

レゴ<sup>®</sup> マインドストーム<sup>®</sup> EV3版

中学校プログラミング教育

# 学習指導案

# 補助資料





# 目次

## 中学校1年生編 P 1

1時間目	2
2時間目	10
3時間目	25
4時間目	36
5時間目	43

## 中学校2年生編 P 55

4/20時間目	57
6/20時間目	61
7/20時間目	62
8/20時間目	65
9/20時間目	75
10/20時間目	77
11/20時間目	85
12～15/20時間目	87
16～19/20時間目	93

## 中学校3年生編 P 97

1時間目	98
2時間目	99
3時間目	103
4時間目	108
5時間目	110
6時間目	114
7～8時間目	116
9時間目	118

## 県内事例集 P 128

# 中学校 1 年生編

## 単元の構成

1時間目：プログラミングでコンピュータに指示してみよう(Hour of Code)

2時間目：センサでコンピュータを動かそう(micro:bit)

3時間目：計算機を作ろう (Studuino)

4時間目：計算クイズを作ろう (Studuino)

5時間目：分かりやすく・使いやすくしよう (Studuino)

# 1 時間目：プログラミングでコンピュータに指示してみよう (Hour of Code)

## ①使用するソフトウェア

Hour of Code (アワー・オブ・コード) <http://code.org/hoc>

- コンピュータサイエンス※教育週間に行われるプログラミング体験活動
  - ・ 全世界的なムーブメント
  - ・ 幅広い対象向けの教材開発

※コンピュータ活用全般を扱う学問分野 (プログラミングは構成要素の1つ)

- 教材の目的
  - ・ コンピュータサイエンスに対する興味を持たせる
  - 注) プログラミング技術を学ぶものではない
- パズル型プログラミング
  - ・ コードブロックの組み合わせで迷路などの課題を解く

## ②ソフトウェアの操作方法

②-1

ブラウザ (インターネットエクスプローラなど) を立ち上げる



インターネット  
エクスプローラ



グーグル  
クローム



マイクロソフト  
エッジ



サファリ  
※アイパッド等

②-2

Hour of Codeのサイトへ移動

※Googleの例

http://code.org/hoc  
(URLを入力)

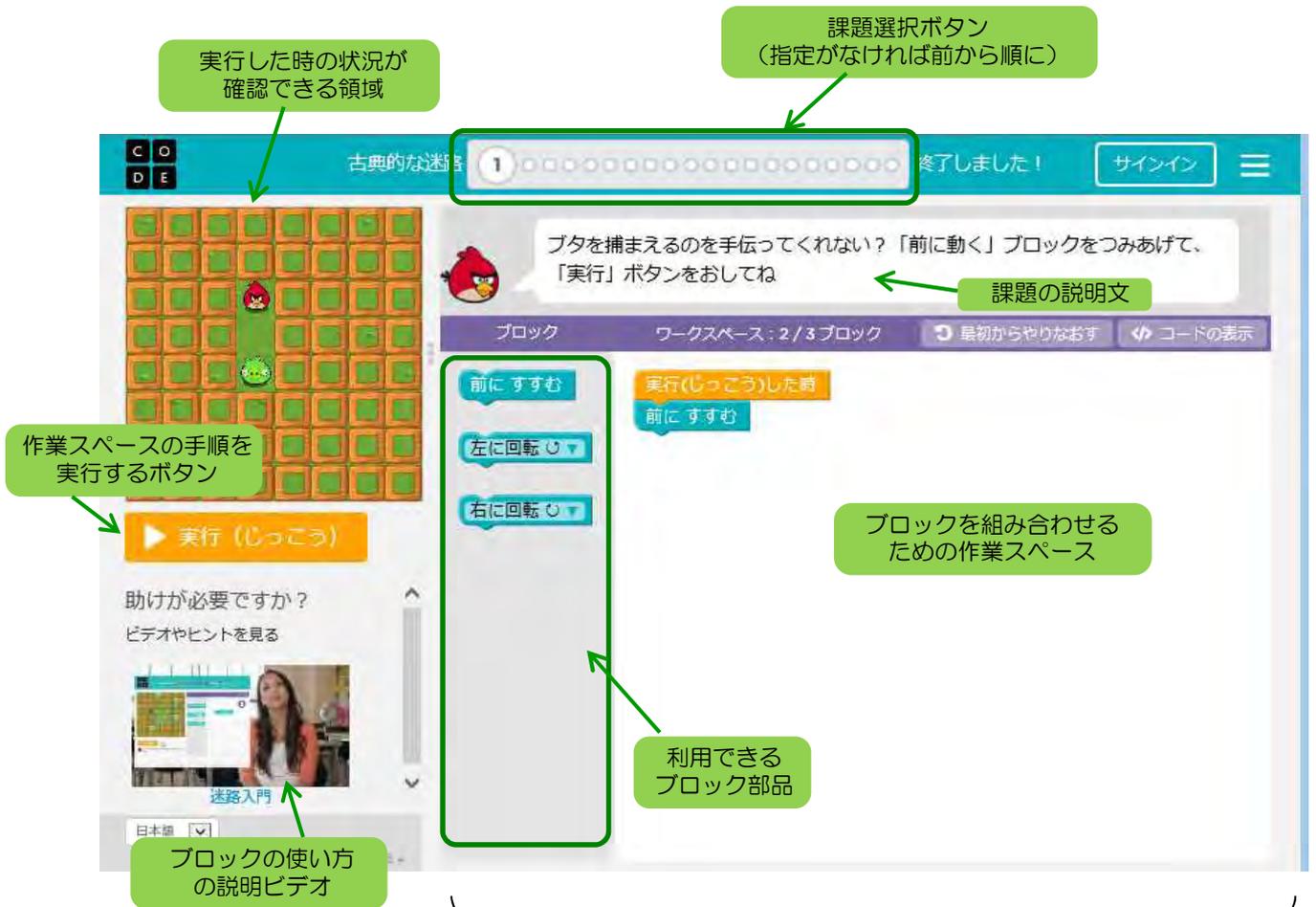
↓

入力後「Enter」  
※クリックは不要

動画が流れる。  
スキップする時は  
×ボタンを押す

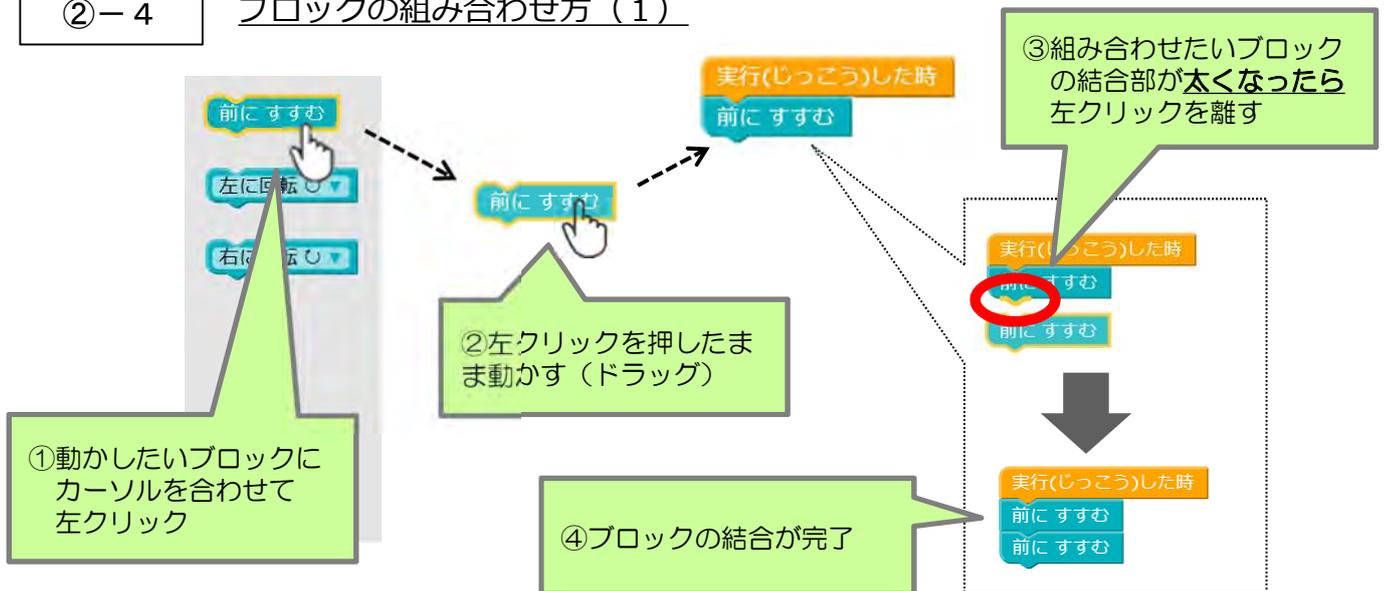
②-3

Hour of Codeの操作画面



②-4

ブロックの組み合わせ方 (1)



②-5

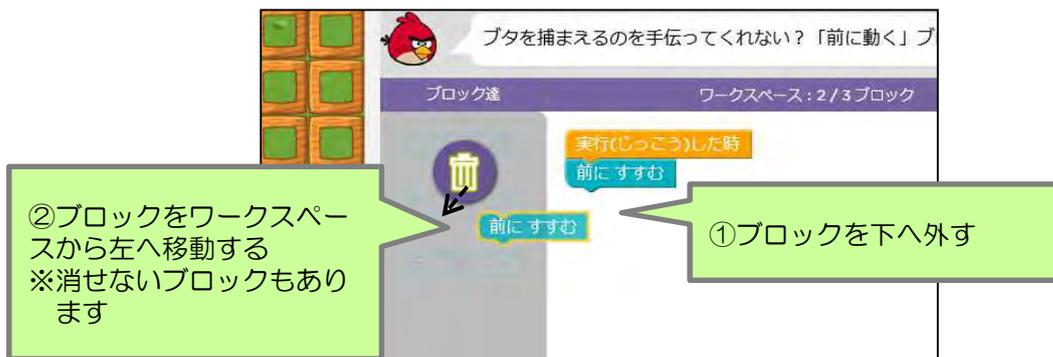
## ブロックの組み合わせ方 (2)



※「繰り返しブロック」「条件分岐ブロック」の詳細はP6参照

②-6

## ブロックの消し方



②-7

## プログラムの実行

① ブタを捕まえるプログラムを組んだら「実行（じっこう）」を押す

② プログラムは上から順に実行される

③ クリアした場合のメッセージ

「次へ」を押せば  
次の課題（パズル）へ移動

The screenshot shows a coding environment with a grid of green blocks on the left and a block-based code editor on the right. A red arrow points to the '実行（じっこう）' (Execute) button. A callout box on the right shows a success message: 'おめでとうございます！あなたはパズル1をクリアしました。' (Congratulations! You have cleared puzzle 1.) with a '次へ' (Next) button.

②-8

## 手動での課題（パズル）の変更

古典的な迷路 1 終了しました！

画面上部にある課題選択ボタンをクリック（1～20まで）

The screenshot shows a horizontal bar with a grid of 20 circles representing puzzle topics. The first circle is highlighted and labeled '1'. The text '古典的な迷路' is on the left and '終了しました！' is on the right.

### ③大切なブロック

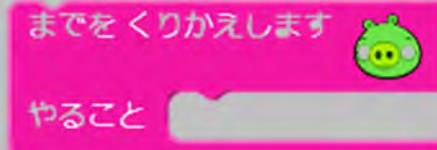
- プログラミング教育においては、「繰り返し」や「条件分岐」の概念を理解することが重要です。
- 授業においては、生徒が「繰り返し」「条件分岐」の両方のブロックに触れられるよう、状況を見ながら進行を工夫してください。

#### 〈繰り返し〉



(回ブロック)

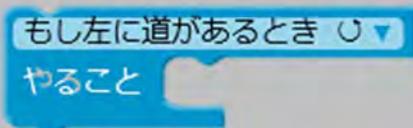
- 回数を指定して繰り返す
  - 内に含んだブロックを、**指定された回数だけ**実行する
- 課題6で登場



(までブロック)

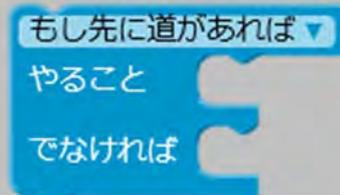
- ある状況になるまで繰り返す
  - 内に含んだブロックを、**キャラクターに会うまで何度も**実行する
- 課題10で登場

#### 〈条件分岐〉



(もしブロック)

- 条件を指定して実行する
  - 内に含んだブロックを、**指定された条件が満たされる時だけ**実行する
- 課題14で登場



(もし/でなければブロック)

- 条件を指定して、満たされる時と、満たされない時とで、**場合分け**して実行する
- 課題18で登場

#### ④プログラム例

- 正しいプログラムが一つとは限りません。確実に目的を達成するのであれば、それは正しいプログラムです。
- 大切なことは、その動きを自分の言葉で説明できることです。それができれば、工夫の余地にも気づきやすいでしょう。

〈課題 1〉

```
実行(じっごう)した時
前にすすむ
前にすすむ
```

〈課題 2〉

```
実行(じっごう)した時
前にすすむ
前にすすむ
前にすすむ
```

〈課題 3〉

```
実行(じっごう)した時
前にすすむ
前にすすむ
右に回転 90度
前にすすむ
```

〈課題 4〉

```
実行(じっごう)した時
前にすすむ
左に回転 90度
前にすすむ
右に回転 90度
前にすすむ
```

〈課題 5〉

```
実行(じっごう)した時
右に回転 90度
前にすすむ
左に回転 90度
前にすすむ
前にすすむ
前にすすむ
左に回転 90度
前にすすむ
```

〈課題 6〉

```
実行(じっごう)した時
くりかえし 5回
  やること 前にすすむ
```

繰り返しの回数を  
手入力に変更

〈課題 7〉

```
実行(じっごう)した時
右に回転 90度
くりかえし 5回
  やること 前にすすむ
```

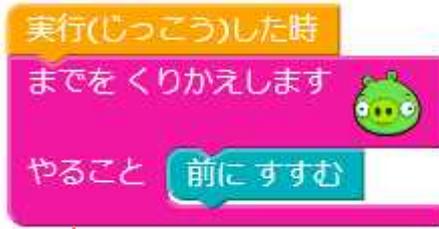
〈課題 8〉

```
実行(じっごう)した時
くりかえし 4回
  やること 前にすすむ
左に回転 90度
くりかえし 5回
  やること 前にすすむ
```

〈課題 9〉

```
実行(じっごう)した時
くりかえし 3回
  やること 前にすすむ
  やること 前にすすむ
  やること 右に回転 90度
```

〈課題10〉



キャラクターに会うまで  
プログラムを繰り返すブロック

〈課題11〉



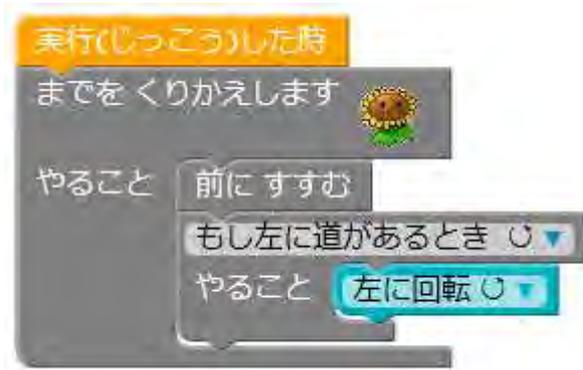
〈課題12〉



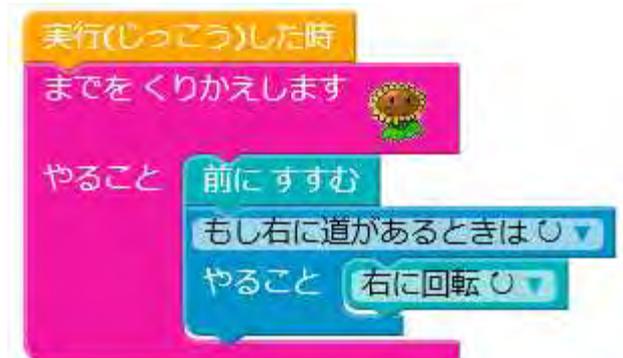
〈課題13〉



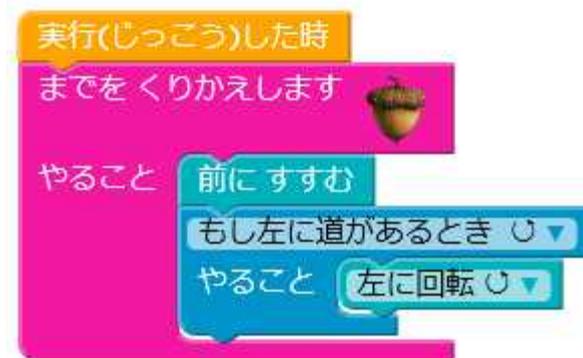
〈課題14〉



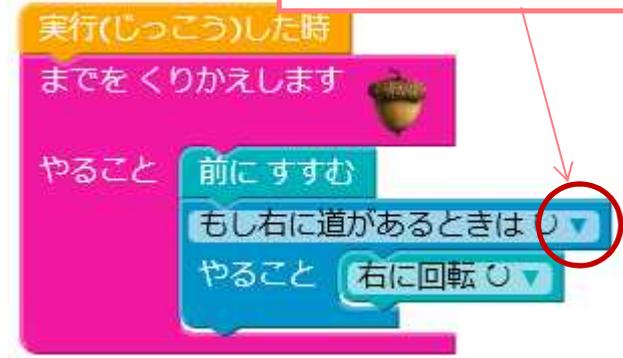
〈課題15〉



〈課題16〉

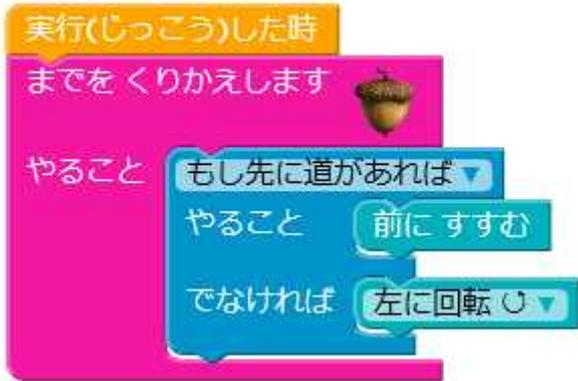


〈課題17〉

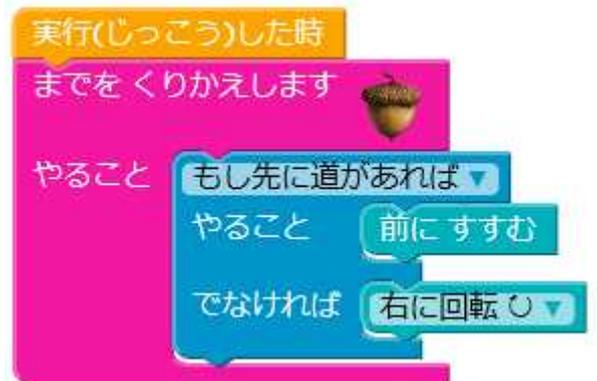


「もし左…」をプルダウン  
メニューから  
「もし右…」に変換

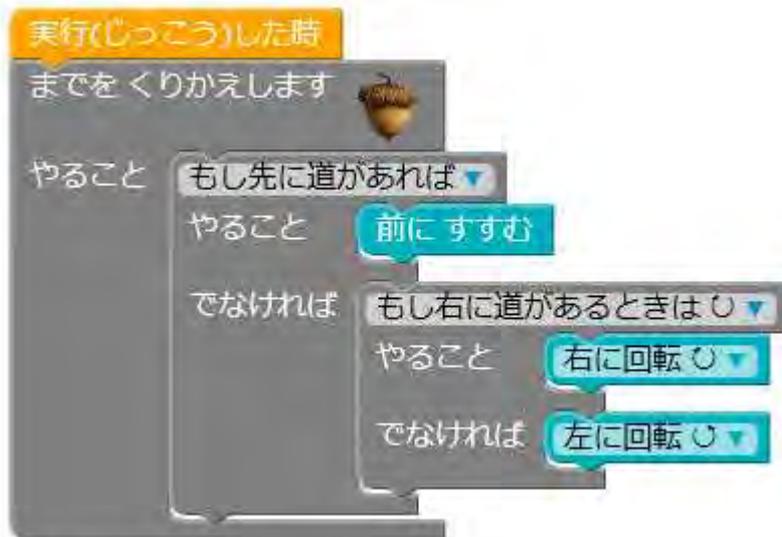
〈課題18〉



〈課題19〉



〈課題20〉



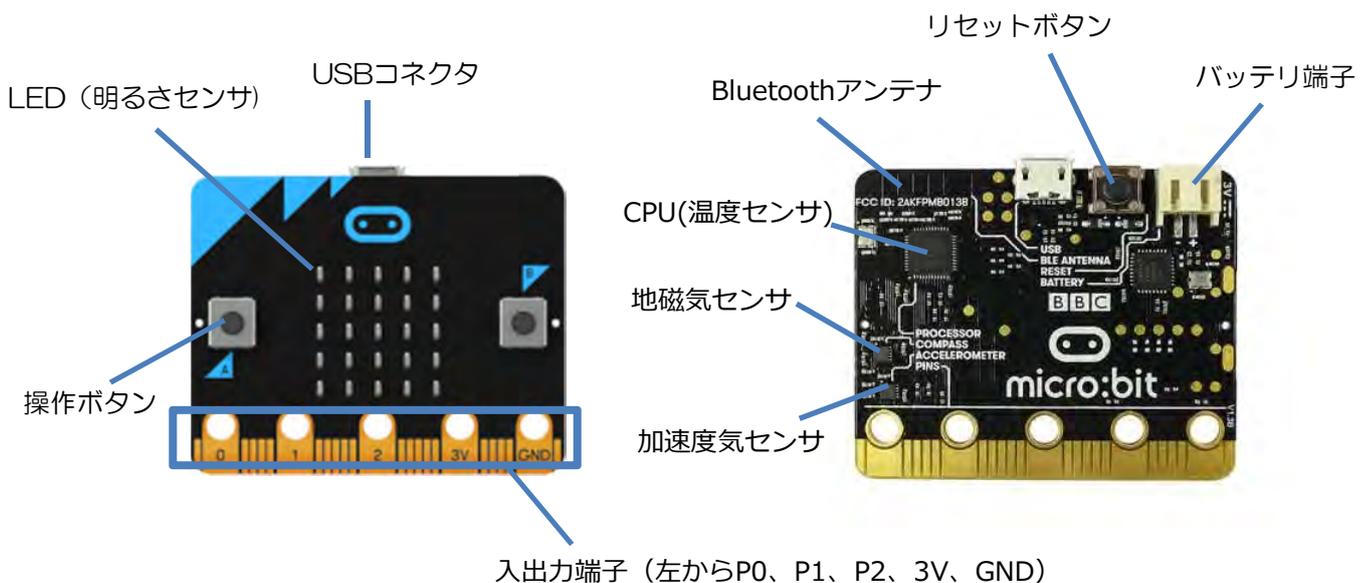
## 2時間目：センサでコンピュータを動かそう(micro:bit)

### ①使用する教材

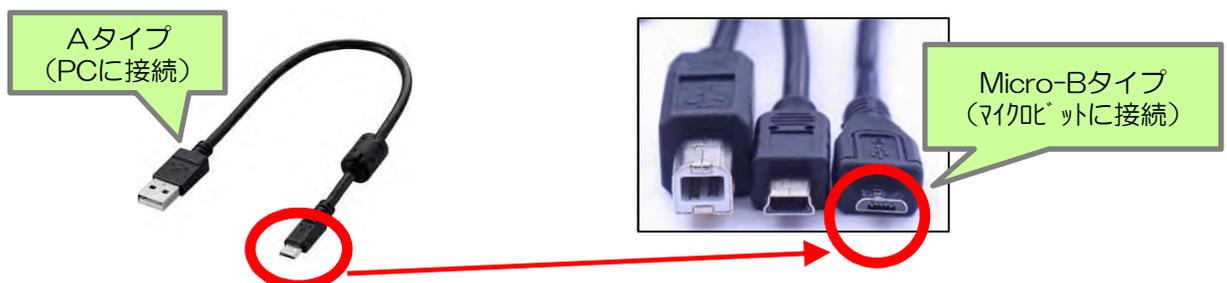
micro:bit (マイクロビット)

<https://makecode.microbit.org/>

- BBC (イギリス放送協会) が企画・開発した教育用シングルボードコンピュータ
  - ・ 11, 12歳の小学生に無料配布
- 単体でできること
  - ・ 簡易表示 (LEDマトリクス)
  - ・ センサ (ボタン, 加速度 (重力), 明るさ, 磁気)
  - ・ 無線通信 (Bluetooth Low Energy)
  - ・ 汎用入出力端子 (25ピンコネクタ)
  - ・ モーターやスピーカーの出力, 人のタッチ入力
- ブロックで作ったプログラムをダウンロードして実行



- パソコンとマイクロビットを接続するには、USBケーブル (片方がAタイプ、もう片方がMicro-Bタイプ) が必要です。 ※Bluetoothで接続する場合は不要



