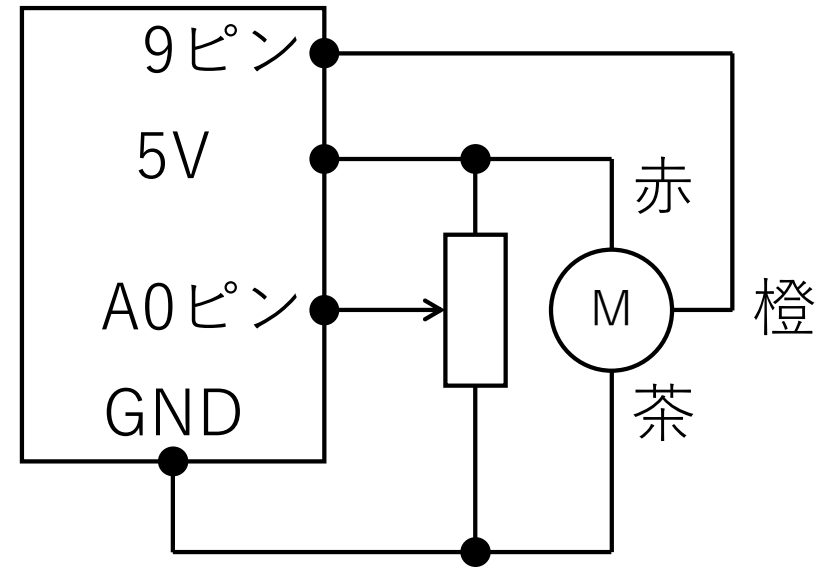
- （動作確認3）ポテンショメーターの値に応じてLEDの点滅時間が変更されているか確認
  - analogRead(A0)で得られた値をdelay()の引数として与える

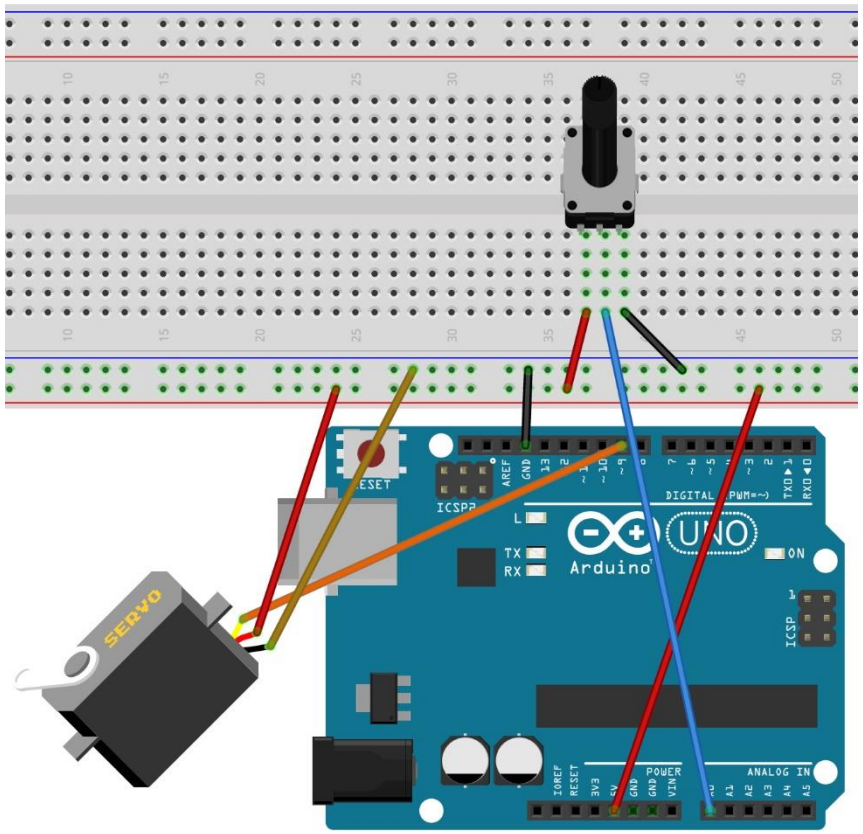
※（動作確認3）ができたらTAに確認依頼（課題3-9）

【演習（応用）】動作確認

※PCからUSBケーブルを外してから作業すること

- ポテンショメーター（10kΩ）
  - 5V（赤）
  - A0（青）
  - GND（黒）
- サーボモーター
  - 橙線→PWM（9ピン）
  - 赤線→５V（赤）
  - 茶線→GND（茶）





※配線ができたらTAに確認依頼（課題3-10）

【演習（応用）】動作確認

- ポテンショメーターの値に応じてサーボモータが左⇔中央⇔右と動くことを確認
  - ポテンショメーターの値をanalogRead(A0)で読み取る
  - 読み取った値をPWM信号として出力したい

※analogRead()の出力は0から1023
※myservo.write()の引数は0から180
※したがって、analogRead()で得られた値（0から1023）を「上手に」0から180の範囲に換算してmyservo.write()へ渡す必要がある

※動作確認ができたらTAに確認依頼（課題3-11）