## ②パソコン画面上（エディター）の操作方法

②-1

ブラウザ（インターネットエクスプローラなど）を立ち上げる



インターネット  
エクスプローラ



グーグル  
クローム



マイクロソフト  
エッジ

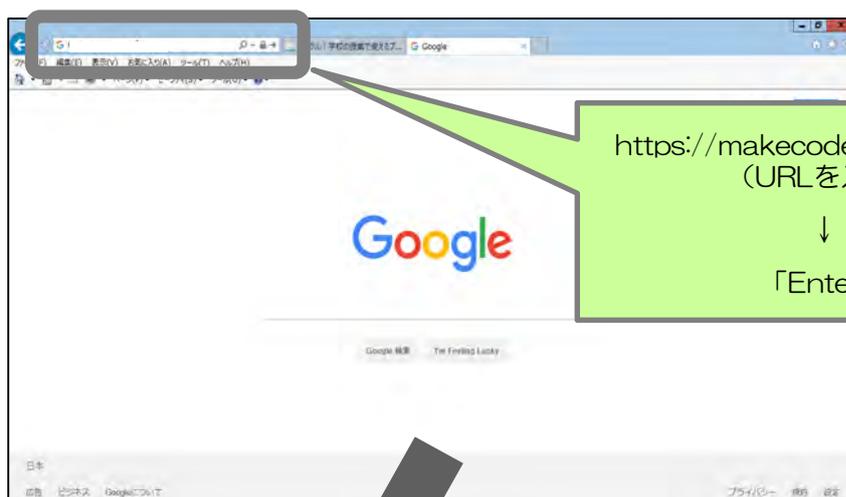


※アイパッドの場合は  
マイクロビットのアプリの  
インストールが必要

②-2

マイクロビットのサイトへ移動

※Googleの例



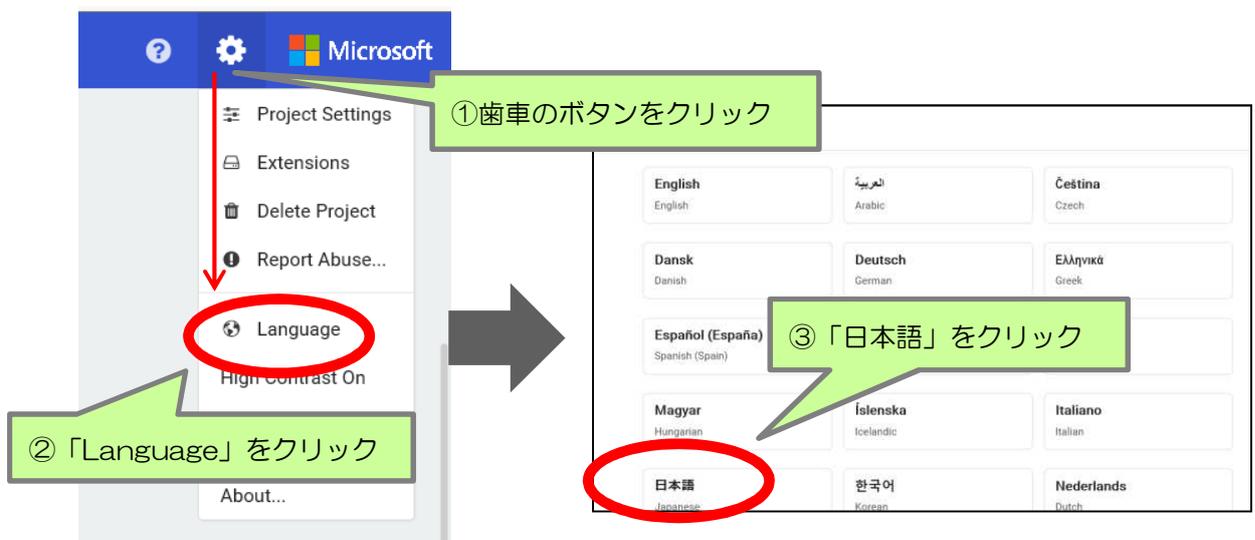
②-3

### マイクロビットのエディターの操作画面



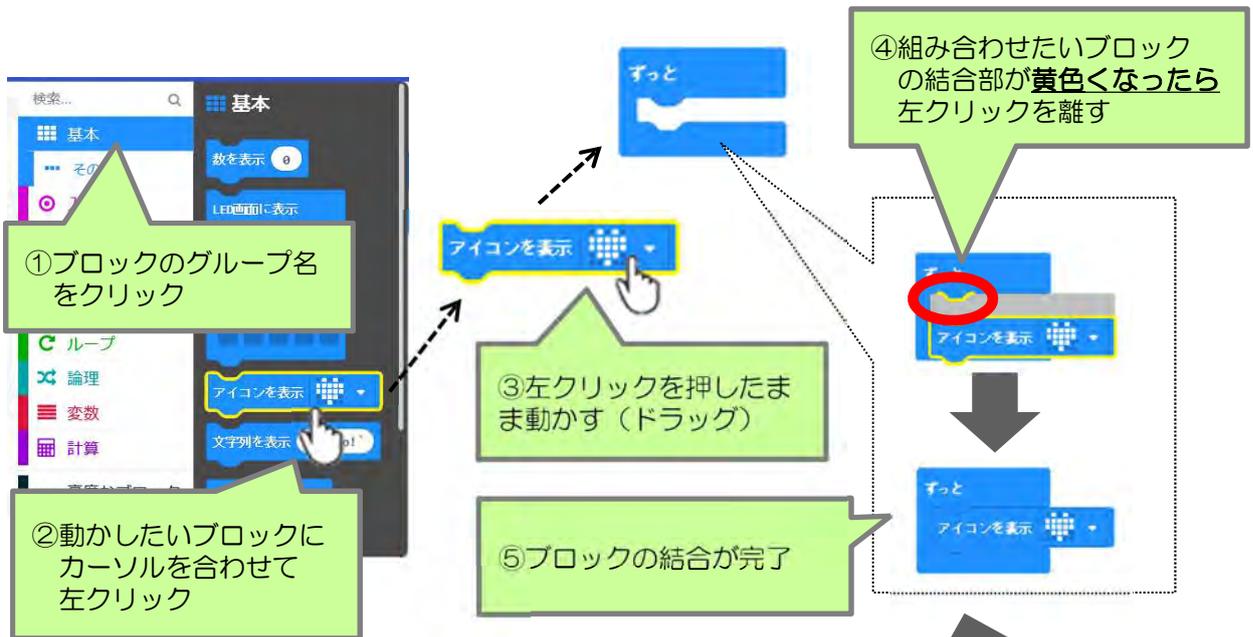
②-4

### 英語表記から日本語表記への変換方法



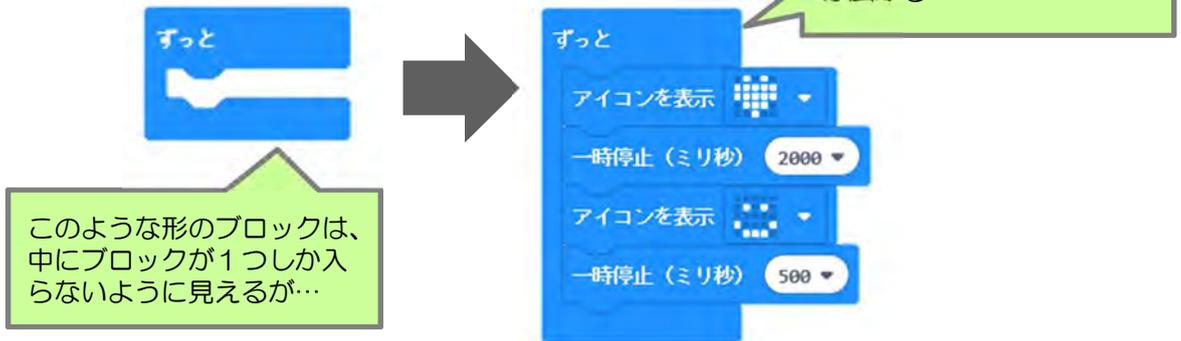
②-5

ブロックの組み合わせ方 (1)



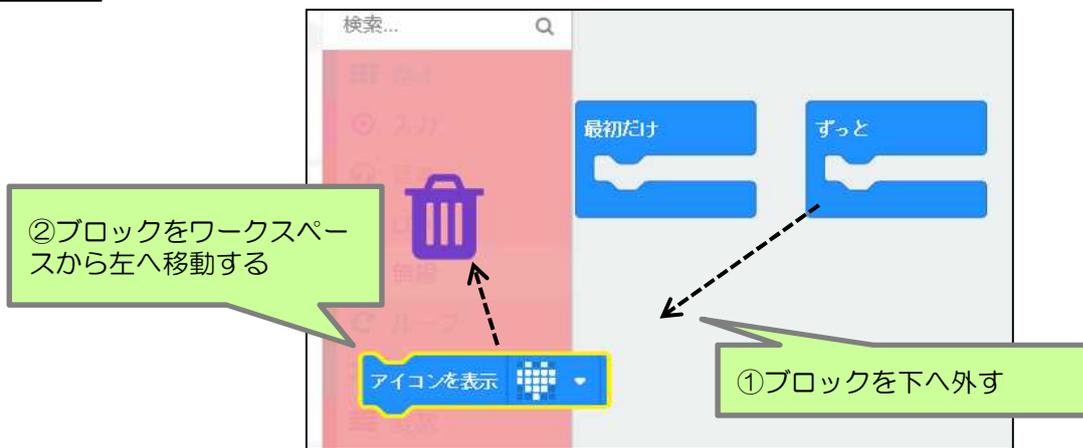
②-6

### ブロックの組み合わせ方 (2)



②-7

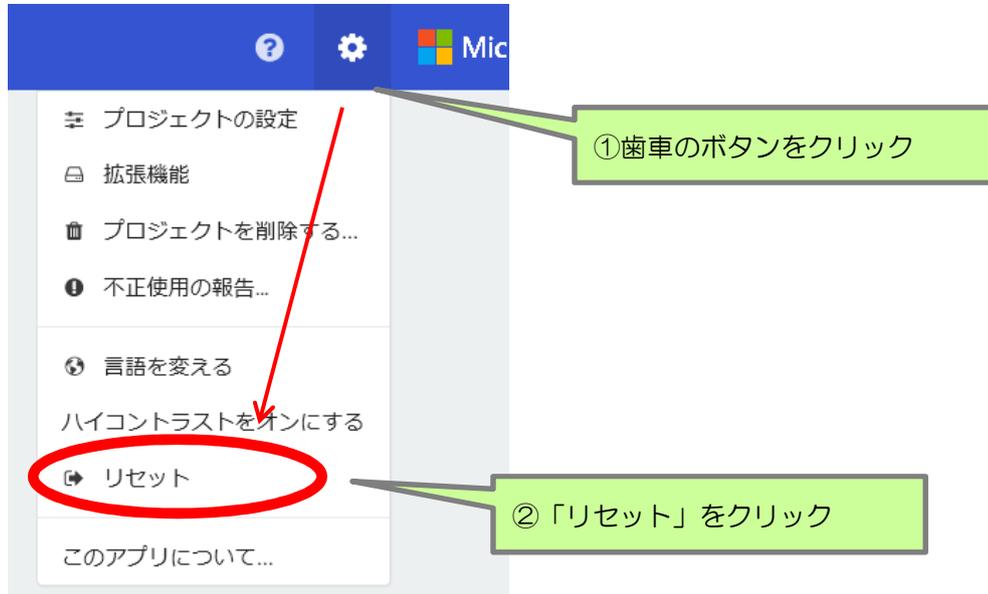
### ブロックの消し方



## ②-8

### 作業環境のリセット

- 同じパソコンで作ったプロジェクト（作業状況）は残っている
- これを削除するには、歯車メニューから「リセット」を実行



## ③ マイクロビット（実機）でのプログラムの実行

### ③-1

#### マイクロビットの接続



① USBケーブルでパソコンとマイクロビットをつなぐ  
※パソコンから電源も供給される

② 「MICROBIT」という名前のドライブ（☒ではD:）が作られる

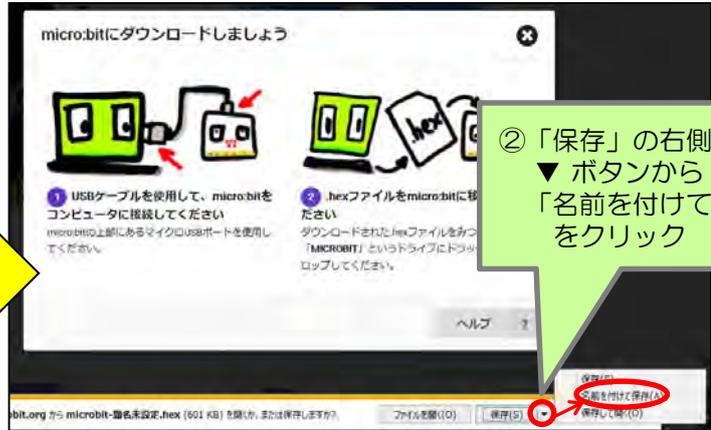


③-2

プログラムのダウンロード



①「ダウンロード」ボタンをクリック



②「保存」の右側の▼ボタンから「名前を付けて保存」をクリック



③MICROBITというドライブを選んで保存

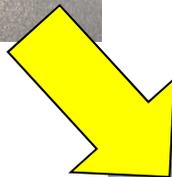
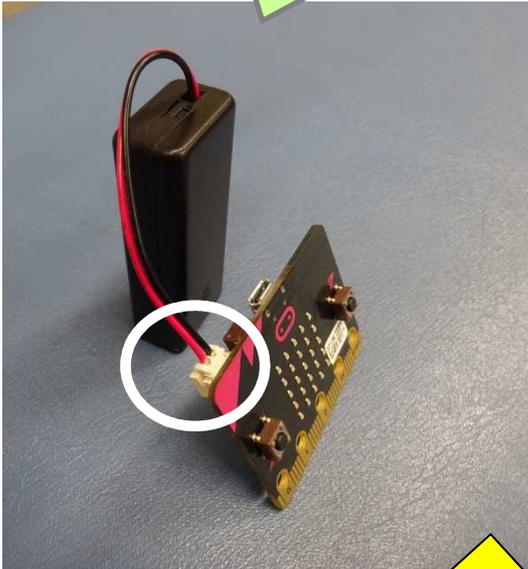
- プログラムの履歴を残したい場合は、一度パソコンに保存 → MICROBITドライブへコピー

③-3

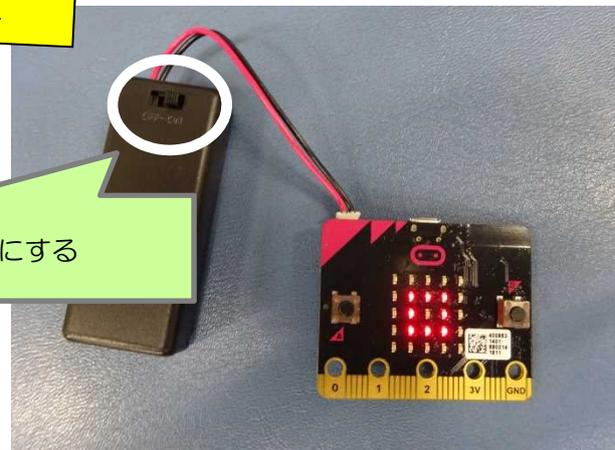
パソコンに接続しない使い方

- 電池ボックスを利用すれば、パソコンに接続しなくてもプログラムを実行できる

①電池ボックスとマイクロビットを接続



②電池ボックスの電源をONにする



## ④ブロックのグループ（主なグループ）

④-1

### 基本グループ



④-2

### 入力グループ（ボタン、センサ）

- ボタンの状況の取得
- 各種センサの値の取得



④-3

### ループグループ (繰り返し)



④-4

### 論理グループ (条件分岐)



## ⑤プログラミングでコンピュータを意図したとおりに動かす

⑤-1

ワークシートの課題： 2) 何もしないときは「ハート」が表示、Aボタンが押されたときだけ自分の名前がローマ字で表示されるようにしよう

### 〈プログラム例〉



⑤-2

ワークシートの課題： 3) 加速度センサーを用いて、ゆさぶられたときに、LED表示が変わるプログラム

### 〈プログラム例〉



LED表示の換え方は、次ページで説明



## 〈LED表示の変え方〉



①「基本」グループのブロックを使用

LEDを表示

手動で変更



ボタン A が押されたとき

LED画面に表示

〈パターン①〉  
点灯させたいLEDをクリックしていく

あらかじめ設定されたアイコンを使用

アイコンを表示

文字列を表示 "Hello!"

文字列で表示変更

〈パターン②〉  
「文字列を表示」ブロックの文字列を変更

〈パターン③〉  
「アイコンを表示」ブロックの右側の ▼ ボタンを押して、変更したいアイコンをクリック

The diagram illustrates the process of changing the LED display on a micro:bit. It starts with the Scratch editor interface, highlighting the 'LED display' block in the 'Basic' group. A callout box explains that this block is used for manual changes. A close-up shows the 'LED display' block with a grid of LEDs, and a callout box indicates that the user should click on the LEDs they want to light up. Another callout box shows the 'LED display' block with a dropdown arrow, and a callout box explains that the user should click on the dropdown arrow to change the icon. A third callout box shows the 'Text display' block with the text 'Hello!', and a callout box explains that the user should change the text in this block. A final callout box shows the 'Text display' block with the text 'Hello!', and a callout box explains that the user should change the text in this block.

⑤-3

ワークシートの課題： 4) 端子「P0」がタッチされたときにLED表示が変わるプログラム

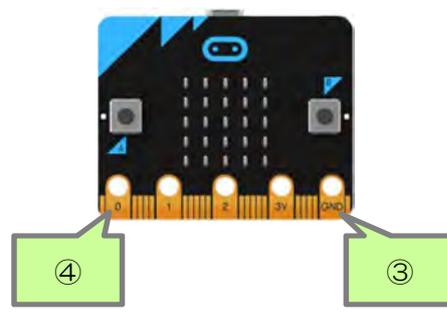
### 〈プログラム例〉



・エディター上で端子P0をクリックすると、「端子P0がタッチされたとき」(プログラム例右側)のプログラムが動く  
・端子の電圧を0 (0Vの意味) から 1023 (3.3Vの意味) までの間の整数として返される ※画面では852

### ●実機でプログラムを実行する場合

- ①プログラムのダウンロード
  - ②電池ボックスの接続
  - ③右手で「GND」端子をつまむ
  - ④左手で「P0」端子をつまみ、左手を離す  
→ LED表示が変化することを確認
- ③-2    ③-3    参照



※マイクロビットの「端子P0がタッチされたとき」ブロックは、タッチする前に片方の手で「GND」端子も触っていないとプログラムは動作しない。  
また、左手で端子P0を触って、端子を離すまでの動きを「タッチ」としているため、タッチしたままだとLED表示は変わらない。

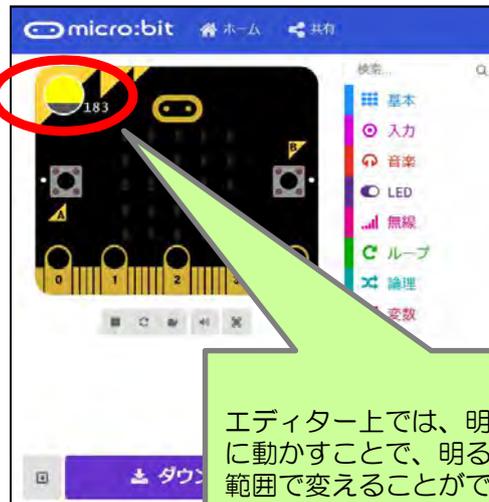
※マイクロビットに保護ケースを装着して素手で触れない場合は、ミノムシリード線を使って素手と同じ要領でタッチする。



⑤-4

ワークシートの課題： 5) 明るさの値の変化を調べる

### 〈プログラム例〉



エディター上では、明るさゲージを上下に動かすことで、明るさを0～255の範囲で変えることができる

#### ● 実機でプログラムを実行する場合

①プログラムのダウンロード

②電池ボックスの接続

③いろいろな場所で明るさの値を調べる

③-2

③-3

参照

- ・ 値が大きいほど明るく、小さいほど暗い
- ・ 次の課題のために、机上で手をかざしたときの明るさも調べておく

⑤-5

ワークシートの課題： 6) 何もしていないときは「小さい四角」が表示されていて、Micro:bitの上で手をかざしたときに「うれしい顔」が表示されるプログラム

### 〈プログラム例〉



条件分岐する明るさ（例では10）は、部屋の明るさなどによって異なるので、手をかざした時の明るさを先に調べておく

明るさが10以下になったら（暗くなったら）うれしい顔を表示

明るさが10以下ではない（明るくなったら）小さい四角を表示

「論理」グループの2つのブロックと、「入力」グループの「明るさ」ブロックを組み合わせる



### 〈 参 考 〉



プログラムが繰り返されるたびに小さい四角を表示

明るさが10以下になったら（暗くなったら）うれしい顔を表示

このプログラムだと、うれしい顔が表示されてもすぐに小さい四角が表示されるため、手をかざすと、うれしい顔と小さい四角が交互に表示される

## 3時間目：計算機を作ろう（Studuino）

### ①使用するソフトウェア

#### Studuino（スタディーノ）



- 「ロボット」モードでは、アーテックロボを動かすためのプログラミングを行う（中学校3年生編で使用）
- 「キャラクター」モードはScratch（スクラッチ）1.4と同等で、パソコン画面上でのプログラミングを行う（中学校1年生編、2年生編で使用）
- 事前にダウンロード・インストールが必要

#### iPad

※iPadの場合はアプリケーション「Tickle（ティックル）App for Studuino」を使用  
【アーテック社 取扱説明書掲載URL】 <https://www.artec-kk.co.jp/studuino/ja/tickle.php>

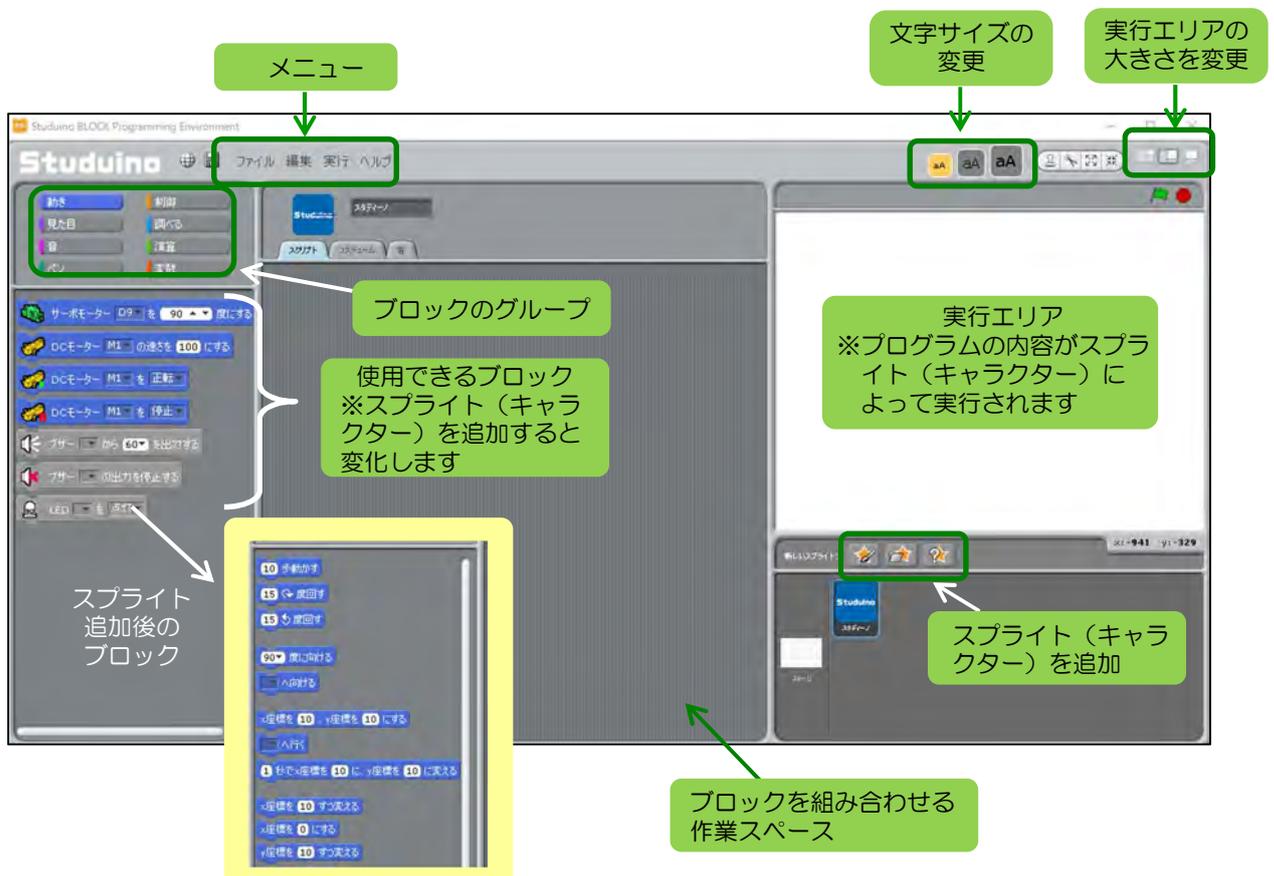
※ただし、「キャラクター」モードは無いため、他のソフトウェアでの対応が必要（「ロボット」モードのみ可）

### ②ソフトウェアの操作方法

②-1

#### Studuino（スタディーノ）の操作画面（キャラクターモード）

※iPadの場合はScratch 3.0 (<https://scratch.mit.edu/>) などで対応



②-2

# スプライトの追加方法

※スプライト=命令を与えるキャラクター



例：roundman

- あらかじめ用意されているスプライトを使用する例

①デスクトップにある「スタディーノ」のアイコンをダブルクリックし、「ブロックプログラミング環境」→「キャラクター」を選択

②右側にある実行エリア下の「新しいスプライトをファイルから選ぶ」をクリック

③任意のフォルダをダブルクリック  
※画面は「People」

④キャラクターを選択し、「OK」をクリック  
※画面は「roundman」

⑤スプライトを追加すると使用できるブロックが変わる

### ③演算子を用いたプログラム

③-1

スプライトを追加

※スプライト=命令を与えるキャラクター

②-2

参照

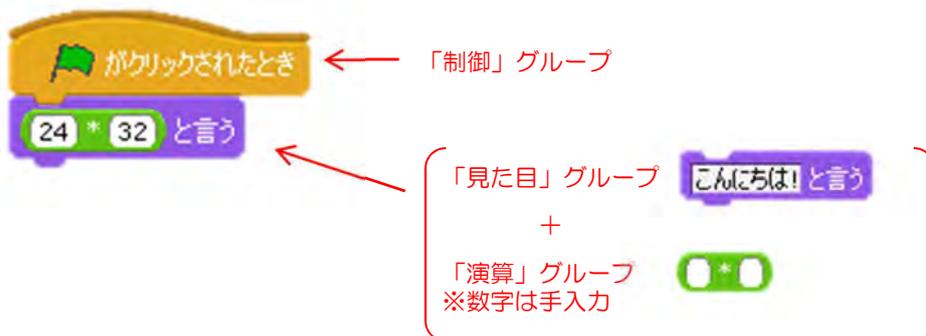


例：roundman

③-2

プログラムの作成

#### 〈プログラム例〉



#### 〈実行エリア〉

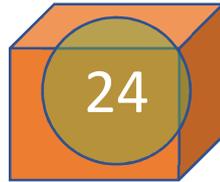


## ④演算子に変数を組み合わせたプログラム

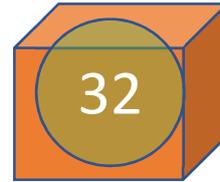
### ④-1 変数とは

- 「変数」とは、プログラムの実行にともなって変化していく情報（数値や文字）を格納しておく箱のことです。
- それぞれの箱に名前をつけることができます。

〈イメージ〉

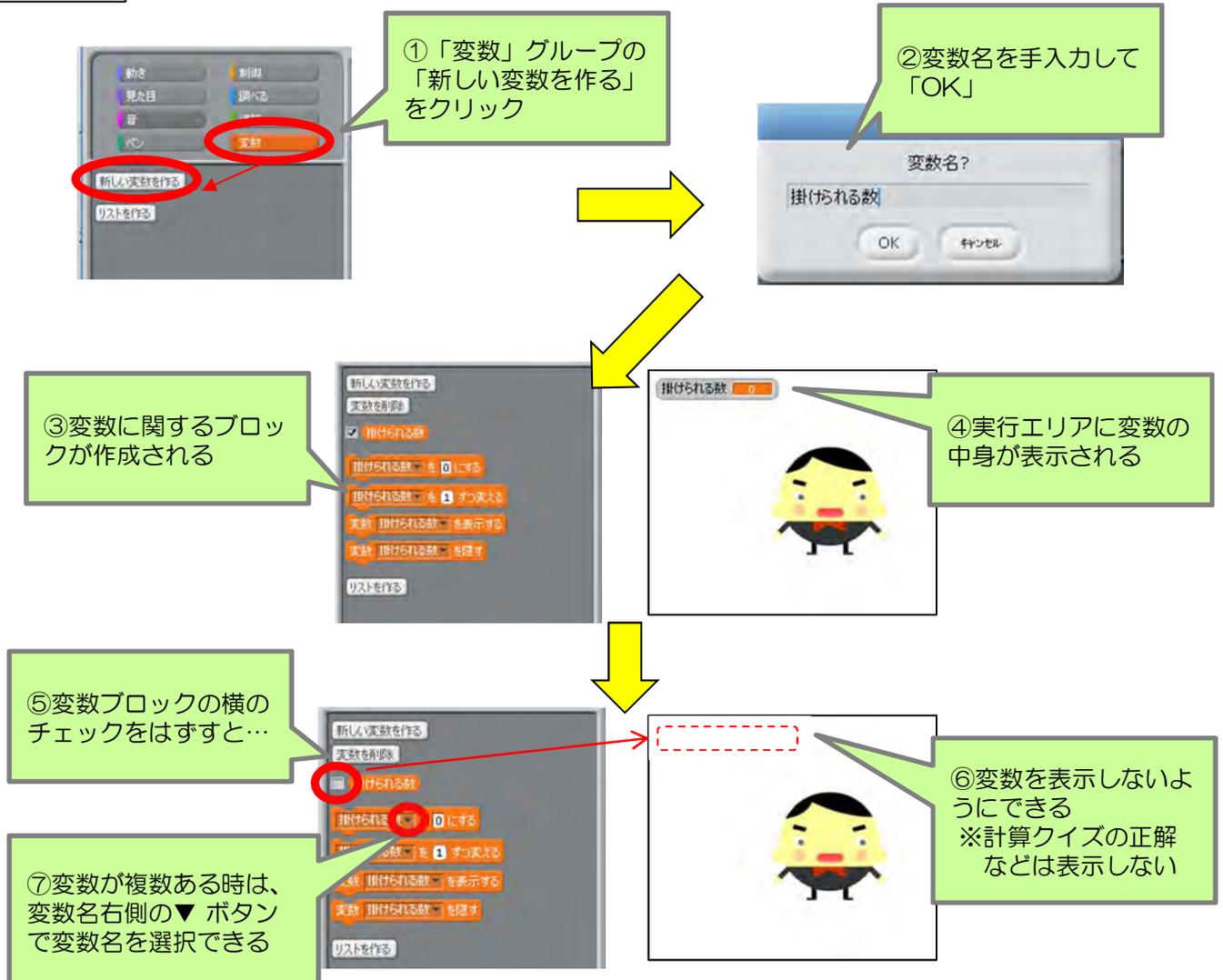


【変数】掛けられる数



【変数】掛ける数

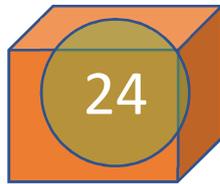
### ④-2 変数の作り方



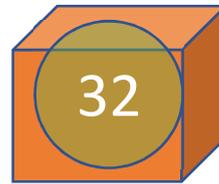
④-3

変数の有用性

・箱の中身を変えるだけで、関係する複数の動作をすべて変化させることができます



【変数】 X



【変数】 Y

名前がXとYの変数



変数を直接入力（計算したい任意の数字）

「X+Y」の計算結果を2秒表示（この場合56）

「X-Y」の計算結果を2秒表示（この場合-8）

「X×Y」の計算結果を2秒表示（この場合768）



●変数（X、Y）の中身だけを入れ替えれば色々な計算ができる

④-4

変数を用いたプログラム

〈プログラム例〉



〈変数〉

掛けられる数

掛ける数

変数「掛けられる数」と「掛ける数」を作成

④-2 参照

「掛けられる数」と「掛ける数」の計算結果を表示（この場合946）

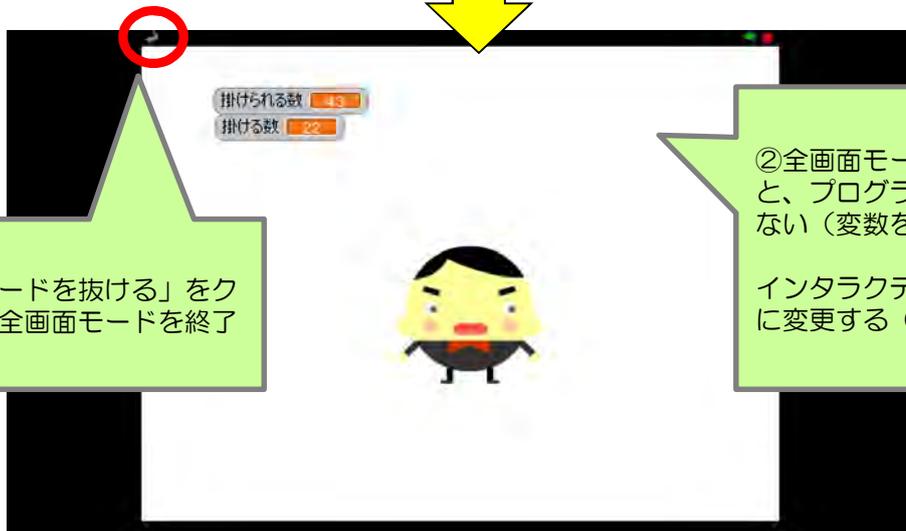
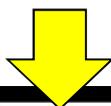


④-5

## 全画面表示



①「発表モードに切り替える」をクリック



③「発表モードを抜ける」をクリックして全画面モードを終了

②全画面モードにして実行すると、プログラムを書き換えられない（変数を変更できない）

↓  
インタラクティブなプログラムに変更する（⑤以降）

## ⑤インタラクション（相互作用）

⑤-1

### 概要（イメージ）

- 実行時にユーザからの情報入力を受け付け、その情報によってプログラムの結果を変えるしくみ

- ① ← 「調べる」グループの、このブロックを使うと入力欄が表示され、好きな文章などが入力できる。
- ②

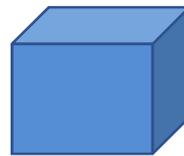


- ① 「...と聞いて待つ」ブロックは、「...」に入力した文字列を吹き出しに表示する。

あなたの名前は何ですか? と聞いて待つ

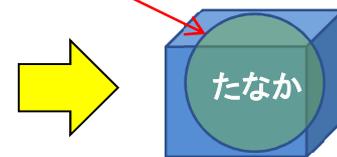
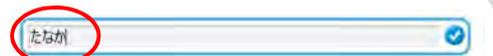


- ② 「答え」ブロックは、**入力欄に入力された文字や値が入る、初めから用意されている変数のブロック。**



〈答え〉

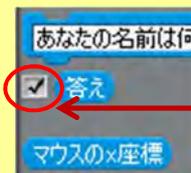
最初は空っぽ



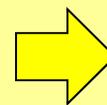
〈答え〉

入力欄と同じ文字や値になる

- ソフトウェア上で「答え」ブロックの中身を見る方法



「調べる」グループのブロック一覧にある「答え」ブロック左のチェックを入れると、実行エリアに表示される



〈例〉



← 「調べる」グループ ※四角内の値を吹き出しに表示

← 「こんにちは! と言う」ブロック

「見た目」グループ 

+

「演算」グループ 

+

「調べる」グループ

「たなか」と入力すると...

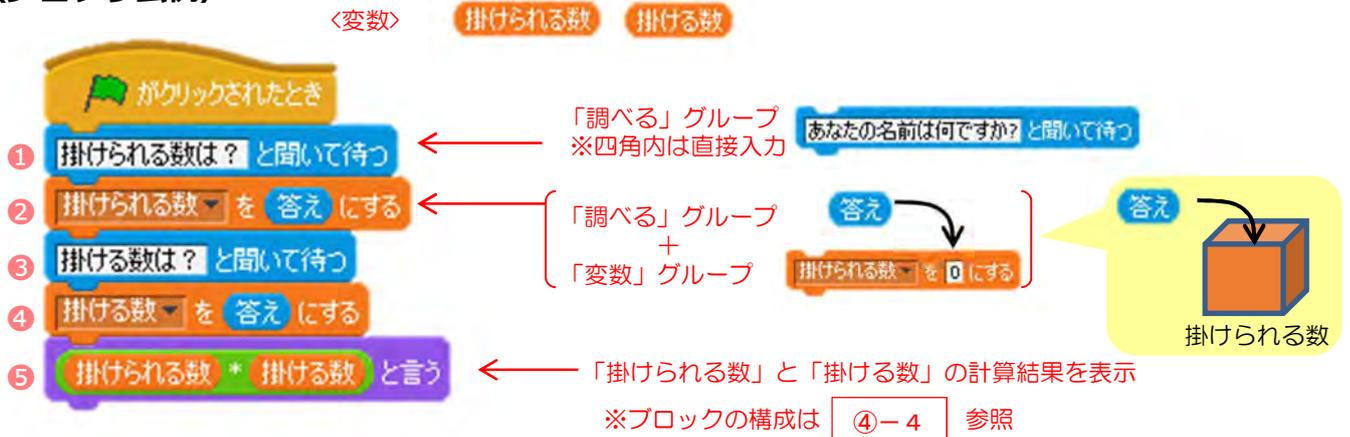


⑤-2

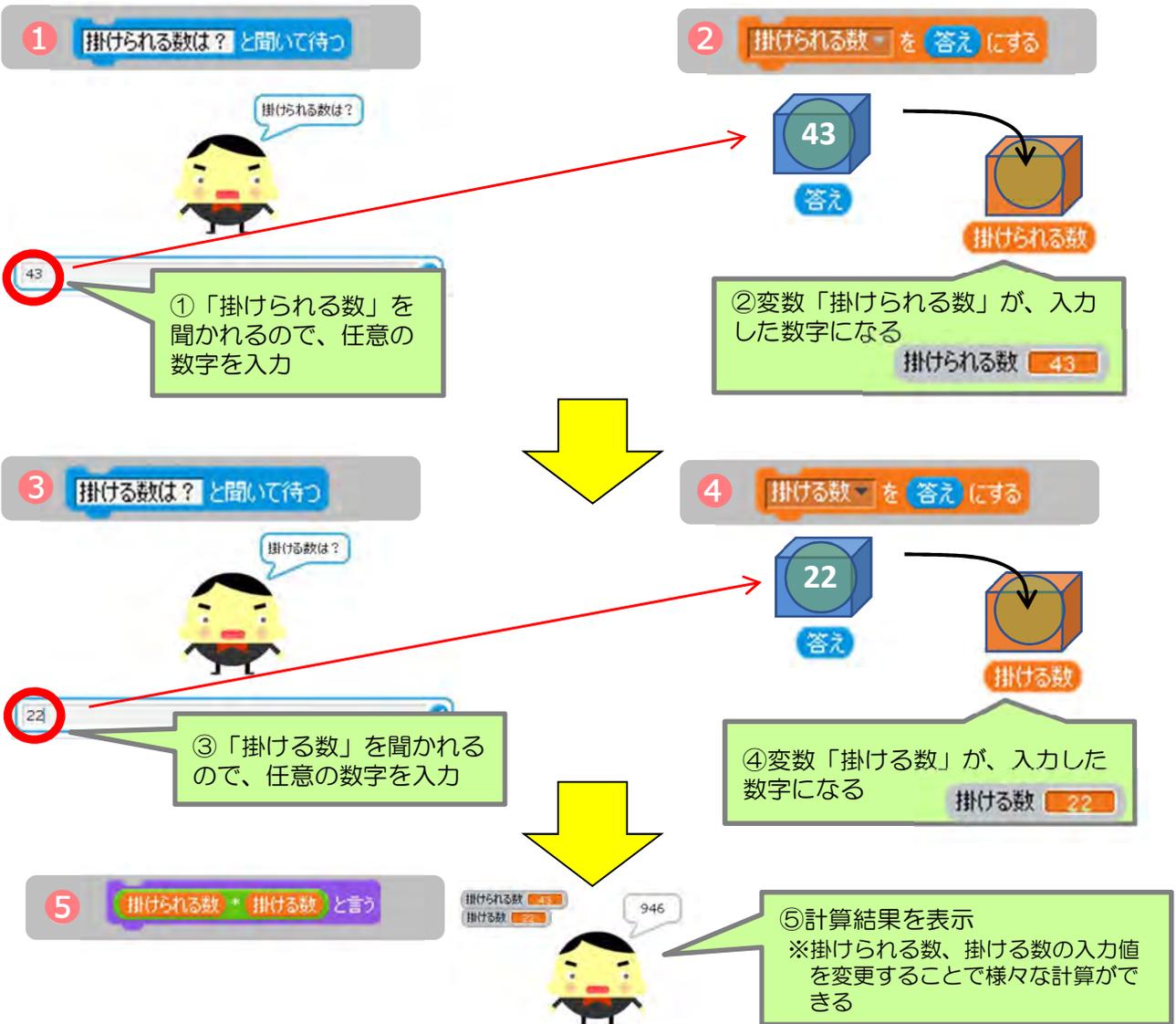
インタラクティブなプログラム

- ④-4 で作成したプログラムを変更していく

〈プログラム例〉



〈プログラムの実行〉



⑤-3

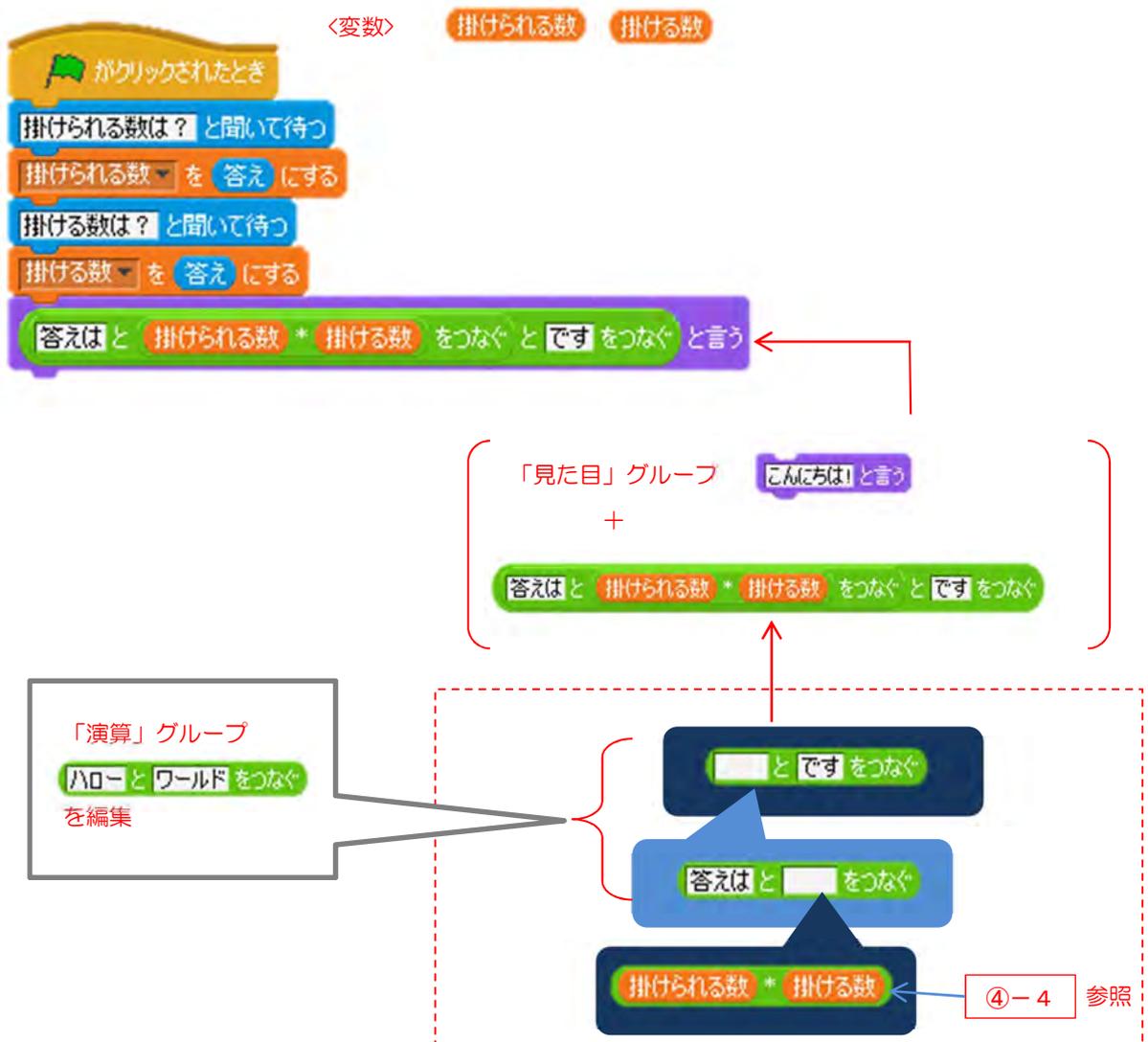
全画面表示

④-5 参照



⑤-4

文字列連結ブロック (「答えは～です」と表示させる)



## ⑥他の計算（足し算や引き算）をするプログラムの作成

⑥-1

スプライトを追加

※スプライト=命令を与えるキャラクター

②-2

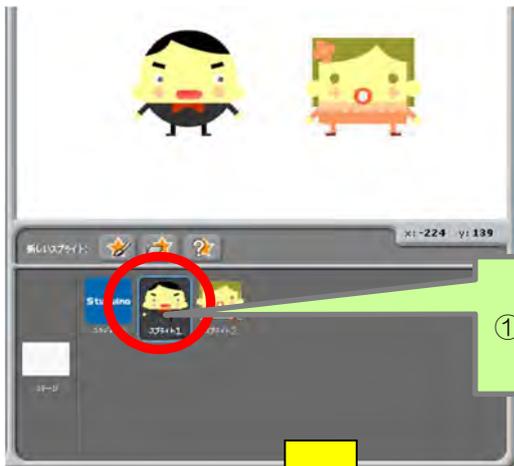
参照



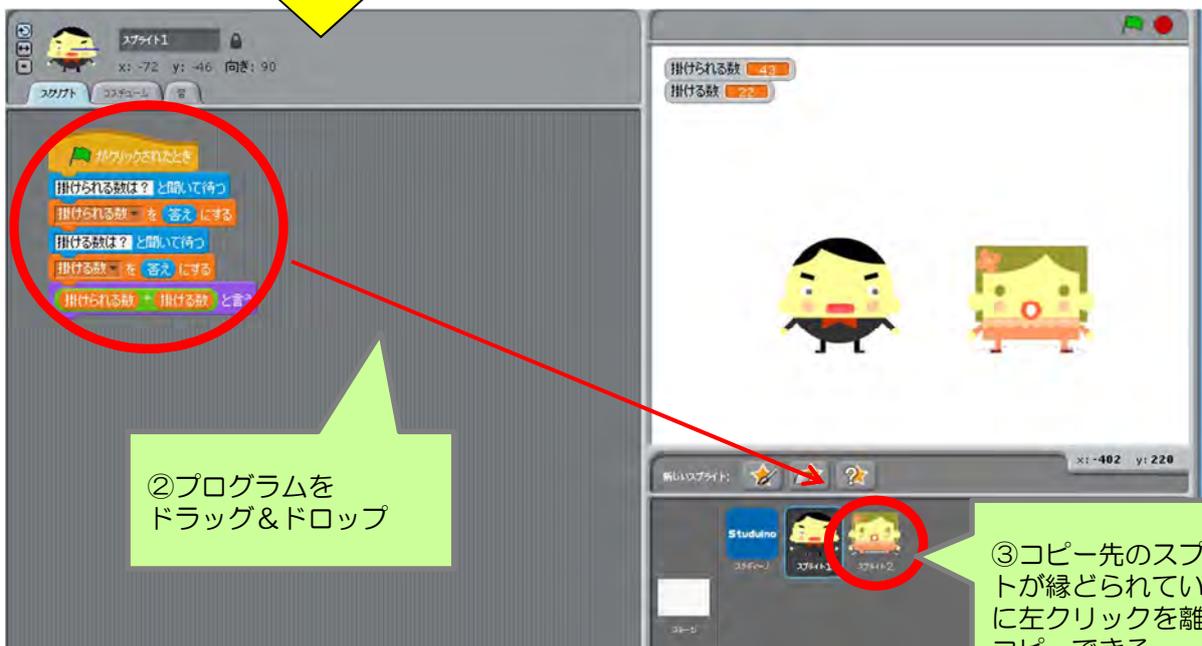
追加したスプライト  
(squaregirl)

⑥-2

プログラムをスプライト間でコピー



①コピー元のスプライトを選択



②プログラムを  
ドラッグ&ドロップ

③コピー先のスプライトが縁どられている時に左クリックを離すとコピーできる

⑥-3

割り算のプログラム例

①「Sprite 2」を選択

Sprite 1 からコピーされたプログラムがあるので、これを編集していく

- Sprite 2 がクリックされたとき
- 掛けられる数は? と聞いて待つ
- 掛けられる数 を 答え にする
- 掛ける数は? と聞いて待つ
- 掛ける数 を 答え にする
- 掛けられる数 \* 掛ける数 と言う

②割り算に用いる変数「割られる数」「割る数」を作る

④-2 参照

<変数> 割られる数 割る数

- Sprite 2 がクリックされたとき ← 「制御」グループのブロックを変更 ※プログラム開始の合図をSprite 1 と別にしておく
- 割られる数は? と聞いて待つ ← 直接入力により質問を変更
- 割られる数 を 答え にする ← 変数名右側の▼ボタンから変数を「割られる数」に変更
- 割る数は? と聞いて待つ ← 直接入力により質問を変更
- 割る数 を 答え にする ← 変数名右側の▼ボタンから変数を「割る数」に変更
- 割られる数 / 割る数 と言う ← 演算を変更（割り算は「/」で表す）

Sprite 2 をクリックするとプログラムがスタート

## 4時間目：計算クイズを作ろう (Studuino)

### ①乱数の使い方 (指導過程 1)

①-1

Studuino (スタディーノ) を起動し、スプライトを追加

※スプライト=命令を与えるキャラクター

中学校1年生編の3時間目 ②-2 参照



例：roundman

①-2

乱数の使い方



### ②乱数を用いた演算 (指導過程 2)

②-1

変数を乱数にする

掛けられる数 を 1 から 10 までの乱数 にする

②ブロックをクリックすると  
変数が変化することを確認

掛けられる数 10

「変数」グループ 掛けられる数 を 0 にする  
+  
「演算」グループ 1 から 10 までの乱数

①まずは変数を作る  
※中学校1年生編の3時間目

④-2 参照

②-2

乱数を用いた計算式

②-1 参照

「掛られる数」と「掛ける数」の計算結果を変数「正解」に代入

①まずは変数「掛ける数」と「正解」を追加  
 ※中学校1年生編の3時間目  
 ④-2 参照

②ブロックをクリックすると変数が変化することを確認

掛られる数 3  
 掛ける数 7  
 正解 21

「変数」グループ 掛られる数 を 0 にする  
 +  
 変数名右側の▼ボタンから変数「正解」を選択

「演算」グループ + \* -  
 +  
 「変数」グループ 掛られる数 掛ける数

③キャラクターにクイズを出させる (指導過程3)

掛られる数 と × と 掛ける数 と の答えは? をつなぐ をつなぐ をつなぐ と聞いて待つ

「調べる」グループ あなたの名前は何ですか? と聞いて待つ  
 +  
 「演算」グループ ハローとワールドをつなぐ  
 ↑ 3つ組み合わせる  
 ※「×」と「の答えは?」は手入力  
 掛られる数 と をつなぐ  
 × と をつなぐ  
 掛ける数 と の答えは? をつなぐ

+  
 「変数」グループ 掛られる数 掛ける数



#### ④ 正解と不正解の判別（指導過程 4）



- ・「入力欄に入力された値」（**答え**）が「変数『正解』」と等しいかどうかで条件分岐  
※③の場合は変数「正解」は21
- ・等しければ「正解!」と言い、等しくなければ「間違い!」と言う

**答え** 「答え」ブロックは、入力欄に入力された文字や値が入る、初めから用意されている変数のブロックです。



「制御」グループ



「演算」グループ



+

「調べる」グループ



+

「変数」グループ



「見た目」グループ



※手入力に変更

## ⑤計算クイズを作る（指導過程5）

⑤-1

### 1問だけの計算クイズ

- ②③④で作成したプログラムを結合する

〈変数〉 掛けられる数 掛ける数 正解

「制御」グループのブロックで開始の合図を指定

②で作成したプログラム  
※まずは変数を決める

③で作成したプログラム  
※変数が決まったら質問させる

④で作成したプログラム  
※答えの入力と判定

ただし、正解が見えてしまっている

4x3の答えは？

⑤-2

### クイズの改良

- 正解を隠す

「変数」グループを表示し、変数「正解」の横のチェックを外す

⑤-3

5問出題し、最後に正解数を表示させるようなプログラム

- ⑤-1、⑤-2で作成したプログラムを改良する

まずは変数「正解数」を作る **正解数**  
 ※中学校1年生編の3時間目  
 ④-2 参照

<変数> 掛けられる数 掛ける数 正解 正解数

がクリックされたとき

正解数 を 0 にする

5 回繰り返す

掛けられる数 を 1 から 10 までの乱数 にする

掛ける数 を 1 から 10 までの乱数 にする

正解 を 掛けられる数 \* 掛ける数 にする

掛けられる数 と × と 掛ける数 と の答えは? をつなぐ をつなぐ をつなぐ と聞いて待つ

もし 答え = 正解 なら

正解! と 2 秒言う

正解数 を 1 ずつ変える

でなければ

間違い! と 2 秒言う

正解数は と 正解数 をつなぐ と 2 秒言う

5問出題し終わったら正解数を言う

「見た目」グループ こんにちは! と 2 秒言う  
 +  
 「演算」グループ ハローとワールドをつなぐ  
 ↑ 「正解数は」は手入力  
 +  
 「変数」グループ 正解数

「変数」グループのブロックを追加  
 ※最初は「正解数」はゼロ。正解する度に増えていく。

「制御」グループの繰り返しブロックを追加し5問出題させる

「変数」グループのブロックを追加  
 ※正解する度に変数「正解数」を1ずつ増やす

掛けられる数 5 } ← 出題の度に乱数により変化  
 掛ける数 10 }  
 正解数 0 ← 正解する度に1ずつ増える

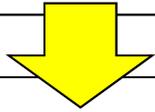


● プログラム改良の手順

・授業中のヒントとして、生徒に示しても良い（習熟度に合わせて適宜修正）

【はじめ】

- ・⑤-1、⑤-2で作成したプログラムをベースにする

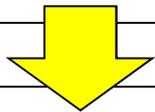


【STEP1】

- ・5問繰り返すプログラムを作る



「制御」グループのブロックを追加

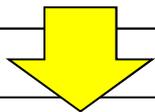


【STEP2】

- ・「正解数」という変数を作る



「変数」グループ

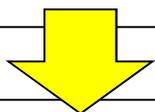


【STEP3】

- ・「正解数」は、はじめはゼロ



- ・「変数」グループのブロックを追加
- ※変数名右側の▼ボタンから変数を「正解数」に変更

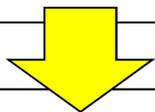


【STEP4】

- ・正解することにより、「正解数」を増やす



- ・「変数」グループのブロックを追加
- ※変数名右側の▼ボタンから変数を「正解数」に変更



【STEP5】

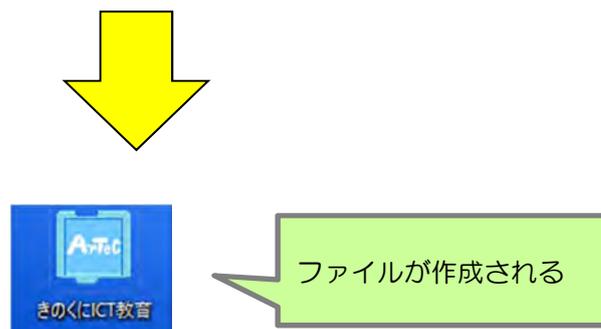
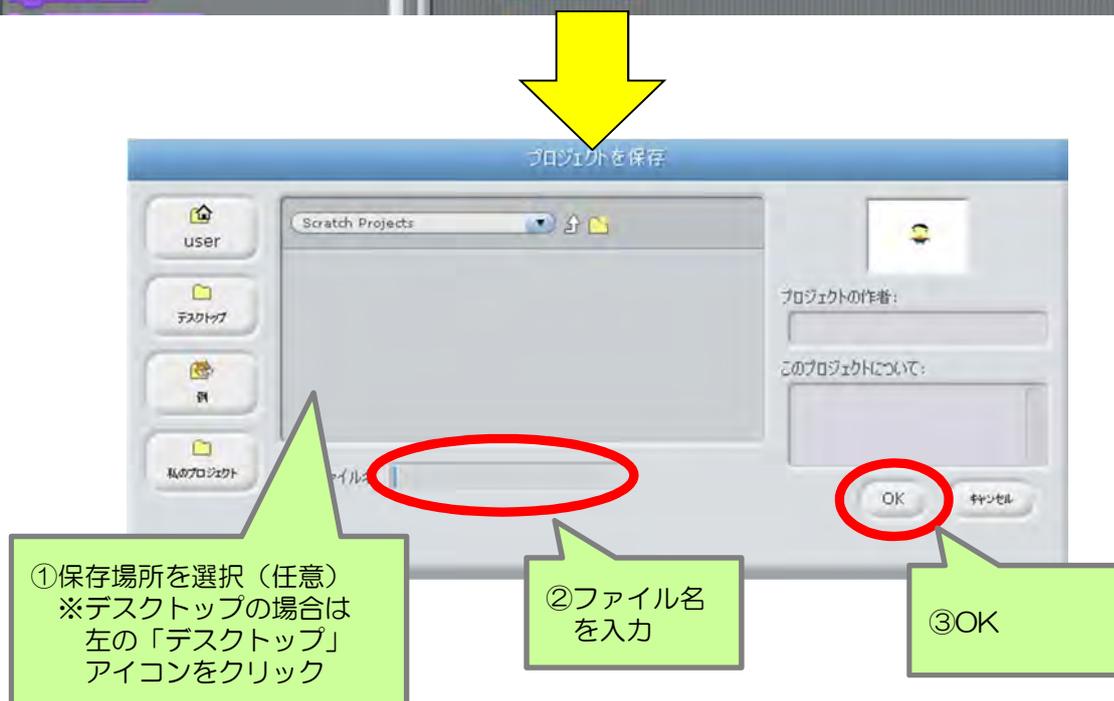
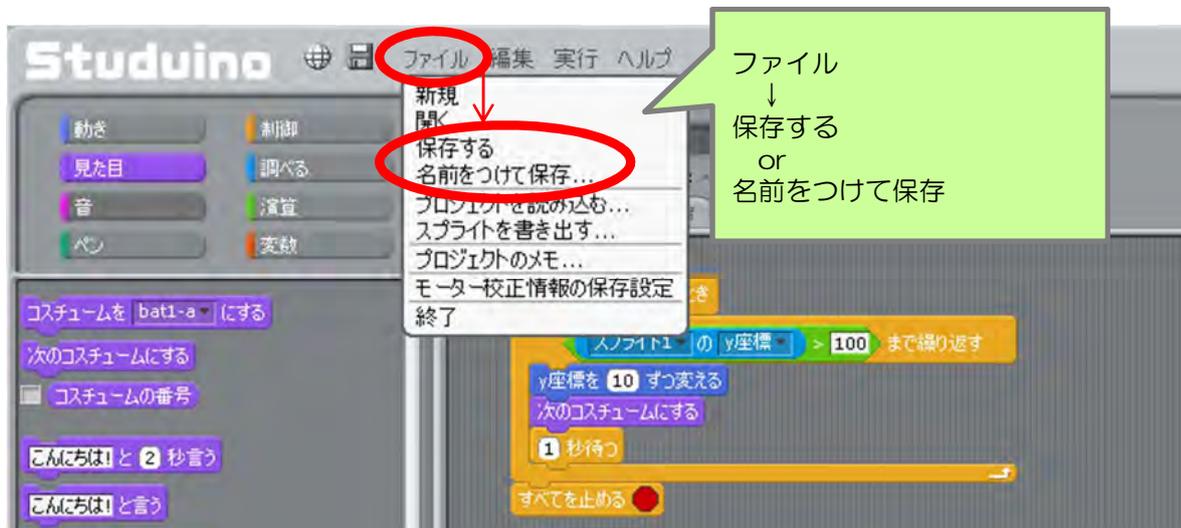
- ・5問終わったら正解数を言う

正解数はと 正解数をつなぐ と 2秒言う

「見た目」グループ **こんにちはと 2秒言う**  
 +  
 「演算」グループ **ハローとワールドをつなぐ**  
 +  
 「変数」グループ **正解数**  
 ↑「正解数は」は手入力

## ⑥プログラムの保存

- プログラムを保存しておき次回以降の授業の際に読み込むことで、保存時のプログラムを使用することができる



# 5時間目：分かりやすく・使いやすくしよう（Studuino）

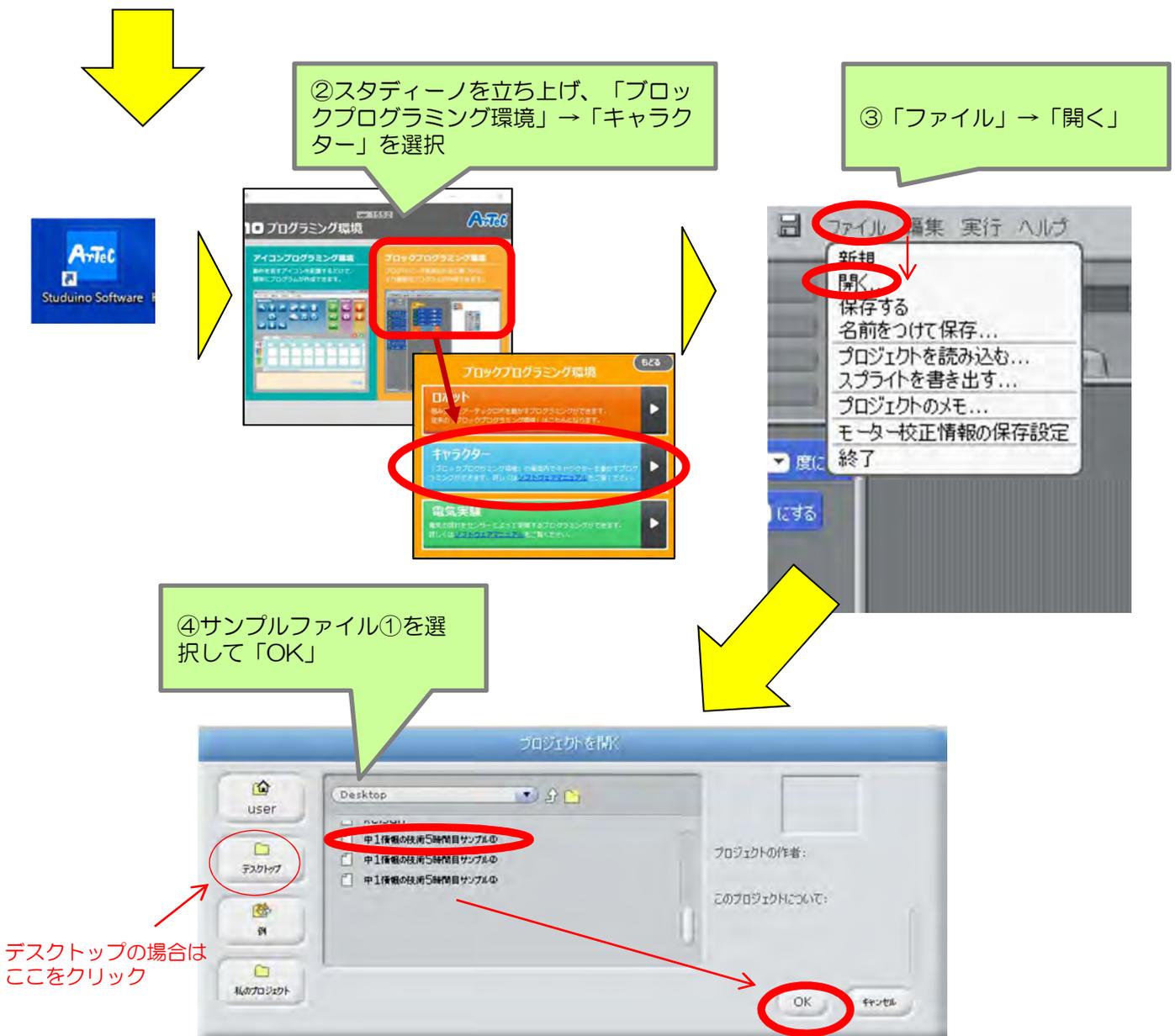
## ① サンプルプログラムの実行（指導過程 1）

### ①-1 サンプルプログラム①の読み込み

- 中1情報の技術5時間目サンプル①
- 中1情報の技術5時間目サンプル②
- 中1情報の技術5時間目サンプル③

① サンプルプログラムをパソコンに保存しておく（保存場所は任意）

サンプルプログラムは、教育センター学びの丘の「きのくにeラーニング」システムの「きのくにICT教育（中学校）」に掲載



①-2

サンプルプログラム①

- 実行すると計算クイズが出される
  - 正解するとRoundman（人間のキャラクター）は右へ移動
  - コウモリが近づいてきて、触れられたらゲームオーバー



● 実行して、これまで作成してきたプログラムとの動作の違いを見つけさせる

<スプライト1のプログラム>



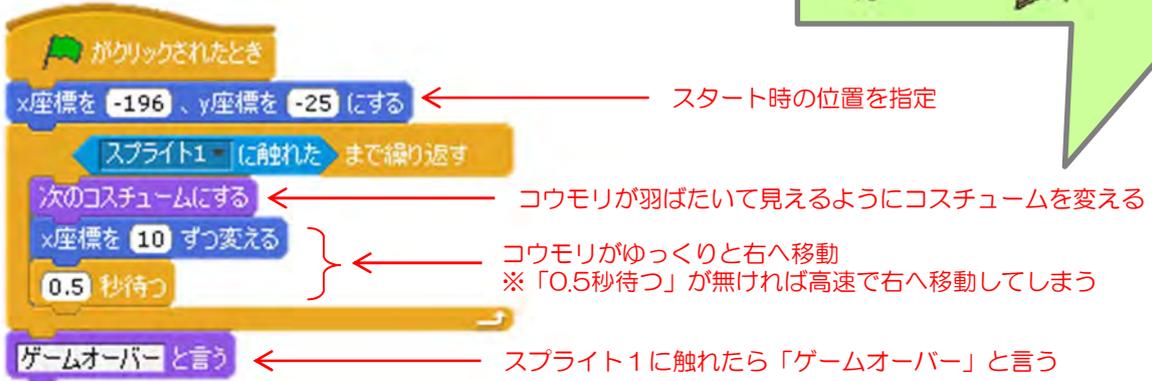
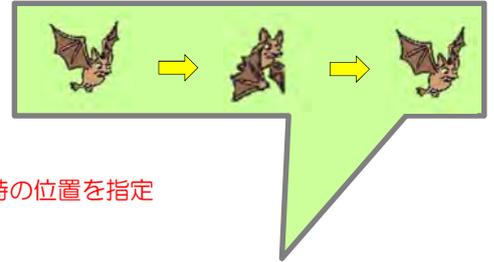
<変数> 掛けられる数 掛ける数 正解 正解数

```

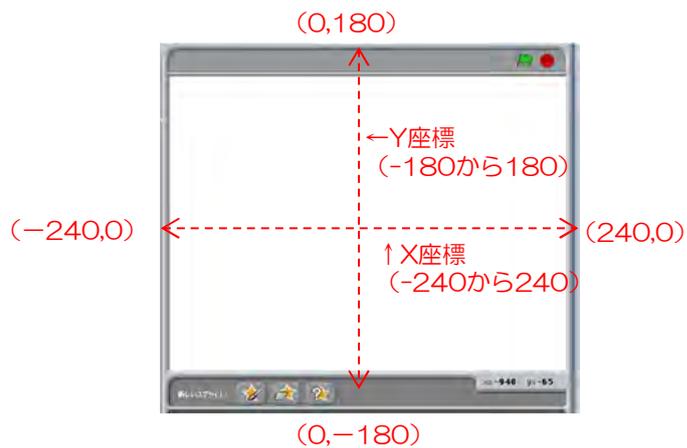
    がクリックされたとき
    正解数を 0 にする
    x座標を 0、y座標を 0 にする ← スタート時の位置を指定
    スプライト2に触れたまで繰り返す ← コウモリに触れられるまで繰り返し
    もし 200 < スプライト1の x座標 なら
        おめでとうと 2秒言う
        すべてを止める
    でなければ
        掛ける数を 1から10までの乱数にする
        掛けられる数を 1から10までの乱数にする
        正解を 掛けられる数 * 掛ける数 にする
        掛けられる数 と x と 掛ける数 をつなぐをつなぐ と の答えは？ をつなぐ と聞いて待つ
        もし 答え = 正解 なら
            正解!と 1秒言う
            正解数を 1ずつ変える
            x座標を 50ずつ変える ← 正解するとスプライト1が少し右へ動く
        でなければ
            間違い!と 2秒言う
    正解数と 正解数 をつなぐ と 2秒言う
    
```

4時間目に作った計算クイズのプログラムがベース

## 〈スプライト2のプログラム〉



## 〈実行エリアの座標〉



## ②制限時間をわかりやすくする工夫（指導過程2）

### ②-1 サンプルプログラム②の読み込み

※ ①-1 参照

### ②-2 サンプルプログラム②

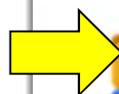
- 実行するとコウモリが時間とともにゆっくり上に移動  
→ 上に着いたら（y座標が100になったら終わり）



- プログラムの数値などを生徒に変更させ、移動距離や速さの設定が難易度を意味することに気付かせる

### 〈参 考〉

- 上のプログラムのままだと、コウモリを下に戻すにはマウスでドラッグするしかないが・・・



毎回低い位置からスタート

### ③正解不正解のわかりやすさの課題解決（指導過程3）

#### ③-1 サンプルプログラム③の読み込み

※ ①-1 参照

※生徒の状況に合わせて、サンプルプログラム①や、4時間目に保存したプログラムを読み込んで良い

#### ③-2 サンプルプログラム③

- 4時間目に作った計算クイズのプログラムにサンプルプログラム②（制限時間のプログラム）を加えたプログラム

<変数> 掛けられる数 掛ける数 正解 正解数

```
when clicked  
while (sprite 1's y-coordinate > 100) loop  
  change y-coordinate by 10  
  change to next costume  
  wait 1 seconds  
stop everything
```

新たに追加された制限時間を表すプログラム

```
when clicked  
set correct number to 0  
loop 5 times  
  set number to be multiplied to random number from 1 to 10  
  set multiplier to random number from 1 to 10  
  set correct answer to (number to be multiplied * multiplier)  
  ask (number to be multiplied * multiplier) what is the answer? and wait  
  if (answer = correct answer) then  
    say Correct! for 2 seconds  
    increment correct number by 1  
  else  
    say Wrong! for 2 seconds  
say Correct number is correct number for 2 seconds
```

4時間目に作った計算クイズのプログラム

## 〈例 1 : 正解・不正解の音を追加〉

変数: 掛けられる数, 掛ける数, 正解, 正解数

がクリックされたとき

正解数を 0 にする

5 回繰り返す

掛けられる数を 1 から 10 までの乱数 にする

掛ける数を 0 から 10 までの乱数 にする

正解を 掛けられる数 \* 掛ける数 にする

掛けられる数 と \* と 掛ける数 と の答えは \* をつなぐ をつなぐ をつなぐ と聞いて待つ

もし 答え = 正解 から

正解 と 2 秒言う ← 正解時の音を出すブロックを追加する箇所

正解数を 1 ずつ変える

でなければ

間違い! と 2 秒言う ← 不正解時の音を出すブロックを追加する箇所

正解数 と 正解数 をつなぐ と 2 秒言う

4時間目に作った  
計算クイズのプログラム

## ● 「音」グループのブロック

の音を鳴らす

終わるまで の音を鳴らす

すべての音を止める

48 のドラムを 0.2 拍鳴らす

0.2 拍休む

60 の音符を 0.5 拍鳴らす

楽器を 1 にする

音量を -10 ずつ変える

音量を 100 % にする

音量

テンポを 20 ずつ変える

テンポを 60 BPM にする

テンポ

●音の換え方

①「音」タブをクリック

②「読み込み」をクリックし、音を選択して「OK」

③「...の音を鳴らす」ブロックの▼ボタンから音を選択

〈例2：スプライトの大きさを変更〉

「見た目」グループのブロックに変更

「見た目」グループのブロックに変更

## ④利用者がもっとやりたいと思えるような工夫をしよう（ワークシート解答例）

④-1

ステージクリア方式（10問クリアしたら、ステージ1クリア）

### ●設定

- ① 1ステージごとに15問出題し、10問以上正解なら次のステージへ
- ② ステージは5まで

### ●4時間目に作成した計算クイズのプログラムにブロックを追加

まずは変数「ステージ」を作る ※中学校1年生編の3時間目  
④-2 参照

このScratchスクリプトは、クイズの進行を管理するためのプログラムです。変数として「掛けられる数」、「掛ける数」、「正解」、「正解数」、「ステージ」が定義されています。

**初期設定:** クリックされたときに「ステージ」を1にする、「正解数」を0にする。

**ループ:** 「ステージ」が5を超えない限り繰り返す。1ステージごとに15問出題する。

**問題生成:** 1から10までの乱数で「掛けられる数」と「掛ける数」を設定し、それらの積を「正解」として設定する。

**質問と回答:** 「掛けられる数」と「掛ける数」を掛け合わせた結果を問う。回答が正解の場合、「正解!」と2秒待つ。正解数を1ずつ増やす。そうでなければ「間違い!」と2秒待つ。

**進捗確認:** 「正解数」が9を超えたら、次のステージに進む（「ステージ」と「クリア」を通知し、2秒待つ）。ステージを1ずつ変え、正解数を0にする。

**終了条件:** 「ステージ」が5を超えたら、「ステージは」と「ステージ」を通知し、2秒待つ。「正解数は」と「正解数」を通知し、2秒待つ。すべてを止める。全ステージクリア!と2秒待つ。

**注釈:**

- ステージは1からスタート
- ステージが5を超えるまで繰り返す（設定②） ※ステージ6になった時点で繰り返しが終了し、出題もされない
- 1ステージごとに15問出題（設定①）
- 15問出題の後、正解数が9を超えているか（10以上か）で条件分岐
- 「制御」グループ + 「演算」グループ + 「変数」グループ
- 10問以上正解でステージを1増やし、正解数を0に戻す。 ※0に戻さなければ正解数はどんどん増えていくので、ステージごとの正解数が分からなくなる
- 正解9問以下で、ステージ数と正解数を告げて終了
- ステージ5まで出題を繰り返した後、正解数が10問以上であれば「全ステージクリア!」と告げて終了

● ④-1 ステージクリア方式の別解

● 設定

- ① 1 ステージごとに15問出題し、10問以上正解なら次のステージへ  
ただし、1ステージで6問間違えた時点で終了
- ② ステージは5まで

赤字部分の設定を追加

〈変数〉 掛けられる数 掛ける数 正解  
 正解数 ステージ 間違い数

まずは変数「間違い数」を作る  
 ※中学校1年生編の3時間目

④-2 参照

The Scratch script is as follows:

```

    旗がクリックされたとき
    ステージ を 1 にする
    正解数 を 0 にする
    間違い数 を 0 にする
    ステージ > 5 まで繰り返す
    15 回繰り返す
    掛けられる数 を 1 から 10 までの乱数 にする
    掛ける数 を 1 から 10 までの乱数 にする
    正解 を 掛けられる数 * 掛ける数 にする
    掛けられる数 と * と 掛ける数 と は? をつなぐ をつなぐ をつなぐ と聞いて待つ
    もし 答え = 正解 なら
        正解! と 2 秒言う
        正解数 を 1 ずつ変える
    できれば
        間違い! と 2 秒言う
        間違い数 を 1 ずつ変える
        もし 間違い数 = 6 なら
            クリア失敗! と 2 秒言う
            ステージは と ステージ をつなぐ と 2 秒言う
            正解数は と 正解数 をつなぐ と 2 秒言う
            すべてを止める
        繰り返す
    ステージ と ステージ と クリア をつなぐ をつなぐ と 2 秒言う
    ステージ を 1 ずつ変える
    正解数 を 0 にする
    間違い数 を 0 にする
    全ステージクリア! と 2 秒言う
    もし 正解数 > 9 なら
    
```

Annotations:

- Red circle around "間違い数" in the initialization block: 間違い数は0からスタート
- Red arrow pointing to "間違い数 を 1 ずつ変える": 間違える度に変数「間違い数」を1ずつ増やす
- Red arrow pointing to the "もし 間違い数 = 6" block: 間違い数が6になった時点で、ステージとそれまでの正解数を告げ、終了するプログラム
- Red arrow pointing to the "もし 正解数 > 9" block: 15問出題後、クリア失敗していなければ必ずステージクリアになるので、正解数による条件分岐は不要
- Red arrow pointing to the "もし 正解数 > 9" block: ↑ 不要

④-2

ステージが上がる毎に、問題が難しくなる（例：ステージ1は一桁の計算、ステージ2は二桁の計算）

● 設定 ※ ④-1 の設定に③を追加

- ① 1ステージごとに15問出題し、10問以上正解なら次のステージへ
- ② ステージは5まで
- ③ ステージごとの出題は「1から(9×ステージ数)までの数」から「掛けられる数」と「掛ける数」をランダムに選択する  
※ステージ1は「1～9」、ステージ2は「1～18」、ステージ3は「1～27」、…

● ④-1 で作成した計算クイズのプログラムにブロックを追加

<変数>    掛けられる数    掛ける数    正解    正解数    ステージ

がクリックされたとき

ステージ を 1 にする

正解数 を 0 にする

ステージ > 5 まで繰り返す

15 回繰り返す

掛けられる数 を 1 から 9 \* ステージ までの乱数 にする

掛ける数 を 1 から 9 \* ステージ までの乱数 にする

正解 を 掛けられる数 \* 掛ける数 にする

掛けられる数 と × と 掛ける数 と (は? をつなぐ をつなぐ をつなぐ と聞いて待つ

もし 答え = 正解 なら

正解! と 2 秒言う

正解数 を 1 ずつ変える

でなければ

間違い! と 2 秒言う

もし 正解数 > 9 なら

ステージ と ステージ と クリア をつなぐ をつなぐ と 2 秒言う

ステージ を 1 ずつ変える

正解数 を 0 にする

でなければ

ステージは と ステージ をつなぐ と 2 秒言う

正解数は と 正解数 をつなぐ と 2 秒言う

すべてを止める

全ステージクリア! と 2 秒言う

ステージ数に応じた乱数（設定③）

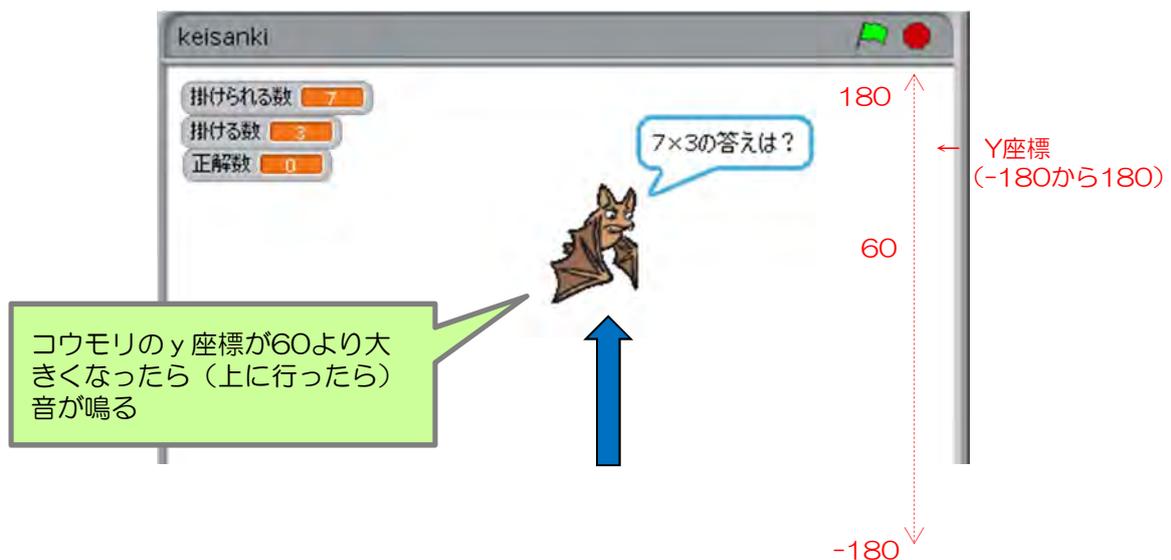
「演算」グループ  +

「変数」グループ  ステージ

④-3

制限時間が少なくなると、警告音になる

- サンプルプログラム③にブロックを追加



④-4

ハイスコアを残す仕組み

- 4時間目に作成した計算クイズのプログラムにブロックを追加

変数

掛けられる数    掛ける数

正解    ハイスコア

まずは変数「ハイスコア」を作る ハイスコア

※中学校1年生編の3時間目

④-2 参照

問題数を増やしておく

プログラムの下へ結合

追加するブロック  
※正解数がハイスコアを超えた場合、その正解数がハイスコアになる

「制御」グループ   

+

「演算」グループ   

+

「変数」グループ    正解数    ハイスコア

+

「見た目」グループ    こんにちは! と 2 秒言う

# 中学校 2 年生編

## 単元の構成

1 時間目～5 時間目：  
生活や社会を支える技術（情報通信技術について調べる活動）

1 時間目：デジタル化された情報を扱うコンピュータ

2 時間目：デジタル化された情報を伝えるネットワーク

3 時間目：ネットワークで情報を表現する Web

4 時間目：プログラムで処理を自動化する仕組み

5 時間目：デジタル化された情報と知的財産、サイバーセキュリティ

6時間目～15時間目：

技術による問題の解決（問題発見・課題設定、設計の活動）

6時間目：SNSの文字コミュニケーション技術をより良くしよう

7時間目：アクティビティ図で主要な機能と全体設計しよう

8時間目：基本設計の情報処理の手順を分解して考え、プログラムでアルゴリズムを実現しよう（Studuino）

9時間目：より使いやすくするための機能を考えよう（Studuino）

10時間目：グループで話すための機能を考えよう（Studuino）

11時間目：社会からの要求に応えよう(Studuino)

12～15時間目：

安全・適切なプログラムにしよう(Studuino)

16時間目～19時間目：

情報を利用するための基本的な仕組みと技術の評価

16～19時間目：

自分たちが制作したプログラムの紹介webページの制作

20時間目：

成果の評価と技術の概念

## 4/20時間目：プログラムで処理を自動化する仕組み

### ①アクティビティ図

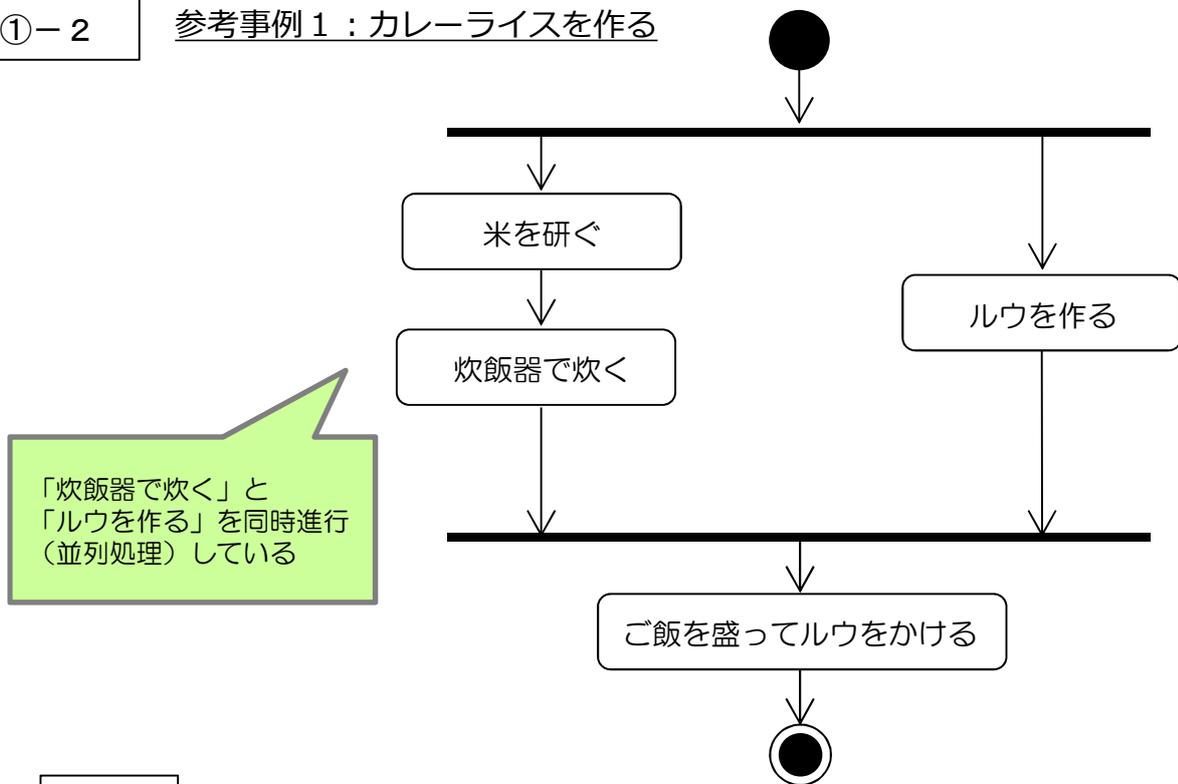
①-1

フローチャートではなくアクティビティ図を使う理由

- アクティビティ図では「並列処理」が表現できる
- また、後に学習するメッセージ交換などは、「並列処理」がなければ表現できない

①-2

参考事例1：カレーライスを作る

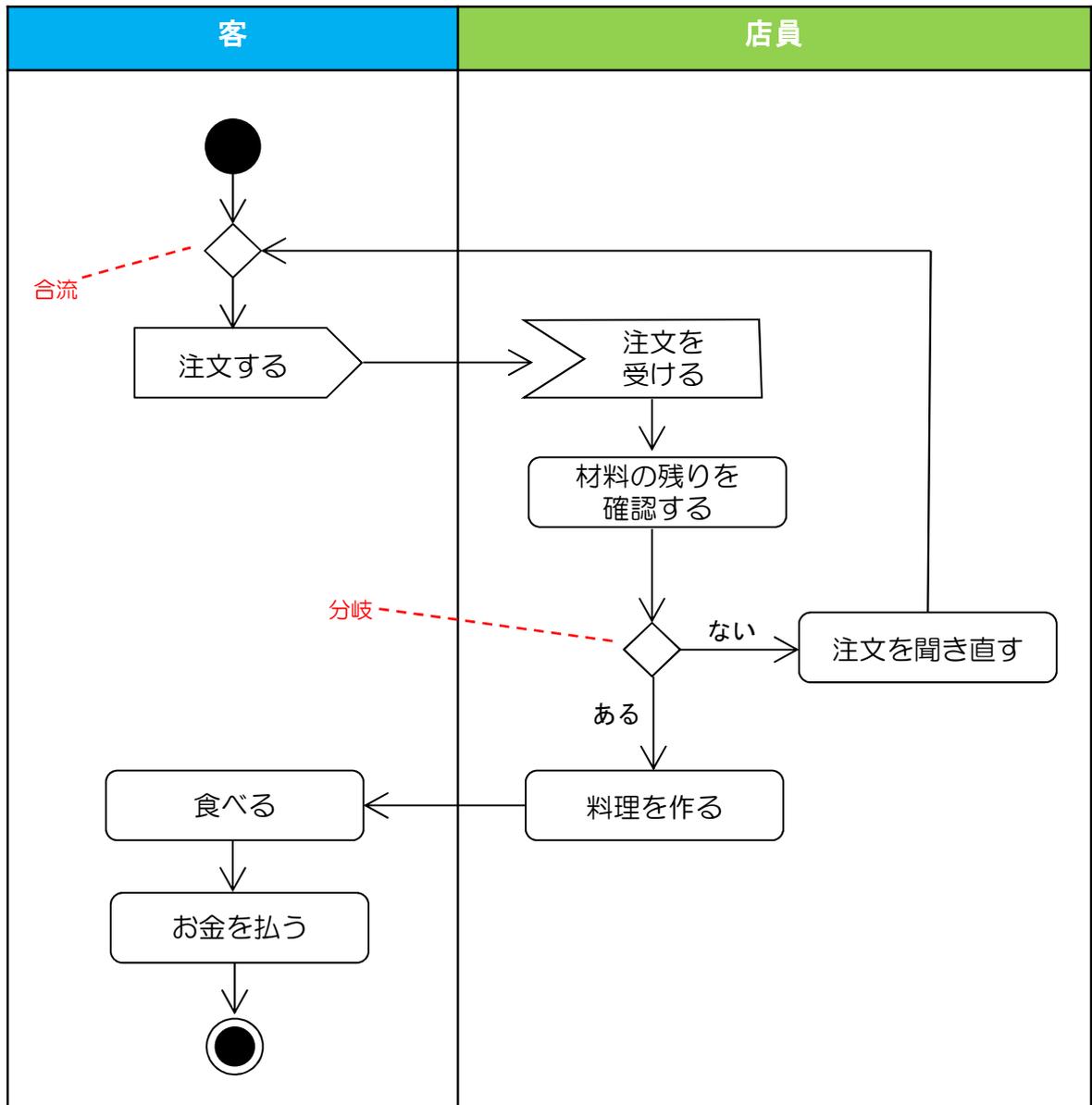


#### 凡例

- : 処理の開始
- ◎ : 処理の正常終了
- ⊗ : データを保持しない終了
- ◇ : 分岐 (Decision node) , 合流 (Merge node)
- : 並列, 非同期の処理開始(Fork node)と終了(Join node)

制御名 : 出力に関する処理

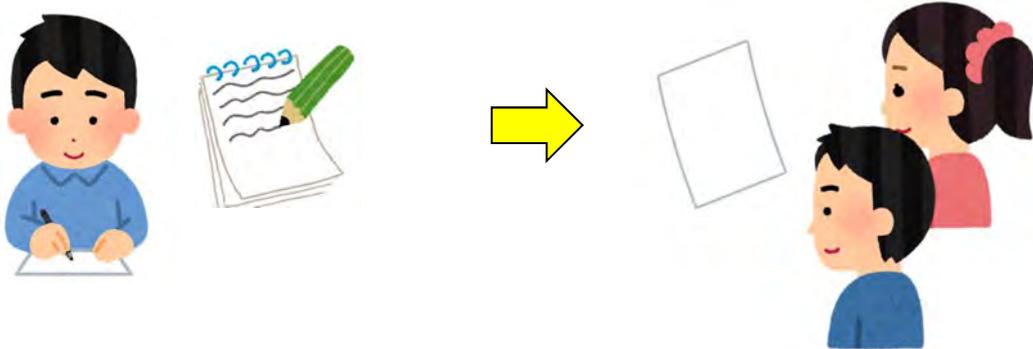
制御名 : 入力に関する処理



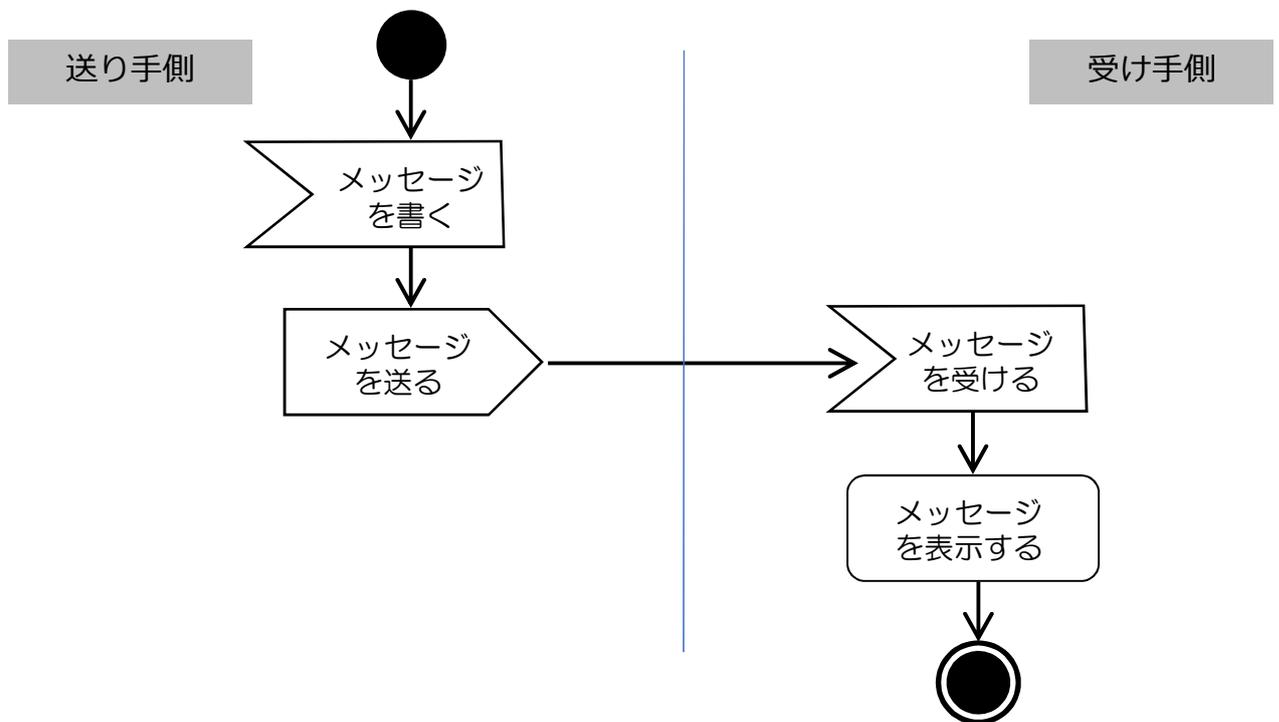
## ② SNS等の文字コミュニケーションアプリの仕組み

### ②-1 実際の人間による、記録が残る多人数コミュニケーションの手順

- メッセージを書いて、相手に渡す

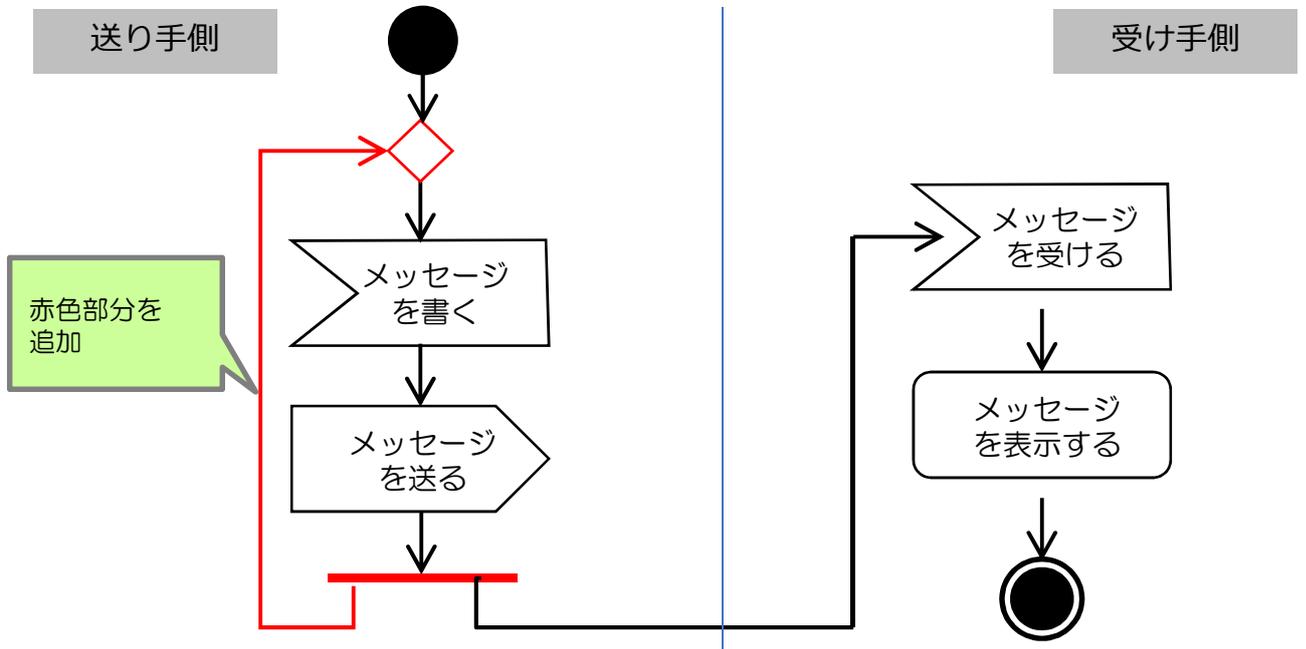


### ②-2 ②-1 をアクティビティ図で表現



②-3

メッセージを繰り返し送れるアクティビティ図



## 6/20時間目：SNSの文字コミュニケーション技術をより良くしよう

### ①問題解決のための課題の設定（指導過程5）

#### 〈技術で解決・解消・軽減できる課題〉

- [1]使い過ぎ防止のためのタイマー機能
- [2]他人に使用されないための個人認証機能

など

### ②機能の詳細・精緻化（指導過程6）

#### [1]使い過ぎ防止のためのタイマー機能

##### 〈詳細・精緻化の例〉

- 指定の時間になったらどうなるか
  - ・ 画像で知らせる
  - ・ 音で知らせる
  - ・ 強制的に電源を切る
- 知らせる方法
  - ・ どのような画像か
  - ・ どのような音か
  - ・ 音の大きさは

次の時間にアクティビティ図を書くので、手順が分かるように分解して考えさせる

など

#### [2]他人に使用されないための個人認証機能

##### 〈詳細・精緻化の例〉

- パスワードの入力方法は
  - ・ 数字を手入力
  - ・ 0～9の数字をクリックした順番（4桁）
- パスワードを間違った時
  - ・ 強制終了
  - ・ 繰り返しパスワードを入力できる

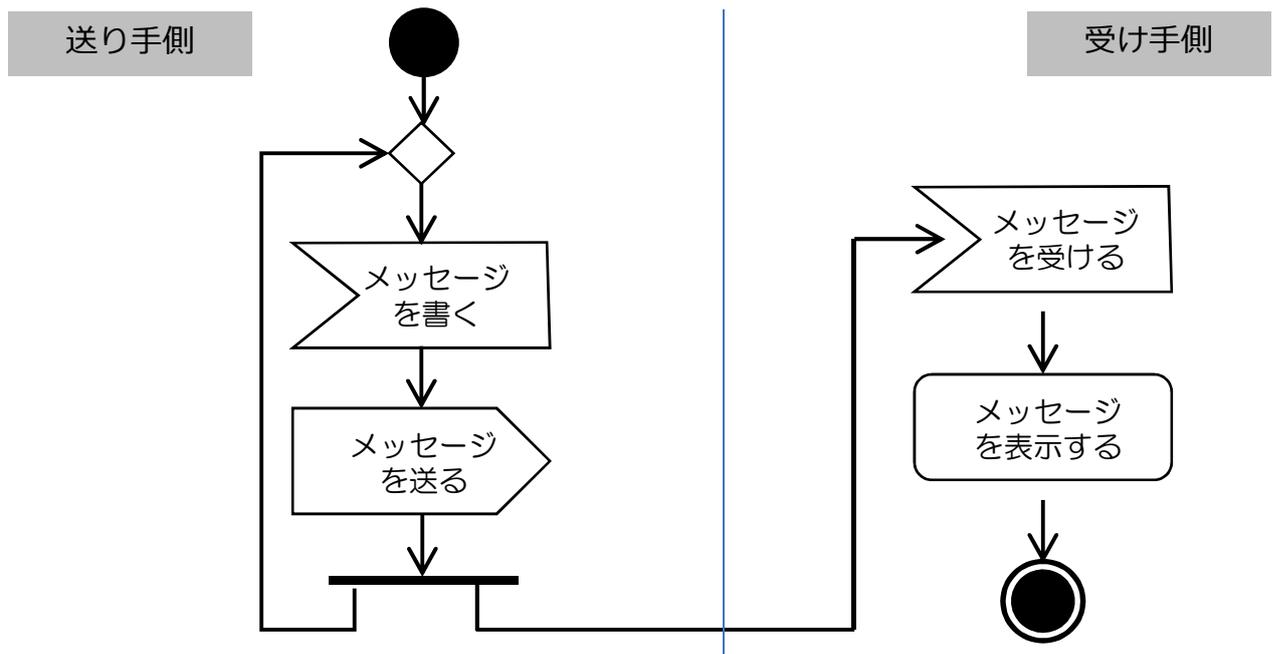
など

## 7/20時間目：アクティビティ図で主要な機能と全体設計しよう

### ①メッセージ交換の処理の確認（指導過程1）

#### 〈文字コミュニケーションの基本手順〉

※ 中学校2年生編の4/20時間目 ②-4 参照



### ②機能の詳細・精緻化（指導過程4）

②-1 アクティビティ図の書き方

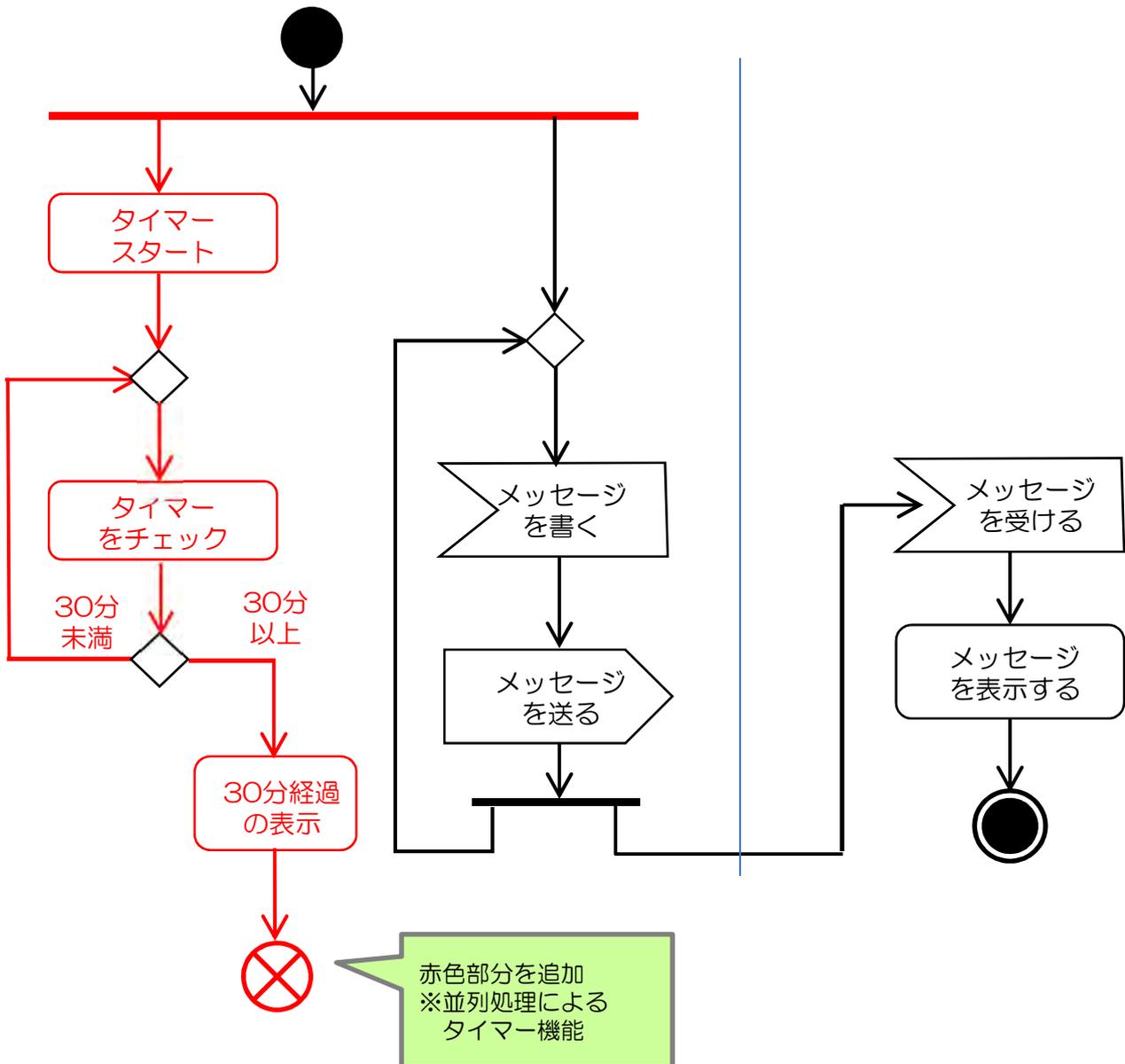
● 中学校2年生編の4/20時間目を参照

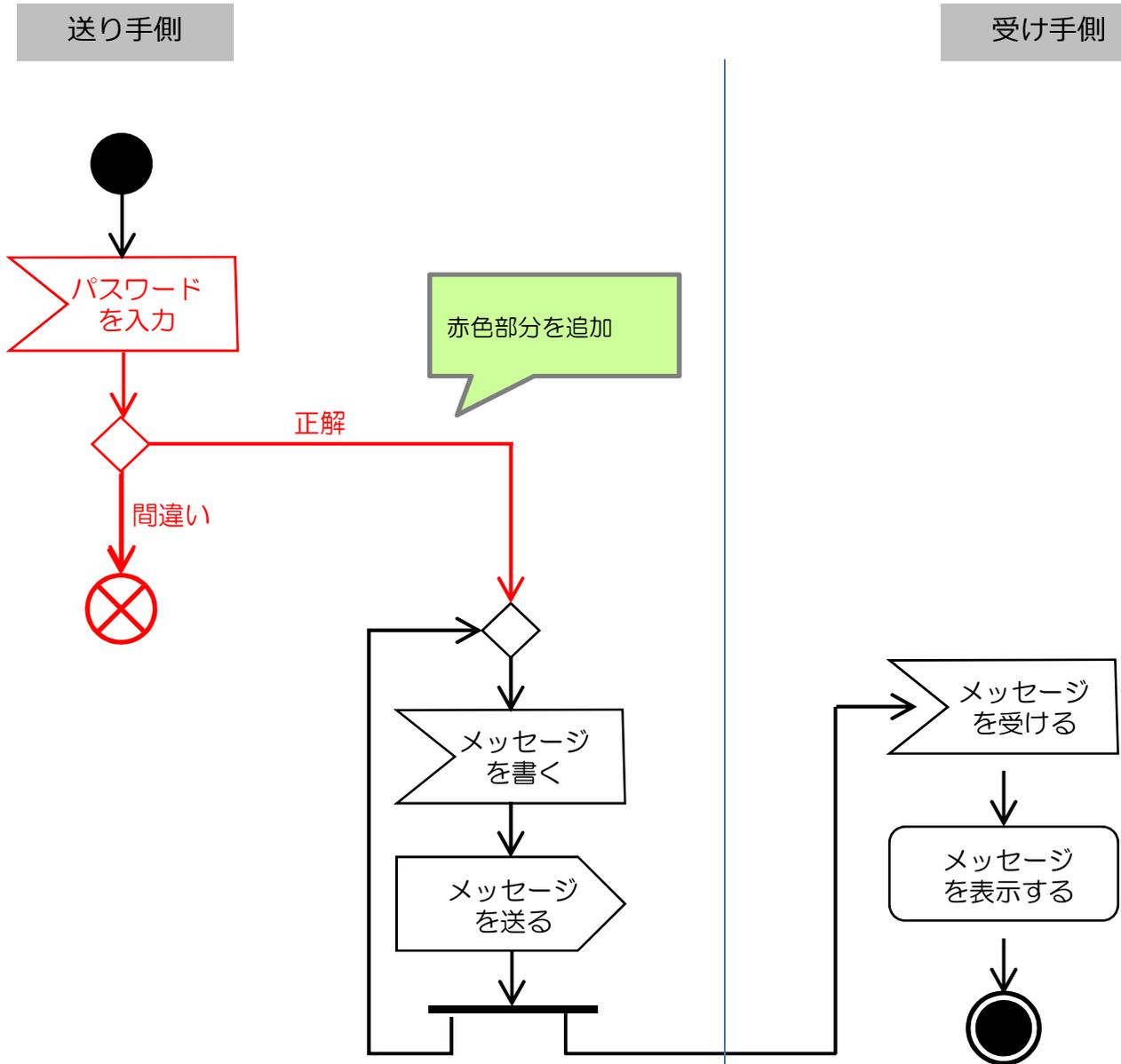
②-2

アクティビティ図の例 (タイマー機能 : 並列処理)

送り手側

受け手側





# 8/20時間目：基本機能の情報処理の手順を分解して考え、プログラムでアルゴリズムを実現しよう

## ①ソフトウェアの準備（指導課程3）

### ①-1 Studuino（スタディーノ）の起動

- スタディーノをキャラクターモードで使用

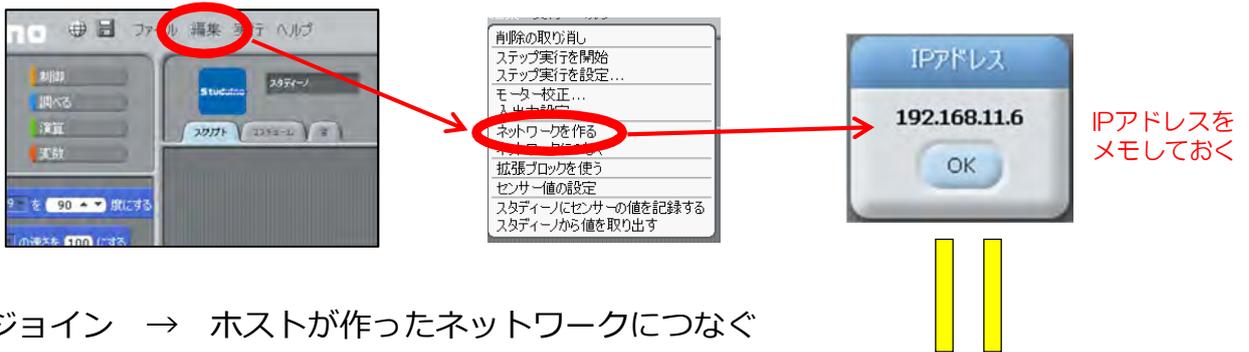


※iPad用アプリケーション「Tickle（ティックル）App for Studuino」では双方向通信はできない

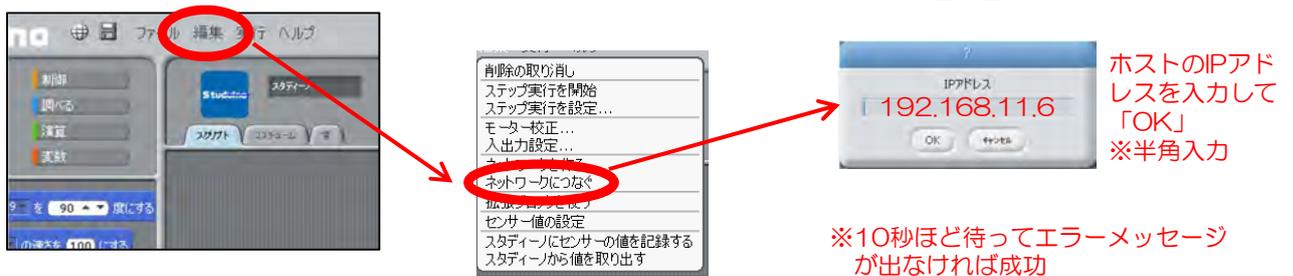
### ①-2 ネットワークを作る

**重要** ネットワークを作るためには、事前にパソコンの42001番ポートを開けておく必要がある

- ペアの中で「ホスト」と「ジョイン」を決める
- ホスト → ネットワークを作る



- ジョイン → ホストが作ったネットワークにつなぐ

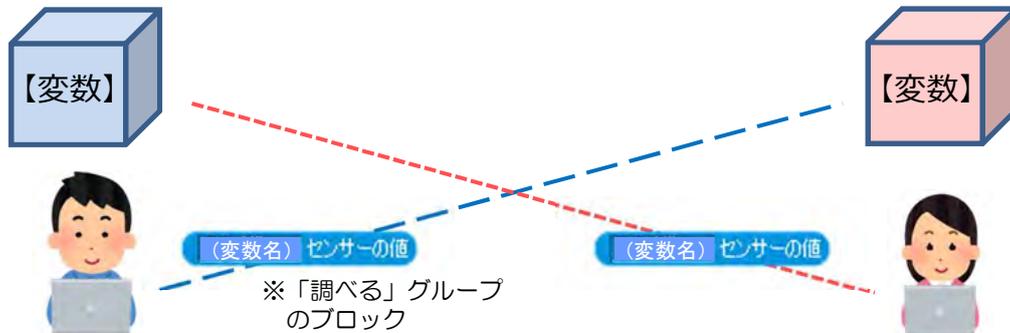


ネットワークが繋がらない場合は、ホストとジョインを入れ替えて試す

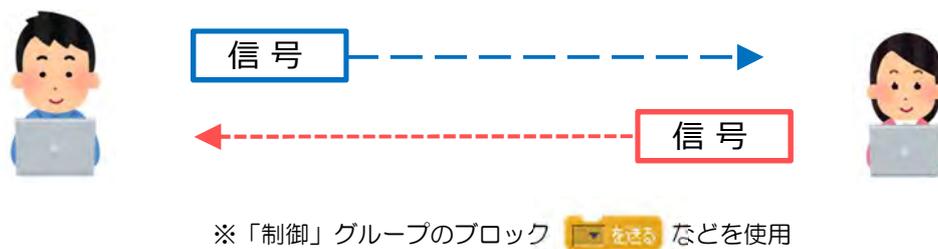
●このソフトウェアでは、ネットワークをつなぐことにより、以下のことが可能になります。

①変数を共有できる

②共有している変数は「センサーの値」として見ることができる



③互いに信号を送ることができる（例：メッセージを送る時の合図）



①-3

スプライトを追加

※スプライト=命令を与えるキャラクター

●Peopleフォルダのroundman（男）やsquaregirl（女）

中学校1年生編の3時間目

②-2

参照



例：roundman

## ②メッセージ送信手順を分解（指導課程3）

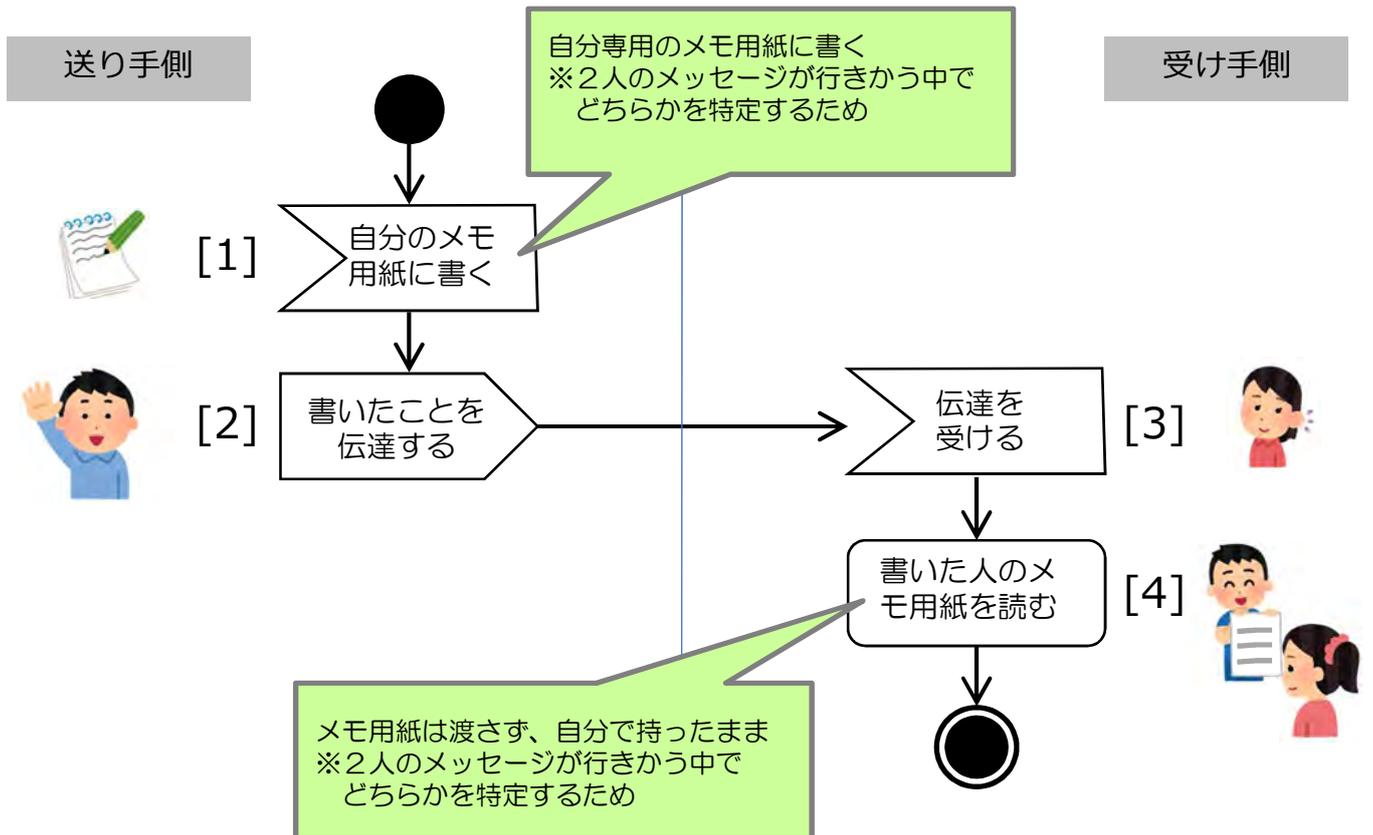
②-1

プログラムの基になるアクティビティ図

※ 中学校2年生編の4/20時間目

②-2

参照



送り手側（田中）

[1]

自分のメモ  
用紙に書く

「変数」グループ

たなかのメモ用紙 を 0 にする

+

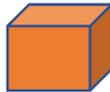
「調べる」グループ

あなたの名前は何ですか? と聞いて待つ

答え

### 〈変数を使う理由〉

- 変数はネットワーク内で共有されるので、それを利用してメッセージを送受信する。
- まずは変数「たなかのメモ用紙」を作る **たなかのメモ用紙**  
 ※中学校1年生編の3時間目 **④-2** 参照  
 ※「たなか」は生徒の名字の平仮名表記（変数名は任意）



たなかのメモ用紙

← この変数がネットワークで共有されるので、誰のメッセージか分かる名前にしておく

### 〈「…と聞いて待つ」「答え」ブロックを使う理由〉



← このブロックを使うと入力欄が表示され、好きな文章が入力できる。

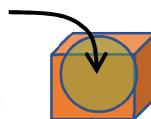
※入力した文章は **答え** ブロックに格納される



入力欄

### 〈ブロックの組み合わせ〉

入力した文章が共有されるように、変数の箱に入れる



たなかのメモ用紙

=



[2]

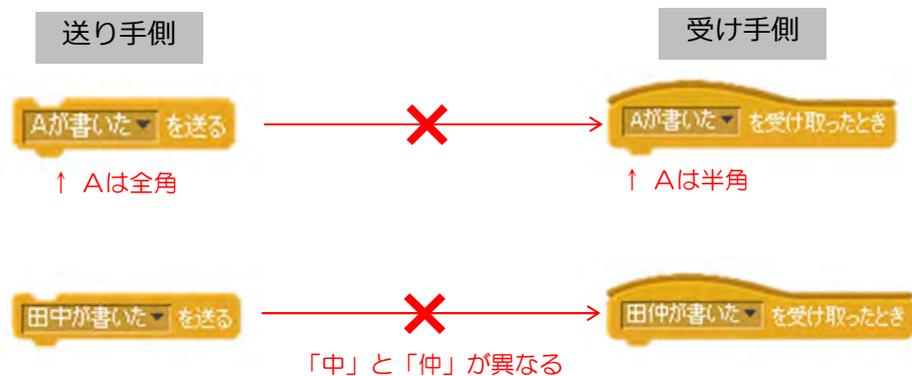
書いたことを  
伝達する



「制御」グループ  を送る  
※「たなかが書いた」は手入力

- これは「書いた」という信号を送るブロックで、変数ではない。
- 重要なことは、四角内の文字（この例では「たなかが書いた」）を受け手側が作るブロックと全く同じ文字を入力すること。

- アルファベットの全角・半角が異なるだけでも別の信号と判断される
- 漢字の場合、変換の際に異なる可能性がある



- よって、この例では生徒の名字を平仮名にして表記している。  
※同じ苗字の生徒がペアになった時は名字に「あ」や「い」を加えて「たなかあ」「たなかい」にするなど工夫する。
- 四角内の文字は任意であるが、この例では、実行内容がイメージしやすいように「…が書いた を送る」としている。

受け手側 (木村)

[3]

伝達を受ける



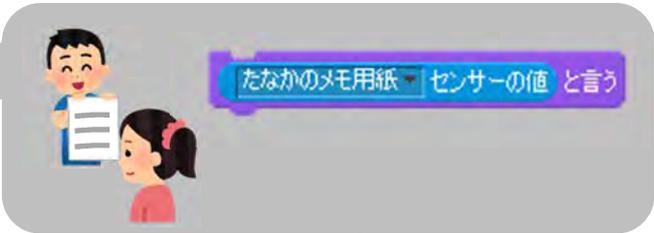
「制御」グループ



• これは「書いた」という信号を受け取るブロックで、変数ではない。  
 • 重要なことは、四角内の文字（この例では「たなかが書いた」）を送り手側が作るブロックと全く同じ文字を入力すること。

[4]

書いた人のメモ用紙を読む



※「たなかのメモ用紙」は変数

「見た目」グループ + 「調べる」グループ  
 こんにちは! と言う + スライダー ▼ センサーの値

• 外部の値を読み取るブロック  
 • 変数は共有されているため▼から「たなかのメモ用紙」を選択

②-3

ブロックの結合

送り手側



受け手側



<変数> たなかのメモ用紙



変数を読み取る

ただし、これではメッセージを送る度に旗をクリックしなければならない。②-4 以降で解決を図る。

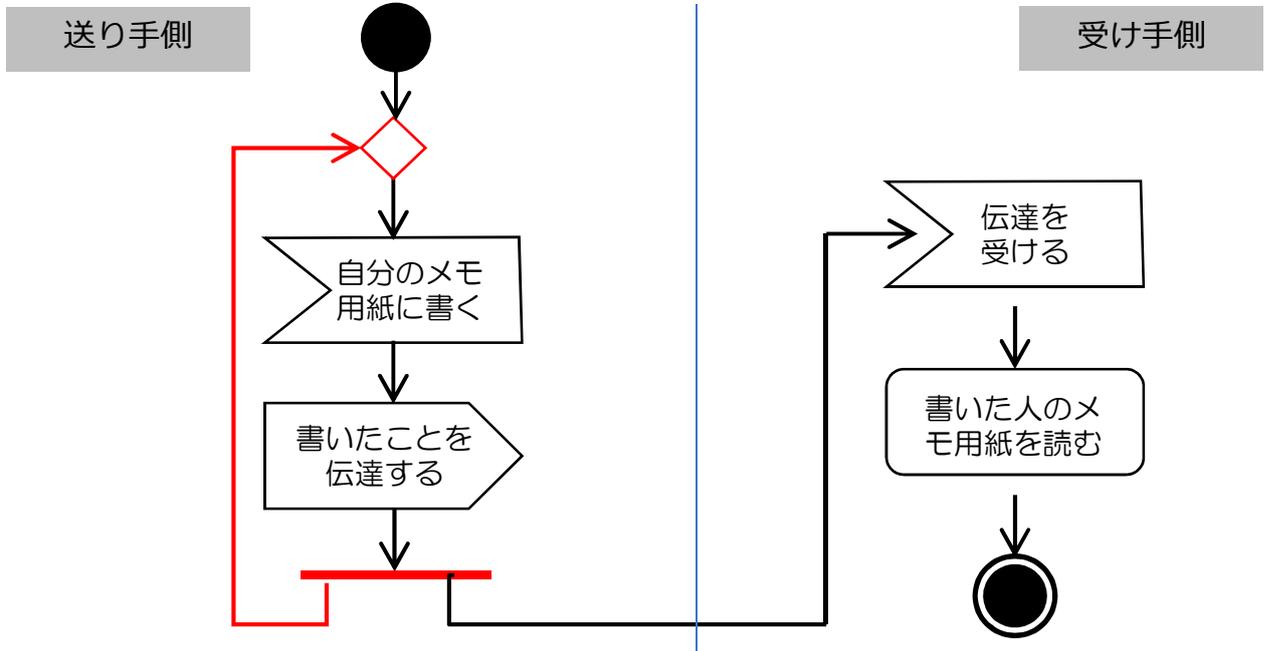
②-4

繰り返しメッセージを送れるアクティビティ図

※ 中学校2年生編の4/20時間目

②-4

参照



②-5

プログラムで実現（繰り返しメッセージ送信）



### ③双方向コミュニケーションの実現（指導過程4）

- **スプライトを二つ用意して**，それぞれに送り手側と受け手側のプログラムを割り付ける

#### ③-1 2つ目のスプライトを追加

- Peopleフォルダのroundman（男）やsquaregirl（女）

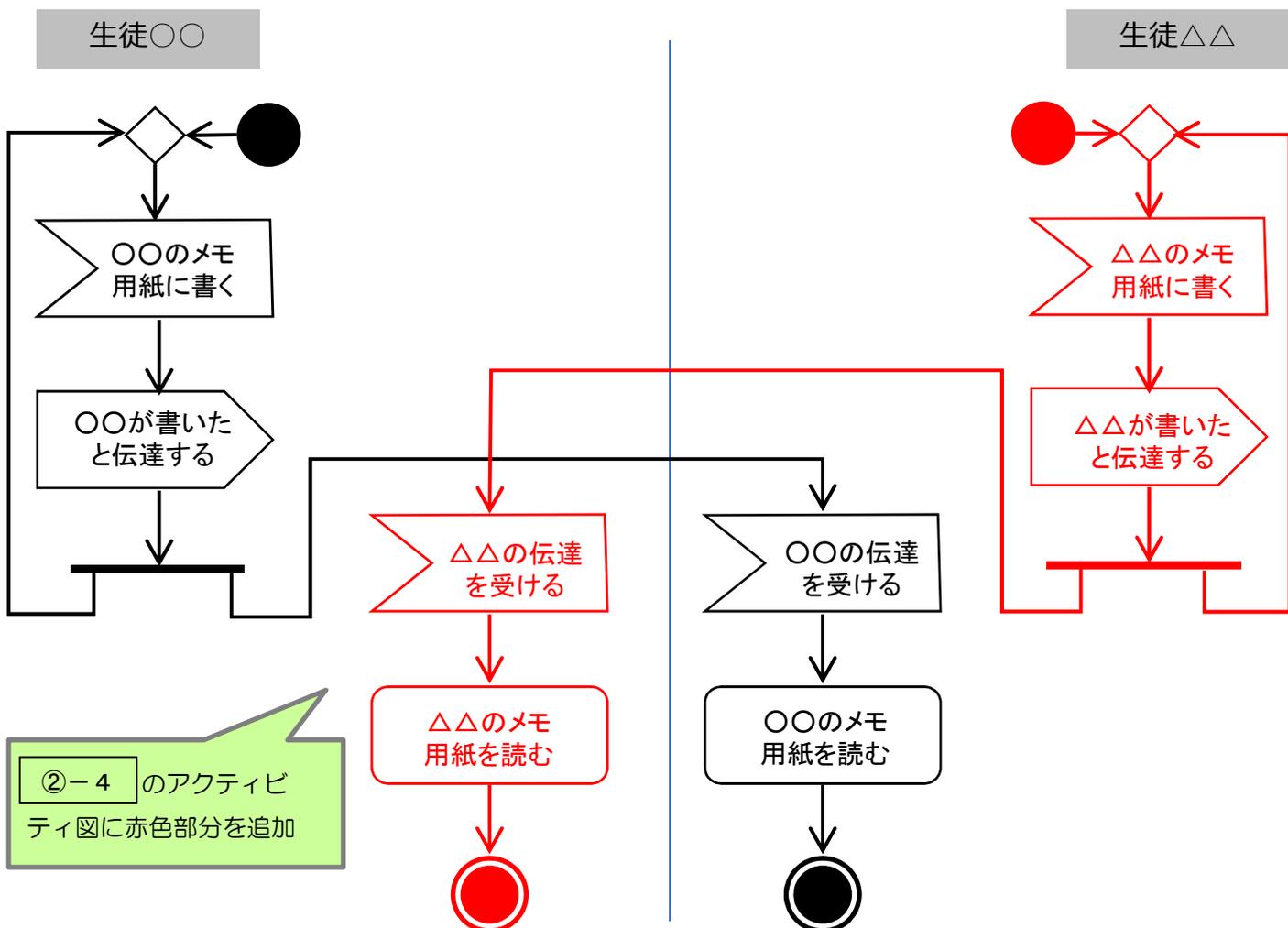
中学校1年生編の3時間目 ②-2 参照

- スプライトを追加する理由

- ・ 1つのスプライトで送信・受信をすると、メッセージが表示される場所が同じになり、誰のメッセージか分からなくなる。Studiinoで解決するには、送信と受信でスプライトを分けるしかない。

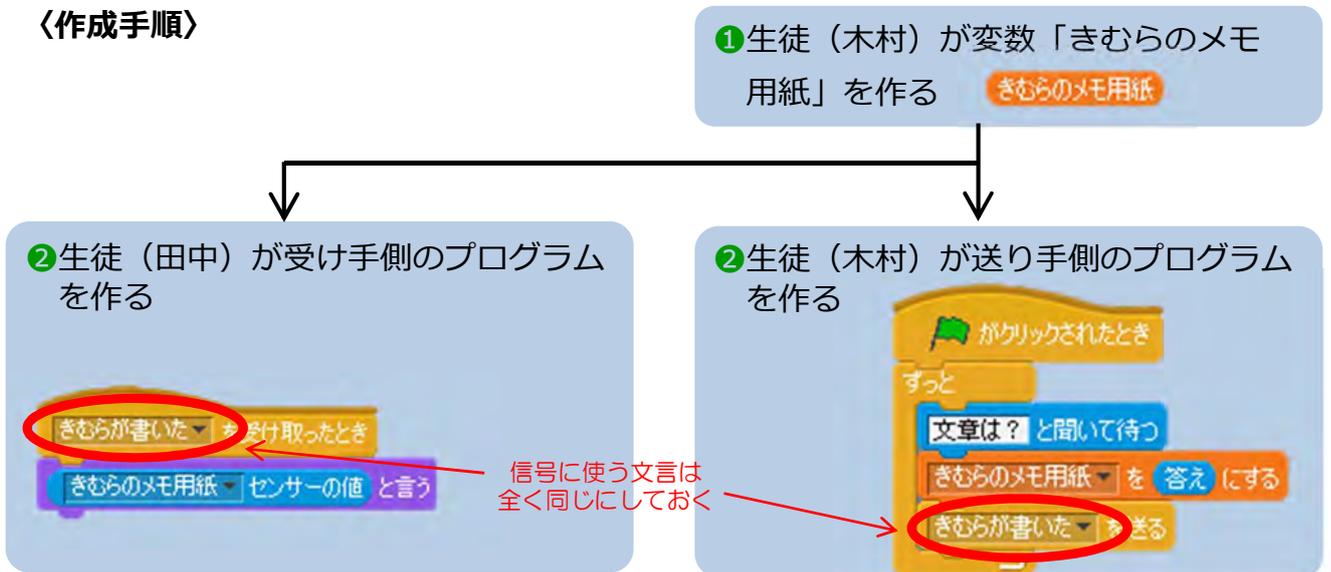


#### ③-2 双方向のアクティビティ図



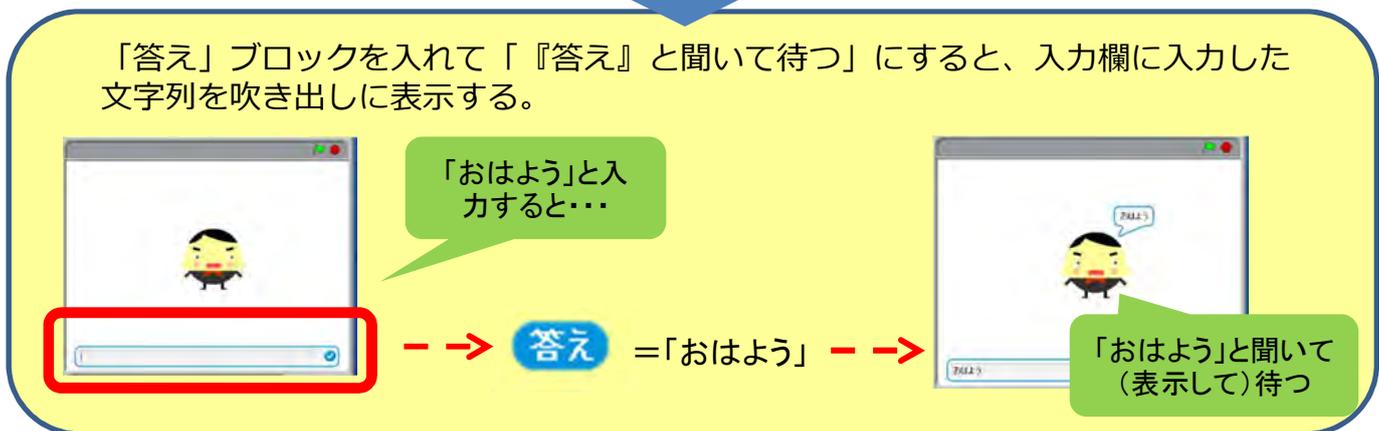
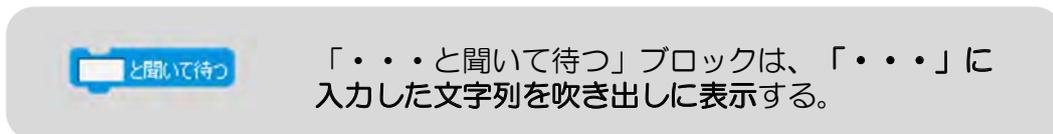
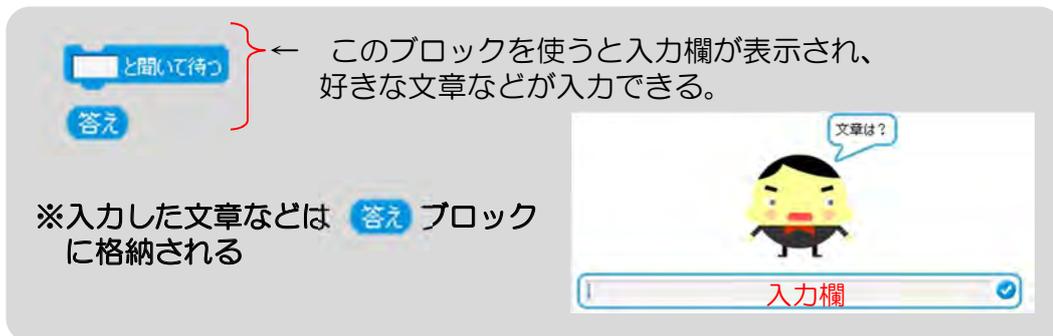


〈作成手順〉



#### ④自分が送った文章の表示（指導過程4）

- 入力した文章を表示しながら次の入力を待つ



#### ⑤プログラムの保存（指導過程4）

- 双方向で送受信できるプログラムを保存しておき、次の授業で拡張していく

※保存の方法は 中学校1年生編の4時間目⑥ 参照

## 9/20時間目：より使いやすくするための機能を考えよう

### ①ソフトウェアの準備（指導過程1）

#### ①-1 Studuino（スタディーノ）の起動

※ 中学校2年生編の8/20時間目 ①-1 参照

#### ①-2 ネットワークを作る

※ 中学校2年生編の8/20時間目 ①-2 参照

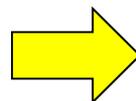
#### ①-3 双方向でメッセージを送受信するプログラムを開く

※ 中学校1年生編の5時間目 ①-1 参照

※ 開くプログラムは、前時に保存したもの又は教員が準備したサンプルプログラム

#### ①-4 プログラム保存時点とグループ（ペア）のメンバーが異なる場合は、変数と信号のブロックを調整する

〈プログラム保存時点：田中、木村ペア〉  
※田中のプログラム



〈本時：山本、木村ペア〉  
※山本のプログラム



## ②プログラム例（指導過程4）

### ②-1 受信時に着信音が鳴るプログラム例

#### 〈受信用スプライト〉



※ 音の換え方などは中学校1年生編の5時間目 ③-3 参照

### ②-2 受信時に画面が変化するプログラム例

#### 〈受信用スプライト〉



#### 〈受信を知らせるスプライト〉

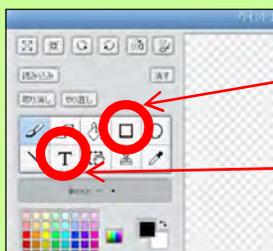
※新規作成

メッセージ

#### 作成手順



新しいスプライトを描く

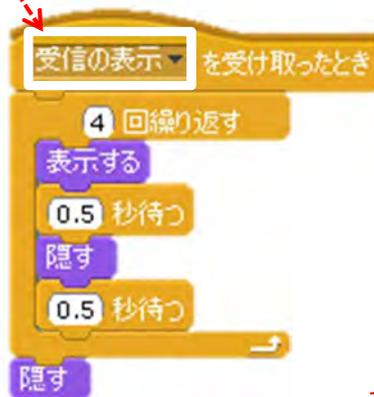


「四角形ツール」で色を付ける

「テキストツール」で文字を書く

信号

このように、自分から自分へ信号を送ることも可能



信号を受けたら点滅するプログラムを作成

実行エリア

メッセージ



## 10/20時間目：グループで話すための機能を考えよう

### ①ソフトウェアの準備（指導過程1）

- ※ 中学校2年生編の9/20時間目①参照
- ※ ネットワーク作成時は、以下のとおりにする。
  - 一人のコンピュータがホスト役
    - ・ネットワークを作る
  - 他の生徒のコンピュータはすべてジョイン（参加）
    - ・ホストが作ったネットワークにつなぐ

この時点では1対1の送受信ができるプログラム。以降でグループチャットに拡張していく。

### ②グループチャットができるプログラム（指導過程5）

- 人数分のスプライトを用意して、自分のスプライトには送り手側のプログラムを、他のスプライトには、それぞれの生徒の受け手側のプログラムを作成する

#### ②-1 スプライトを追加

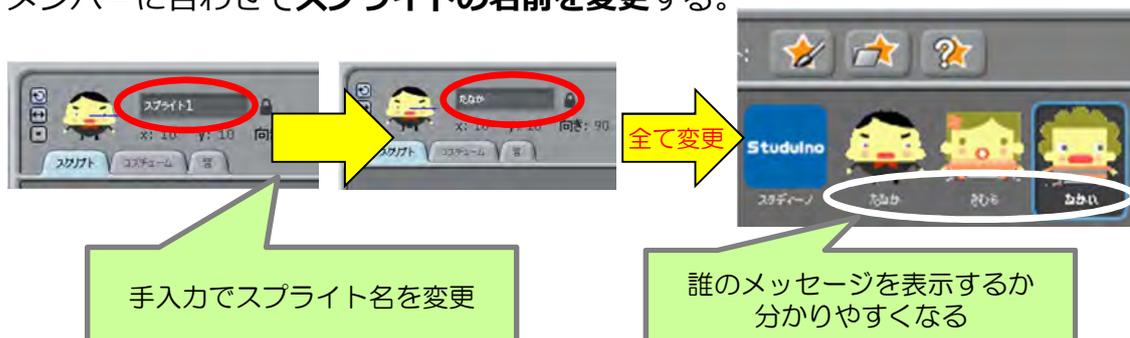
中学校1年生編の3時間目 ②-2 参照

- 3人グループの場合は、3つのスプライトが必要になるため、1つ追加する。

〈例：田中、木村、中井グループ〉  
※田中のプログラム



- どのスプライトが誰のメッセージを表示するか分かりやすくするために、グループのメンバーに合わせてスプライトの名前を変更する。



## ②-2 スプライト間でプログラムをコピー

- 受信のプログラムは、ほとんど同じなのでコピー機能を使うと早い

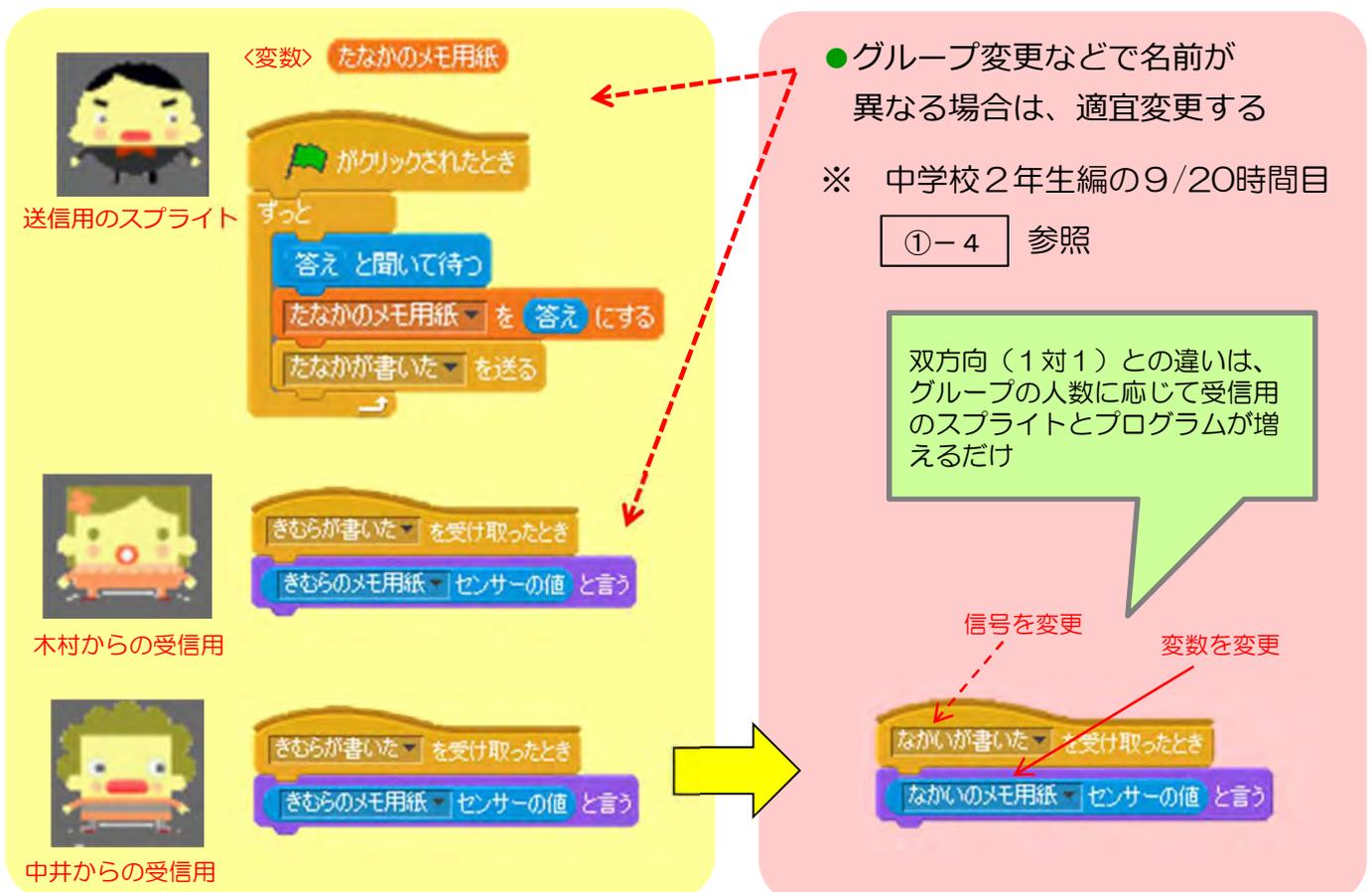
中学校1年生編の3時間目 ⑥-2 参照



## ②-3 変数と信号のブロックを調整

- グループのメンバーに合わせて、変数と信号のブロックを変更する

〈例：田中、木村、中井グループ〉 ※田中のプログラム



## ③プログラムの保存（指導過程5）

- グループで送受信できるプログラムを保存しておき、次以降の授業で拡張していく

※保存の方法は 中学校1年生編の4時間目⑥ 参照

#### ④送る相手を指定（ダイレクトメッセージ）（指導過程5）

④-1

Stduino（スタディーノ）でのダイレクトメッセージのイメージ

〈例：田中、木村、中井グループ〉

①グループのメンバーとの間にはドアがある。



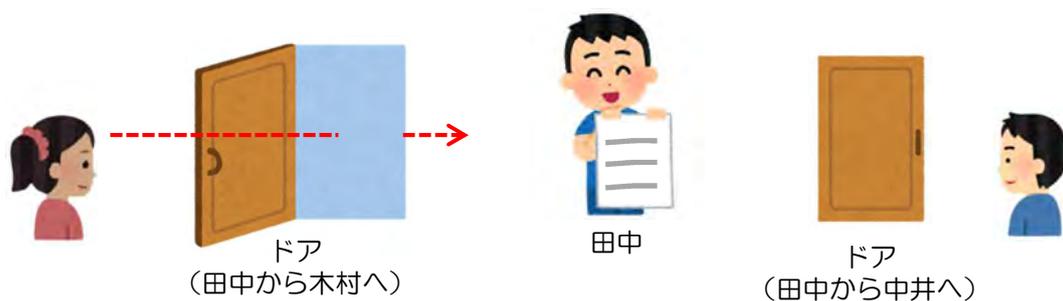
②送り手は、送りたい相手のドアだけ開ける。



③送り手は、自分のメモ用紙にメッセージを書いたら、書いたことを伝達する



④伝達はグループ全員に届くが、ドアが開いている人だけ送り手のメモ用紙を読む



※ ④-1 のイメージに沿って作成

①送信する時に使うドアを変数として作る。

(ドアは開閉により状態が変化するため変数を使う)



たなかからきむらへ

たなかからなかいへ

※変数の作り方は 中学校1年生編の3時間目 ④-2 参照

値が0なら閉まっている(送らない)、1なら開いている(送る) ことにする

たなかからきむらへ 0  
 たなかからなかいへ 0



たなかからきむらへ 1  
 たなかからなかいへ 0



たなかからきむらへ 1  
 たなかからなかいへ 1



最初は全てのドアを閉めておく

がクリックされたとき

- たなかからきむらへ を 0 にする
- たなかからなかいへ を 0 にする

ずっと

- 答え と聞いて待つ
- たなかのメモ用紙 を 答え にする
- たなかを書いた を送る

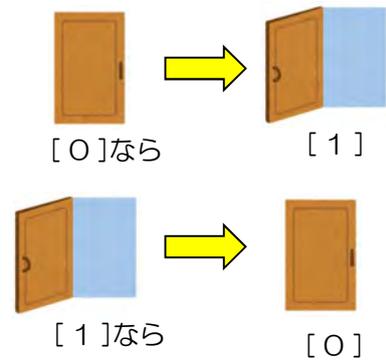
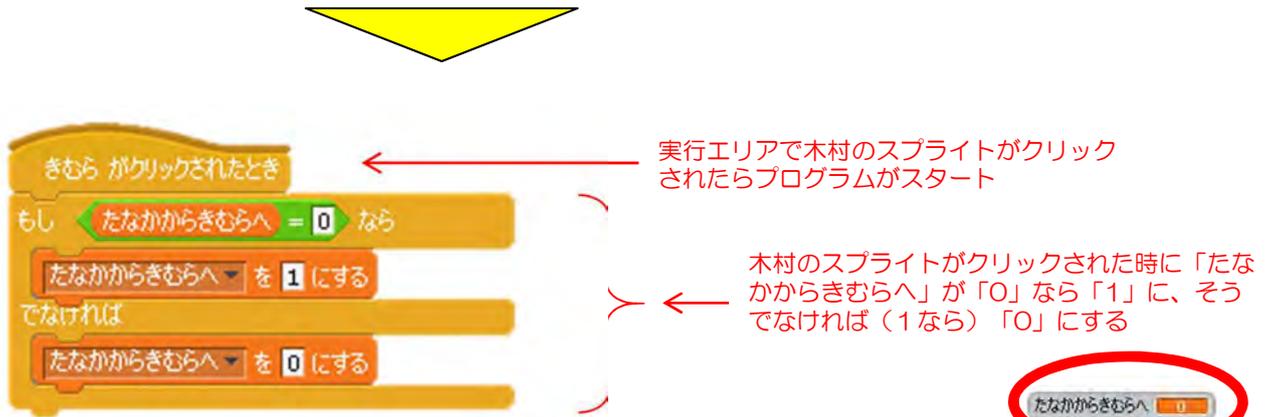
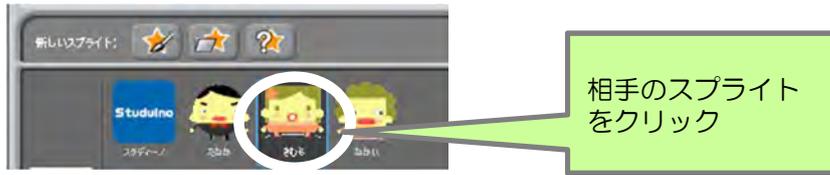
送信プログラムに  
変数ブロックを追加

たなかからきむらへ を 0 にする

※ ▼ボタンで変数名を選択する

②送り手が、送りたい相手のドアを開閉できるようにする。

(相手のスプライトをクリックすると変数が変わるようにする)

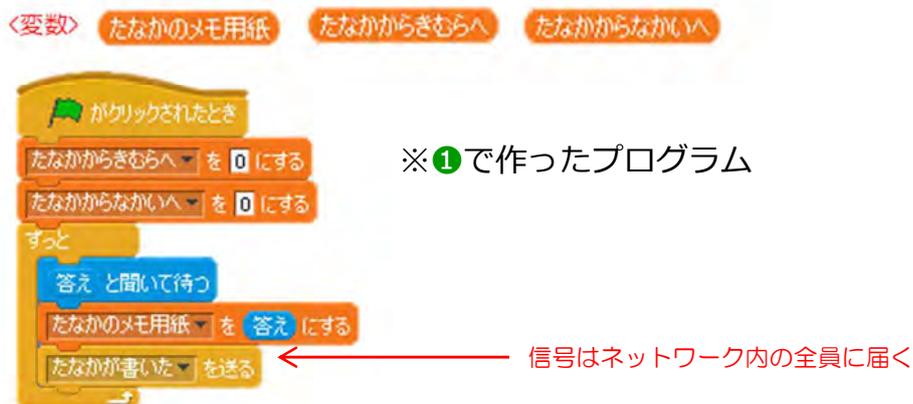


- コピー機能を使って、別の受信用スプライトにプログラムをコピーしておく

中学校 1 年生編の 3 時間目 ⑥-2 参照



- ③ 送り手は、自分のメモ用紙にメッセージを書いたら、書いたことを伝達する



- ④ 受け手は、ドアが開いている人だけ送り手のメモ用紙を読む

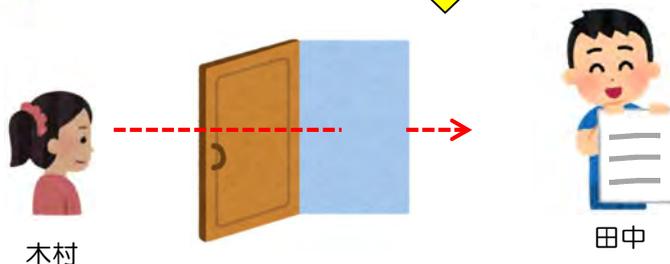
**注意：このプログラムは、送り手ではなく受け手が設定しなければならない**

〈例：木村のプログラム〉



※「たなかからきむらへ」は送り手が作った変数のため、受け手は「センサーの値」ブロックで確認する。

中学校 2 年生編の 8/20 時間目 ①-2 参照



## 〈例：田中、木村、中井グループの、田中のプログラム〉

## 送信用プログラム

〈変数〉 たなかのメモ用紙

たなかからきむらへ たなかからなかいへ



たなか

```

    がクリックされたとき
    たなかからきむらへ を 0 にする
    たなかからなかいへ を 0 にする
    ずっと
    答え と聞いて待つ
    たなかのメモ用紙 を 答え にする
    たなかを書いた を送る
  
```

## 木村へ送信するかどうかの選択用



きむら

```

    きむらがクリックされたとき
    もし たなかからきむらへ = 0 なら
    たなかからきむらへ を 1 にする
    でなければ
    たなかからきむらへ を 0 にする
  
```

たなかからきむらへ 0

↑ 1なら送信、0なら送信しない

## 中井へ送信するかどうかの選択用



なかい

```

    なかいがクリックされたとき
    もし たなかからなかいへ = 0 なら
    たなかからなかいへ を 1 にする
    でなければ
    たなかからなかいへ を 0 にする
  
```

たなかからなかいへ 0

↑ 1なら送信、0なら送信しない

## 受信用プログラム

## 木村からの受信用

```

    きむらを書いた を受け取ったとき
    もし きむらからたなかへ センサーの値 = 1 なら
    きむらのメモ用紙 センサーの値 と言う
  
```

## 中井からの受信用

```

    なかいを書いた を受け取ったとき
    もし なかいからたなかへ センサーの値 = 1 なら
    なかいのメモ用紙 センサーの値 と言う
  
```

## ⑤ コスチュームの変更（指導過程 5）

- 送る相手として指定されていることがわかるように
  - ・ コスチュームをコピー&編集して、クリックするたびに切り替える

① 送る相手のスプライトを選択し、「コスチューム」→「コピー」

② 「編集」をクリックして、○を付けるなど、ノーマルとの違いが分かるように編集する

③ コスチューム変更のプログラムを使うと、送る相手の選択状態が分かりやすくなる

「見た目」グループのブロックを追加

```

        スプライト2 がクリックされたとき
        もし たなかからきむらへ = 0 なら
            たなかからきむらへ を 1 にする
            コスチュームを squaregirl1 にする
        でなければ
            たなかからきむらへ を 0 にする
            コスチュームを squaregirl にする
    
```

[0]なら → [1] (コスチューム squaregirl1)
 [1]なら → [0] (コスチューム squaregirl)

## ⑥ さらに拡張するなら（指導過程 5）

- グループチャットとダイレクトメッセージの併用
  - ・ 送る相手が一人も指定されていなかったら、全員に送る

送り手

```

        がクリックされたとき
        たなかからきむらへ を 0 にする
        たなかからなかないへ を 0 にする
        ずっと
        答えと聞いて待つ
        たなかからきむらへ を 答え にする
        もし たなかからきむらへ = 0 かつ たなかからなかないへ = 0 なら
            たなかから全員 を 1 にする
        でなければ
            たなかから全員 を 0 にする
        たなかと言いた を送る
    
```

受け手

```

        たなかと言いた を受け取ったとき
        もし たなかからきむらへ センサーの値 = 1 または たなかから全員 センサーの値 = 1 なら
            たなかのメモ用紙 センサーの値 と言う
    
```

まずは変数「たなかから全員」を作る ※中学校1年生編の3時間目  
④-2 参照

送る相手が指定されているかどうかで条件分岐  
誰も指定されていなければ全員を指定  
誰かが指定されていれば全員指定はしない（指定済みの人のみ指定）  
自分又は全員が指定されていればメッセージを読む

## 11/20時間目：社会からの要求に応えよう

### ①ソフトウェアの準備（指導過程1）

- ※ 中学校2年生編の10/20時間目①参照
- ※ ただし、開くプログラムは中学校2年生編の10/20時間目③で保存したプログラム（グループチャットができるプログラム）
- ※ 中学校2年生編の10/20時間目 ②-1 を参照し、スプライト名も変更する

### ②プログラム例：絵文字としてスプライトを表示（指導過程4）

#### ②-1 プログラムの初期状態

- グループチャットができるプログラム（ダイレクトメッセージなし）

〈例：田中、木村、中井グループ〉 ※田中のプログラム

The screenshot displays a Scratch workspace with a yellow background. At the top, a variable named 'たなかのメモ用紙' (Tanaka's Memo Paper) is defined. Below it, three sprites are shown with their respective scripts:

- 送信用のスプライト (Sending Sprite):** A character with a sad face. Its script starts with 'がクリックされたとき' (When clicked), followed by a 'ずっと' (Forever) loop containing: '答え と聞いて待つ' (Ask and wait for answer), 'たなかのメモ用紙 を 答え にする' (Set Tanaka's Memo Paper to answer), and 'たなかを書いた を送る' (Send Tanaka's Memo Paper).
- 木村からの受信 (Receiving from Kimura):** A character with a happy face. Its script starts with 'きむらを書いた を受け取ったとき' (When Kimura's Memo Paper is received), followed by 'きむらのメモ用紙 センサーの値 と言う' (Say Kimura's Memo Paper sensor value).
- 中井からの受信 (Receiving from Nakai):** A character with a happy face. Its script starts with 'なかいを書いた を受け取ったとき' (When Nakai's Memo Paper is received), followed by 'なかいのメモ用紙 センサーの値 と言う' (Say Nakai's Memo Paper sensor value).

②-2

絵文字として表示するスプライトを追加

※ 中学校1年生編の3時間目 ②-2 参照



笑い



驚き

②-3

追加したスプライトに、特定の文字が送られた時だけ表示されるプログラムを作成



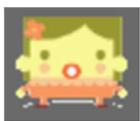
開始時点では表示しない



「おどろき」という信号を受け取ったら1秒だけ表示

②-4

受信スプライトに、特定の信号を送るプログラムを追加



「わらい」という文字が送られたら「わらい」という信号を送る

「おどろき」という文字が送られたら「おどろき」という信号を送る

〈イメージ〉



# 12~15/20時間目：安全・適切なプログラムにしよう

## ①ソフトウェアの準備

- ※ 中学校2年生編の10/20時間目①参照
- ※ ただし、開くプログラムは中学校2年生編の10/20時間目③で保存したプログラム（グループチャットができるプログラム）
- ※ 中学校2年生編の10/20時間目 ②-1 を参照し、スプライト名も変更する

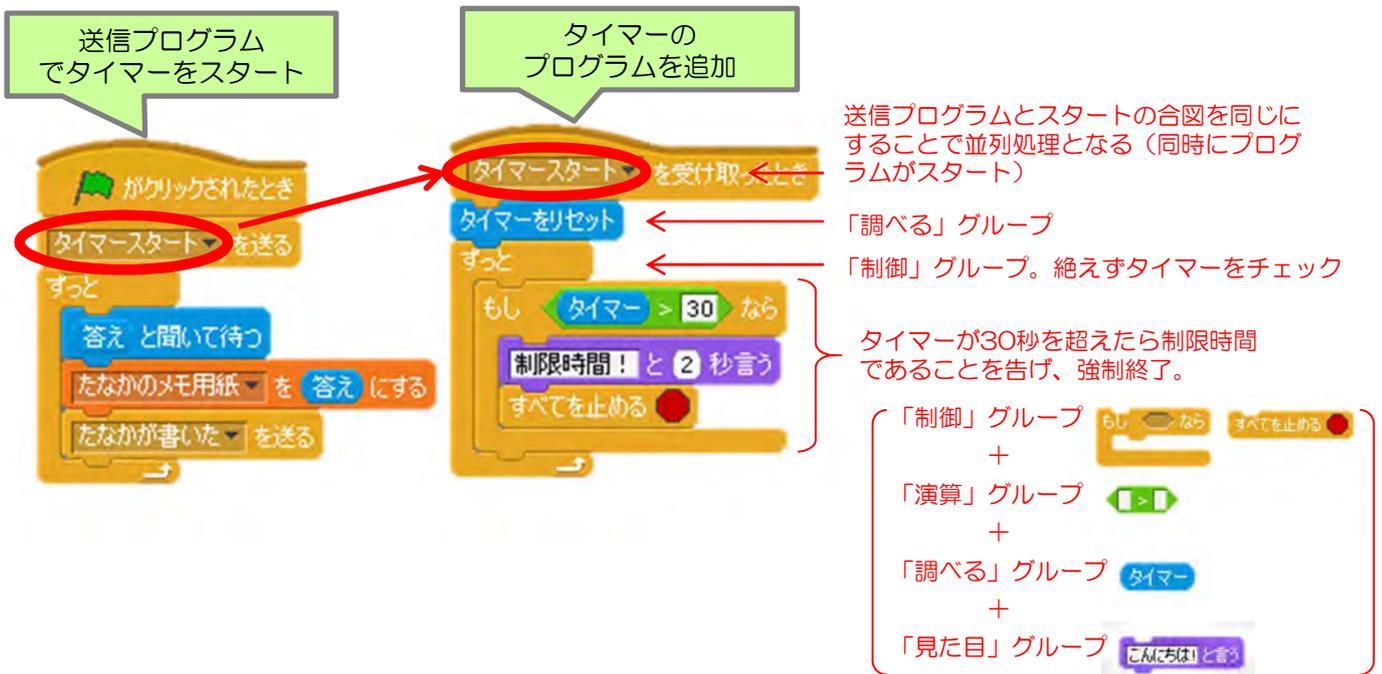
## ②プログラム例（タイマー機能：並列処理）

### ②-1 アクティビティ図

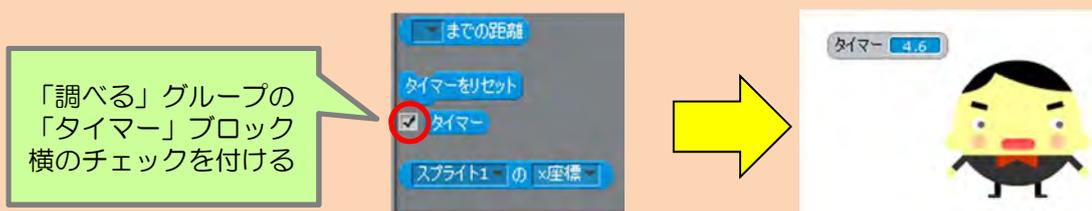
- ※ 中学校2年生編の7/20時間目 ②-2 参照

### ②-2 プログラム例（送信用スプライト）

〈例：田中、木村、中井グループ〉 ※田中のプログラム



### ●タイマーを実行エリアに表示するときは



### ③プログラム例（個人認証機能：「聞いて待つ」ブロックを使用する例）

#### ③-1 アクティビティ図

※ 中学校2年生編の7/20時間目 ②-3 参照

#### ③-2 プログラム例（送信用スプライト）

〈例：田中、木村、中井グループ〉 ※田中のプログラム

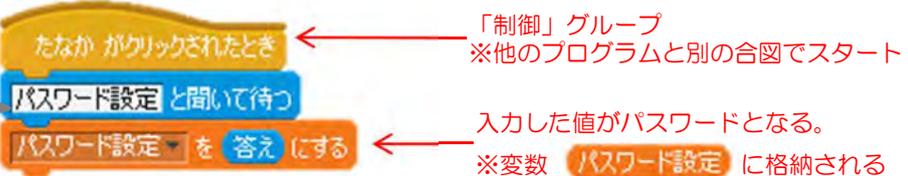


〈変数〉 たなかのメモ用紙 パスワード設定

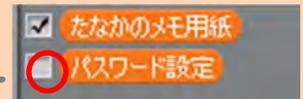
まずは変数「パスワード設定」を作る パスワード設定  
 ※中学校1年生編の3時間目

④-2 参照

[追加]  
パスワードを設定するプログラム



パスワードは非表示にしておく  
※変数のチェックを外す



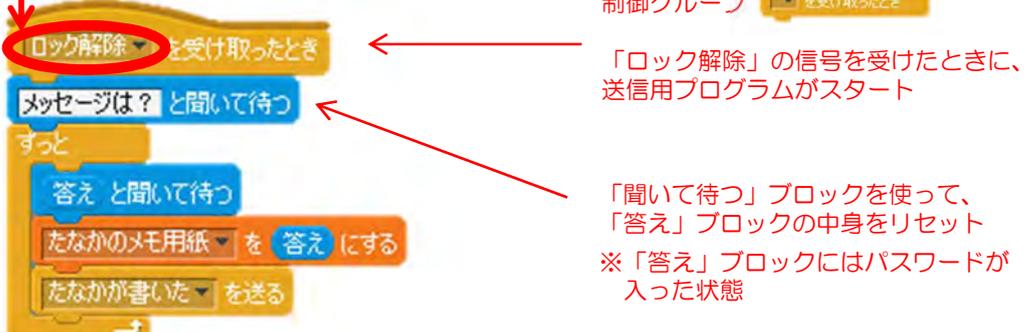
[追加]  
入力されたパスワードを確認するプログラム



※自分のスプライト間で信号を送っている

制御グループ を受け取ったとき

[変更]  
送信用プログラムをベースに微修正



## ④プログラム例（個人認証機能：数字をクリックしてパスワード入力する例）

④-1

### 設定の整理

- 1～9のを用意し、それをクリックして予め設定したPIN番号と同じ順番であるかどうかを正誤判断するプログラム

- パスワード  
4桁の数字をクリックで入力

1 2 3 4 5 6 7 8 9

1～9の

- 確認方法

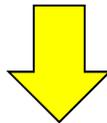
パスワードは、クリックして入力するため  
手入力する方法は使えない

と聞いて待つ

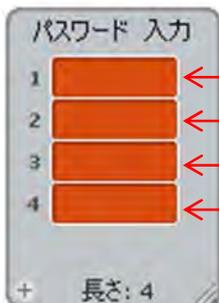
答え

を使って入力欄に

手入力する方法は使えない



- よって、「リスト」という機能を使用する



1番目にクリックした数字が入るようにする

2番目にクリックした数字が入るようにする

3番目にクリックした数字が入るようにする

4番目にクリックした数字が入るようにする

リスト



チェックしやすいように、あらかじめ  
設定しておくパスワードもリストの形にする

### 〈リストとは〉

- 「リスト」とは、複数の変数を入れる  
引き出しのようなものです。



- リストはいくつも作れるため、  
名前を付けておきます。

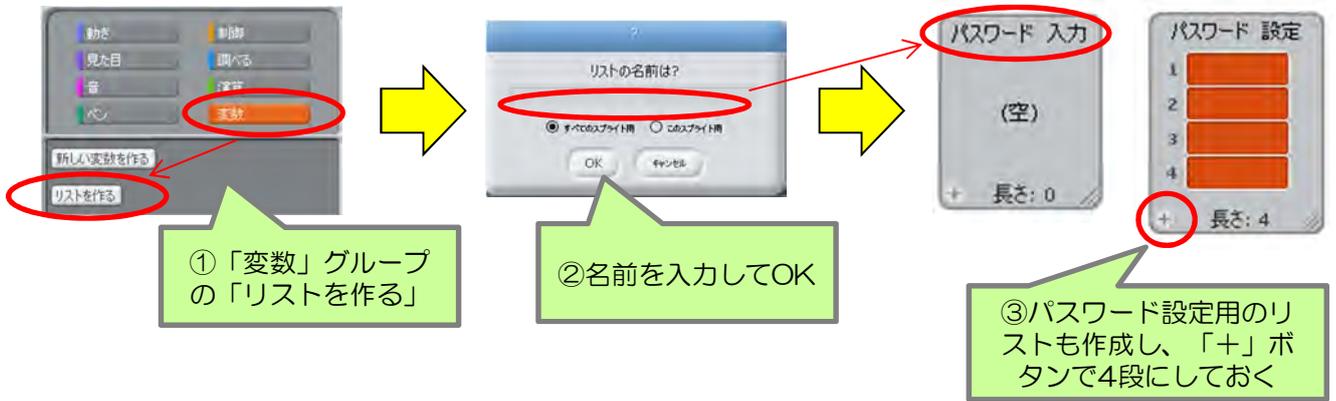
- 入っている変数の数も自由に  
変更できます。





〈例：田中、木村、中井グループ〉 ※田中のプログラム

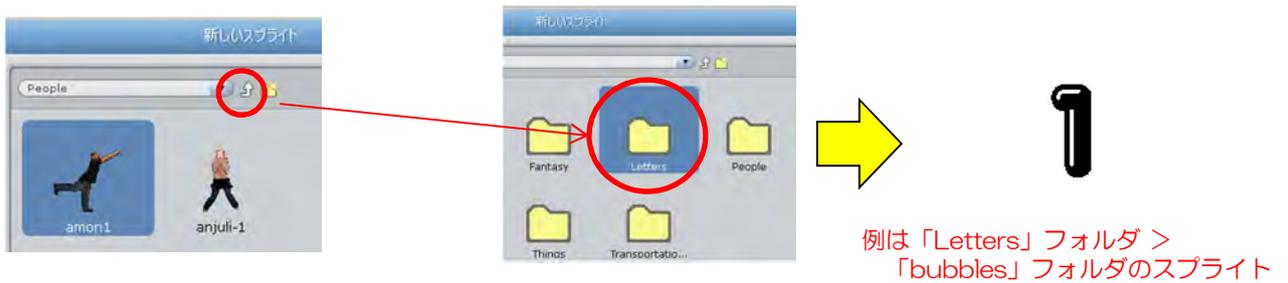
① リストを作成



② 数字のスプライトを追加

※ 中学校 1 年生編の3時間目 ②-2 参照

※ 「Letters」フォルダに数字のスプライトが格納されている



③ 数字のスプライトにプログラムを追加



#### ④パスワードを設定



実行エリアのリスト「パスワード 設定」に手入力する



設定後はリストを非表示にしておく

#### ⑤パスワードを正誤判定するプログラムを作成

まずは変数「チェック段数」を作る ※中学校1年生編の3時間目  
④-2 参照

**チェック段数**  
↑  
リスト内の何段目をチェックしているかを表す変数

〈変数〉 たなかのメモ用紙    **チェック段数**

がクリックされたとき

すべて 番目を パスワード 入力 から削除する ← 「変数」グループ。はじめはリストを空にしておく

表示する ← 「見た目」グループ。パスワード失敗時にスプライトを隠す設定にしているため、毎回開始時に表示

パスワードは? と言う ← 「見た目」グループ。「…と聞いて待つ」ブロックは「答え」ブロックとセットで使わなければうまく動かない

パスワード 入力の長さ = 4 まで待つ ← パスワードが4桁入力されるまで正誤判定を待つ

チェック段数 を 1 にする ← 「変数」グループ。リスト内の1段目から順にチェックしていくため初期値を1にする

チェック段数 > 4 まで繰り返す ← パスワードは4桁で、1桁ずつチェックするので4回チェックを繰り返す

もし パスワード 入力の チェック段数 番目 = パスワード 設定 の チェック段数 番目 なら

チェック段数 を 1 ずつ変える ← 「パスワード入力リスト」と「パスワード設定リスト」の、同じ段数が等しい数字になっているかどうかで条件分岐

でなければ

間違い! と 2 秒言う

隠す ← 同じ段数の数字が等しければ次の段へ (1段目どうしの数字が等しければ2段目へ)

すべてを止める

ロック解除 と 2 秒言う

ロック解除 を送る ← 「見た目」グループ。パスワードが間違っていればスプライトが消える

← 4段目までチェックして間違いがなければ「ロック解除」の信号を送る

自分のスプライト間で信号を送る

ロック解除 を受け取ったとき ← 送信用プログラムが、パスワード解除を合図にスタートするよう変更

メッセージをどうぞ と 2 秒言う ← メッセージ入力が可能になったことを告げる

ずっと

答え と聞いて待つ

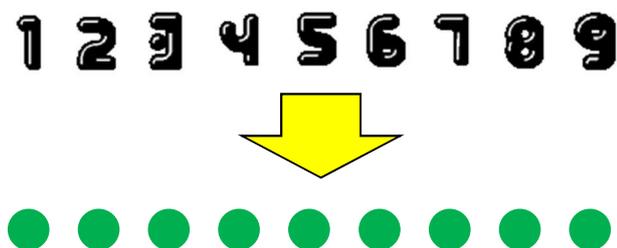
たなかのメモ用紙 を 答え にする

たなかを書いた を送る

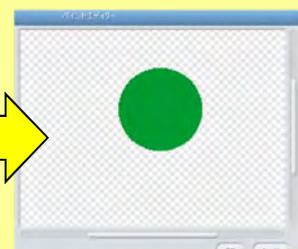
**⑤プログラム例（個人認証機能：ドットをクリックしてパスワード入力する例）**

●④作成した「数字をクリックしてパスワードを入力するプログラム」を変更する

①「1～9」の sprites をドットに変える



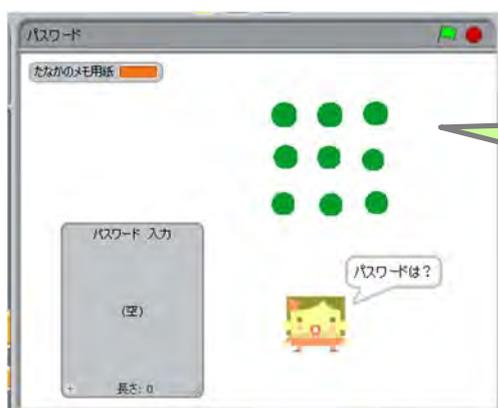
**〈 sprites の変更 〉**



シフトを押しながら  
○を描くと円になる

②ドットの配置を 3 × 3 に並べ替える

※プログラムは変更不要





## ②指導の概要（3～4時間で実施）

- ①前時までの活動について振り返る
- ②プログラムの中で、ホームページで発表したい部分を3つ選ぶ（タイマー機能など）
- ③内容の説明文を考える
- ④画面をスクリーンショットで保存し、必要な部分を画像編集ソフトウェアで適切に編集を行う（トリミングなど）
- ⑤HTMLについて振り返る（「きのくにICT教育」指導案では2年生の3時間目で履修）
- ⑥HTMLで紹介のページを作成する（複数時間にまたがる場合は、作業の進行状況を記録させておく。※どこまで作業が進んだか、次に何から始めるか、など）
- ⑦作成したHTMLをクラスで共有する
- ⑧振り返りを書く

・作成方法は以降で説明

## ③webページの編集

### ③-1 準備

- Webページのサンプルと画像データのサンプルをダウンロードしておく。

Webページのサンプル

画像データのサンプル

※ 画像データが3つ格納されている

1	2019/03/04 17:45	PNG ファイル	70 KB
2	2019/03/04 17:45	PNG ファイル	70 KB
3	2019/03/04 17:45	PNG ファイル	70 KB

サンプルファイルは、教育センター学びの丘の「きのくにeラーニング」システムの「きのくにICT教育（中学校）」に掲載

③-2

webページのソースを表示



- サンプルファイルをブラウザで開き、「表示」→「ソース」

又は

- サンプルファイルを右クリックして「プログラムから開く」→「メモ帳」

③-3

編集

〈サンプルファイルのソースコード〉

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>技術科 プログラミング</title><!--タイトル-->
<style type="text/css"><!--
body {background-color:#ffffff;
}
.title {
color:#006600;
text-align:center;
}
.text {
color:#333333;
line-height:130%;
}
--></style>
</head>
<body>
<h1 class="title">〇組 〇班 〇〇〇〇〇〇</h1><!--組、班名、氏名を入れてください-->

```

〈カラーコード〉  
色を決めるコードは16進数で表される。  
ブラウザで「カラーコード」等で検索すると一覧を確認できる。  
「# f f f f f f」=白  
「# 0 0 6 6 0 0」=濃厚な緑

氏名等を入力

```

<table cellpadding="0" cellspacing="10" border="0" align="center">
<tr>
<td width="320" height="240" valign="top">
<!-- 写真を指
</td>
<td width="250" valign="top" class="text">
<!-- ここより下に工夫した内容を記述します。-->
工夫した点

<!-- ここまで。-->
</td>
</tr>
</table>

```

写真①の大きさを指定

〈写真①を指定〉  
Imageフォルダ内「1.png」という名前の写真を表示 ※別の写真に同じ名前を付けて、同じフォルダに保存すれば、その写真が表示される。  
「1.png」を写真の名前に合わせて編集しても良い

工夫した点①を入力

〈次ページへ続く〉

## 〈前ページへからの続き〉

```
<table cellpadding="0" cellspacing="10" border="0" align="center">
<tr>
<td width="250" valign="top" class="text">
<!-- ここより下に工夫した内容を記述します。-->
```

工夫した点

```
<!-- ここまで。-->
```

```
</td>
```

```
<td width="320" height="240" valign="top">
```

```
<!-- 写真を指定。大
```

```
</td>
```

```
</tr>
```

```
</table>
```

```
<table cellpadding="0" cellspacing="10" border="0" align="center">
```

```
<tr>
```

```
<td width="320" height="240" valign="top">
```

```
<!-- 写真を指定。大
```

```
</td>
```

```
<td width="250" valign="top" class="text">
```

```
<!-- ここより下に工夫した内容を記述します。-->
```

工夫した点

```
<!-- ここまで。-->
```

```
</td>
```

```
</tr>
```

```
</table>
```

```
<table cellpadding="0" cellspacing="10" border="0" align="center" style="width:
```

```
<tr>
```

```
<td width="250" valign="top" class="text">
```

```
<!-- ここより下に感想を記述します。-->
```

感想<br>

```
<!-- ここまで。-->
```

```
</td>
```

```
</tr>
```

```
</table>
```

```
<p align="center"><a href="../index.html">戻る</a></p>
```

```
</body>
```

```
</html>
```

工夫した点②を入力

写真②の大きさを指定

＜写真②を指定＞

Imageフォルダ内「2.png」という名前の写真を表示 ※別の写真に同じ名前を付けて、同じフォルダに保存すれば、その写真が表示される。「2.png」を写真の名前に合わせて編集しても良い

写真③の大きさを指定

＜写真③を指定＞

Imageフォルダ内「3.png」という名前の写真を表示 ※別の写真に同じ名前を付けて、同じフォルダに保存すれば、その写真が表示される。「3.png」を写真の名前に合わせて編集しても良い

工夫した点③を入力

感想を入力

# 中学校3年生編

## 単元の構成

1時間目：ロボットが運転する利点と問題点をしらべよう

2時間目：様々なセンサを扱うコンピュータ

3時間目：問題の解決に役立つ計測・制御

4時間目：ロボットを作って動作を確かめる

5時間目：計測と制御のプログラミングでアクチュエータを動かす

6時間目：計測・制御システムの構想・設計

7～8時間目：プログラミング、制作

9時間目：ビジュアル言語とテキスト言語の比較

10時間目：成果の評価と技術の概念、改良と応用（社会の発展と技術）

## 1 時間目：ロボットが自動運転する仕組みと利点をしらべよう

- 本時で使用するテキスト等



「プログラムによる計測・制御」  
教師用指導案（2014年版）  
（株）アフレル



「プログラムによる計測・制御」  
生徒用ワークブック（2014年版）  
（株）アフレル

### ① センサとアクチュエータの理解（指導過程3）

- 「プログラムによる計測・制御」生徒用ワークブック P2については、教師用指導案 P9も参照

### ② 閾値の理解（指導過程4）

- 「プログラムによる計測・制御」生徒用ワークブック P24 も参照

## 2時間目：様々なセンサを扱うコンピュータ

- マイクロビットの使用方法は、中学校1年生編の2時間目②、③参照

### ① センサープログラムの体験（指導過程3）

- あらかじめ異なるプログラムが入った3種類のmicro:bitを触って、何をしたら何が起こるかを調べる ※各種センサの位置等は、中学校1年生編の2時間目を参照



#### プログラム①

[https://makecode.microbit.org/\\_8eci15KyaAAc](https://makecode.microbit.org/_8eci15KyaAAc) 又は [bit.ly/kinokuni-01](https://bit.ly/kinokuni-01)

#### プログラム②

[https://makecode.microbit.org/\\_VzWFEJJ6h5jM](https://makecode.microbit.org/_VzWFEJJ6h5jM) 又は [bit.ly/kinokuni-02](https://bit.ly/kinokuni-02)

#### プログラム③

[https://makecode.microbit.org/\\_AETdMjLow1b8](https://makecode.microbit.org/_AETdMjLow1b8) 又は [bit.ly/kinokuni-03](https://bit.ly/kinokuni-03)

- あらかじめマイクロビットにプログラムをダウンロードし、電池ボックスをつなげておく

①プログラムのダウンロード

②電池ボックスの接続

} 中学校1年生編の2時間目  
③-2 ③-3 参照

## ②プログラムの確認・変更（指導過程5）

②-1

エディターの起動



②-2

日本語表記への変更 ※初期設定で日本語表記の場合は作業不要



②-3

プログラムの確認

※プログラム①の例



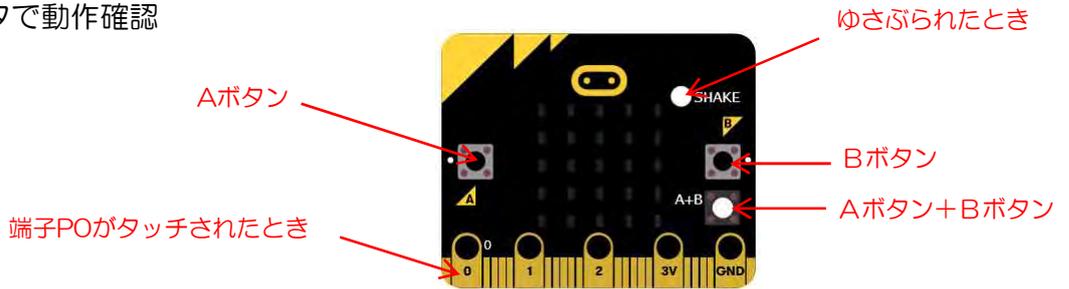
・最初は変数を0にする

・Aボタンを押す度に変数が1ずつ増えていき、その数字が表示される

・Bボタンを押すと変数が0になり、0が表示される

- プログラムを確認し，センサーとアクションの対応を予想しながらプログラムを変更してみる

- ・シミュレータで動作確認

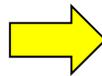


②-4

プログラムの変更例



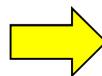
Aボタンを押す度に変数が1ずつ増えていき、その数字が表示される



変数が2ずつ増えるように変更



ゆさぶられたとき、うれしい顔を表示



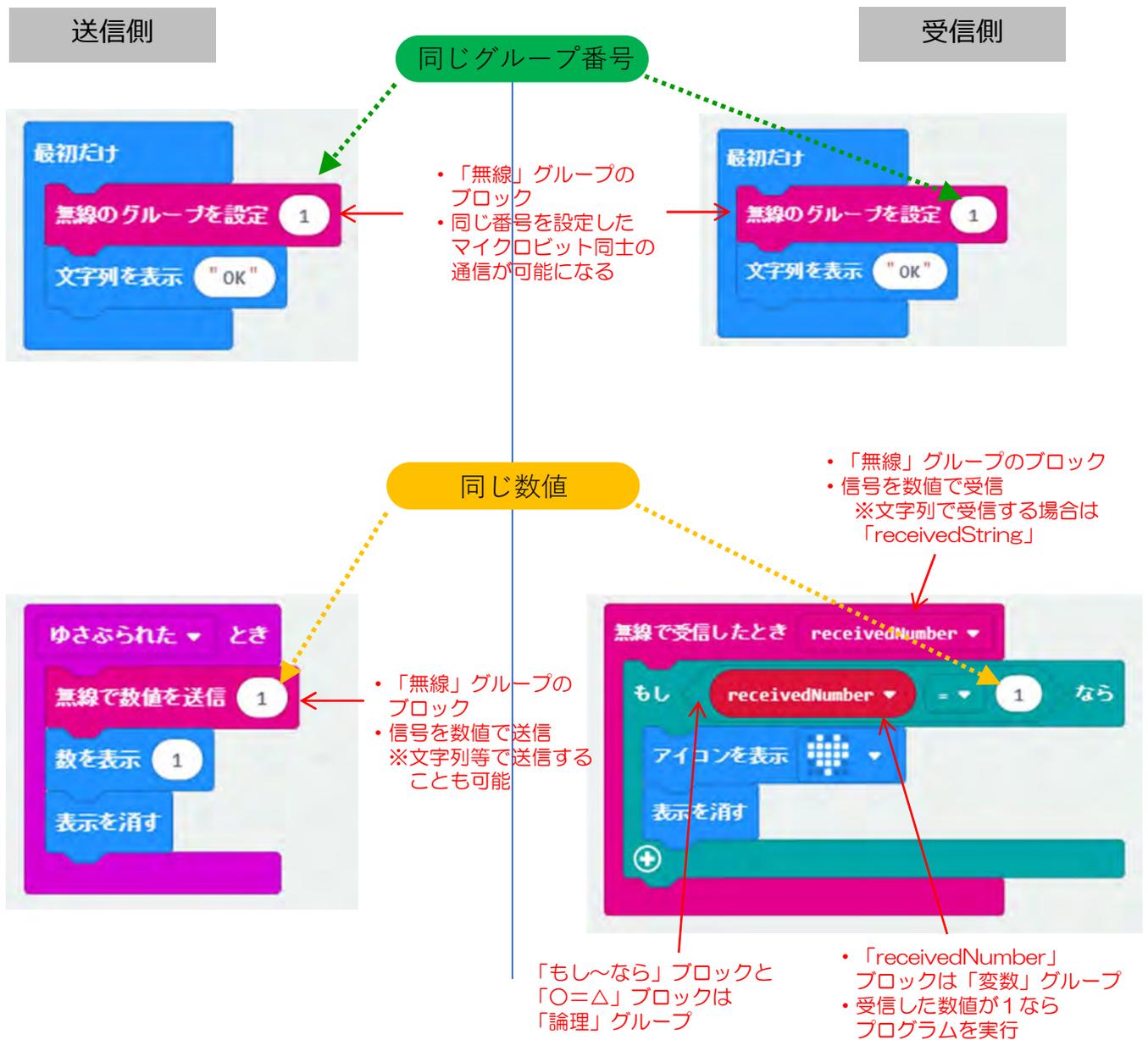
音符が表示されるように変更

### ③micro:bit (マイクロビット) の無線通信 (指導過程 6)

#### ③-1 グループ番号の割り当て

- 生徒の各グループに、「無線のグループ番号」を1, 2, 3...と割り当てていく
  - ・同じ「無線のグループ番号」を設定したマイクロビット同士の通信が可能になる

#### ③-2 プログラム例

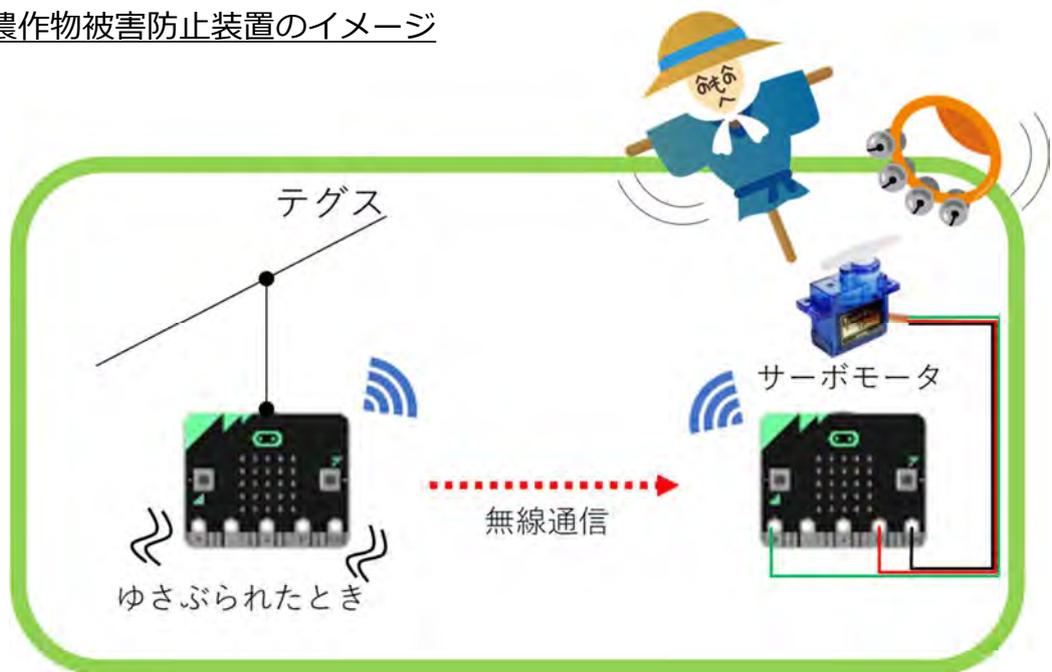


### 3時間目：問題の解決に役立つ計測・制御

#### ①全体の動作をアクティビティ図で整理（指導過程2）

①-1

農作物被害防止装置のイメージ

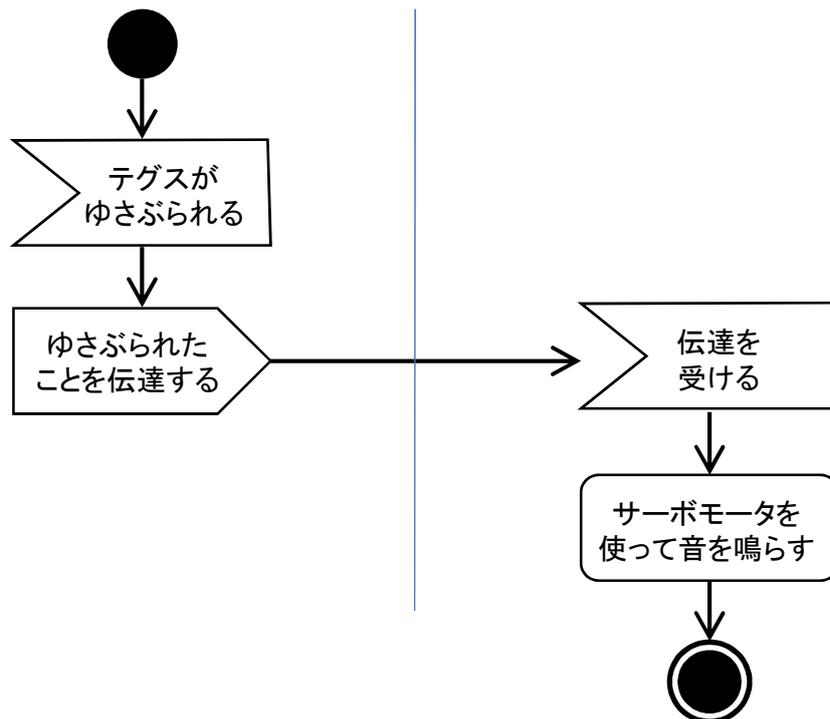


①-2

アクティビティ図の例

送り手側

受け手側



## ②プログラムと製作（指導過程4）

②-1

### エディターを起動

中学校1年生編の2時間目

②-1

②-2

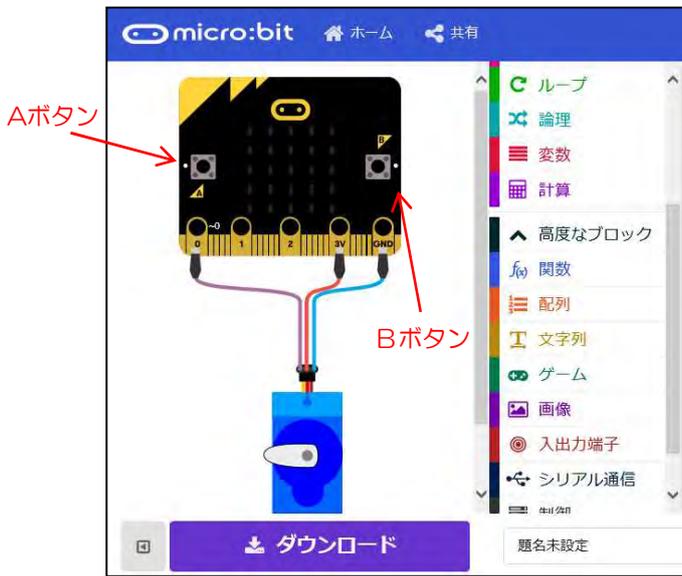
参照

②-2

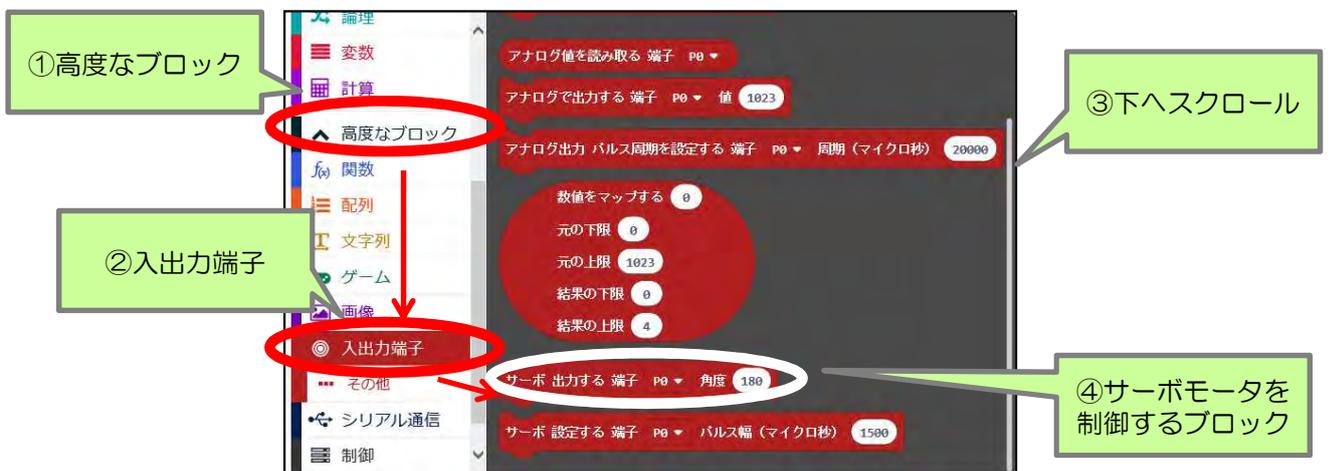
### サーボモータを制御するプログラム

- まずはシミュレータ（PC上）で動作確認

#### 〈プログラム例〉



#### 〈サーボモータを制御するブロックの場所〉



②-3

プログラムのダウンロードと電池ボックスの接続

- USBケーブルでマイクロビットとPCが接続された状態だとサーボモータは動かない

- ①プログラムのダウンロード
  - ②電池ボックスの接続
- 中学校1年生編の2時間目
- ③-2 ③-3 参照

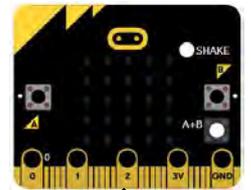
②-4

サーボモータをマイクロビットに接続

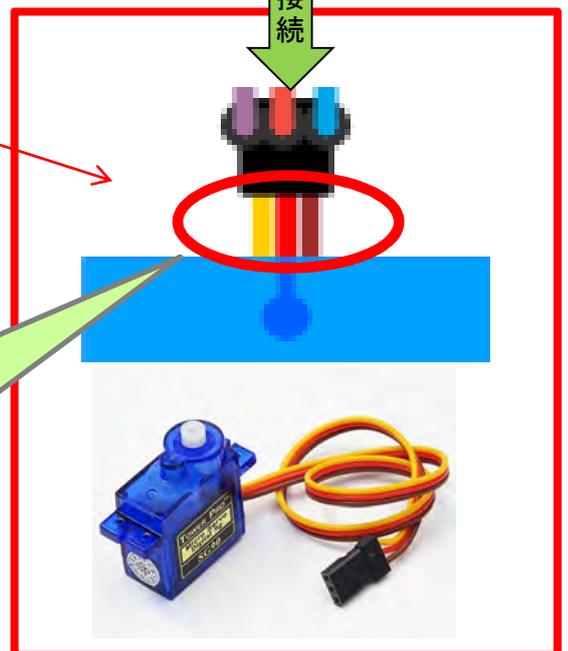
- シミュレータ (PC上) で配線を確認



ジャンプワイヤ (オス〜ワニクチ)



接続



<マイクロサーボの配線>

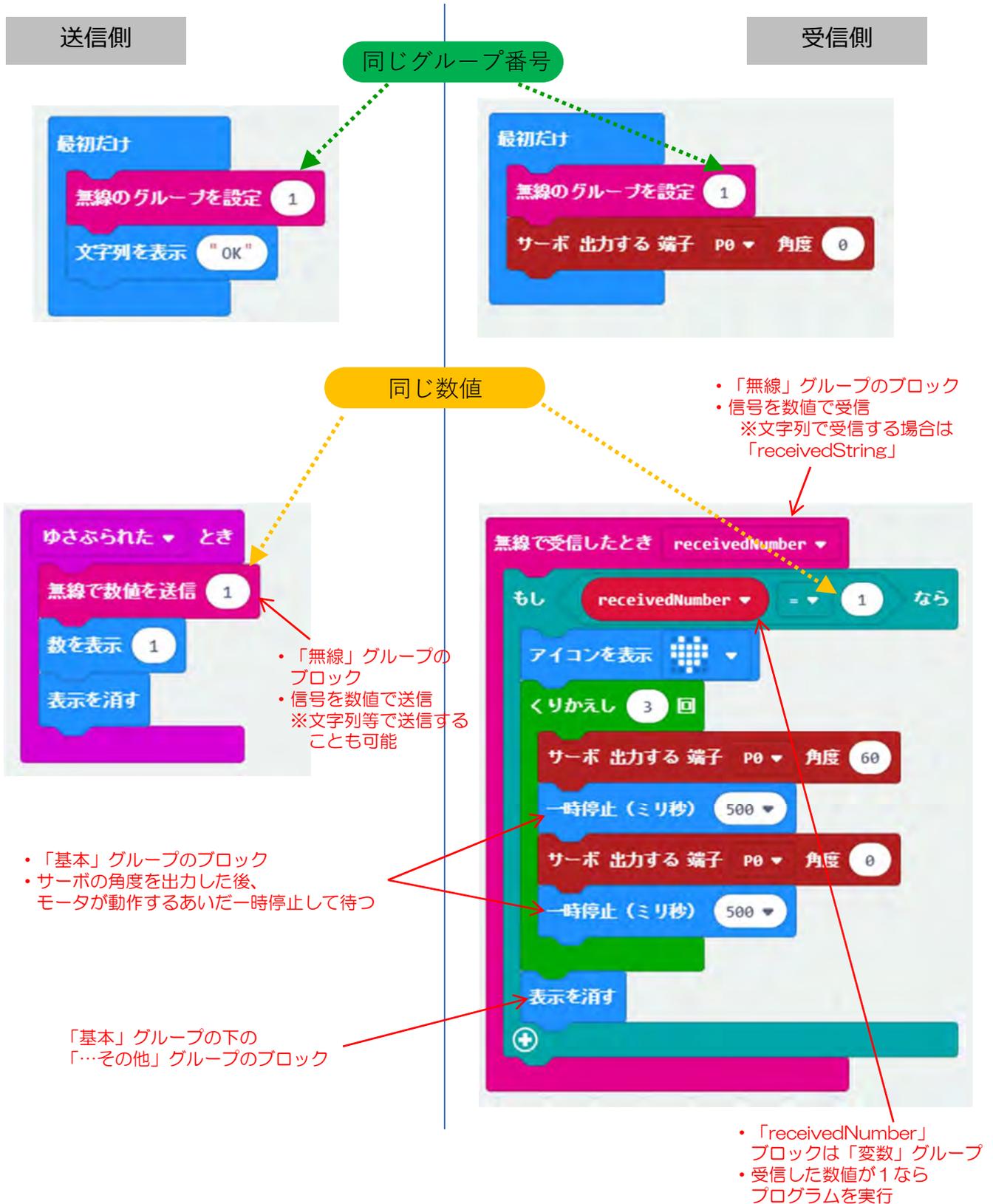
- ・黄色い線はPOへ
- ・赤い線は3Vへ
- ・茶色い線はGNDへ

※その先に接続するジャンプワイヤの色は異なっても問題ない

### ③アクチュエータ側（レシーバ側）の設計とプログラミング（指導過程5）

③-1

“無線通信で”サーボモータを制御するプログラム例



## 〈プログラム例〉



「計算」グループの乱数ブロックを使う

0 から 10 までの乱数

※数値を手入力で変更する

## 4時間目：ロボットを作って動作を確かめる

- 本時で使用するテキスト等



「プログラムによる計測・制御」  
教師用指導案



「プログラムによる計測・制御」  
生徒用ワークブック

### 〈ロボット教材〉



レゴ® マインドストーム® EV3  
※4,5,7,8時間目で使用

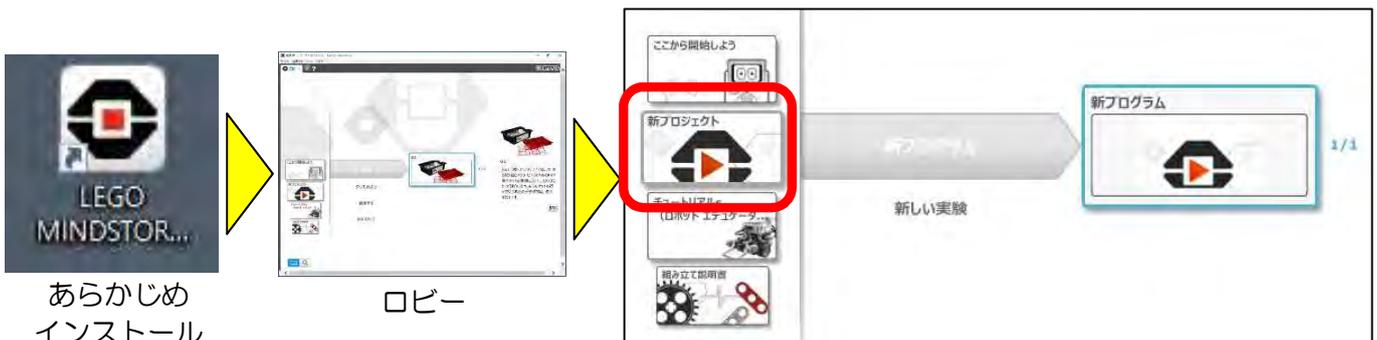
### ①センサーカーの組み立て（事前準備）

- 「プログラムによる計測・制御」教師用指導案付属CD-ROM または MINDSTORMSソフトウェアのロビー内を参照

・センサーカーの組み立てには時間を要するため、授業前に組み立てておくことが望ましい

### ②プログラミング環境（指導過程2）

- 生徒用ワークブック P10-12 参照



### ③パソコンと切り離して走らせる（指導過程4）

- 「プログラムによる計測・制御」生徒用ワークブック P15を参照

(1) EV3インテリジェントブロックの電源が入っていることを確認し、USBケーブルでPCと接続（無線接続も可能）



(2) ハードウェアページで接続が確認できたら、ダウンロードボタンをクリック



(3) EV3から「ピポッ」と音がしたらダウンロード完了

(4) USBケーブルを外し、本体ボタンを操作して実行するプログラムを選択して実行

### ④ワークシートの解答例（指導過程6）

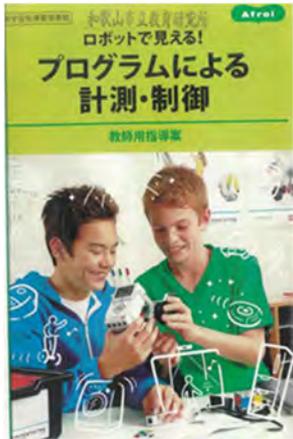
- 教師用ワークブックに記載

### ⑤ワークシートの解答例（指導過程7）

- 教師用ワークブックに記載

## 5時間目：計測と制御のプログラミングでアクチュエータを動かす

- 本時で使用するテキスト等



「プログラムによる計測・制御」  
教師用指導案



「プログラムによる計測・制御」  
生徒用ワークブック

### ①使用できるセンサの説明（指導過程3）

- 「プログラムによる計測・制御」生徒用ワークブック P5

※教育版レゴマインドストームEV3に含まれているセンサ

- ・カラーセンサ
- ・タッチセンサ
- ・ジャイロセンサ
- ・超音波センサ

### ②タッチセンサによるスイッチング（指導過程4）

#### ②-1 タッチセンサの取り付け

- タッチセンサーを入力ポート1に接続

## ②-2 タッチセンサの数値確認

- 指導案付属のワークシート

「様々なセンサとアクチュエータを扱うシステムとコンピュータその1」

### 3. タッチセンサーの数値確認

EV3ブロックでPort Viewを実行し、タッチしたときの値の変化を確認

1. EV3ブロックを起動
2. 左右ボタンで「アプリ画面」
3. 「Port View」を選択
4. 左右ボタンでセンサーのポートを選択



## ②-3 動作の整理とアクティビティ図

・実社会でより多く使われている、タッチされると「停止」するプログラムを作成する

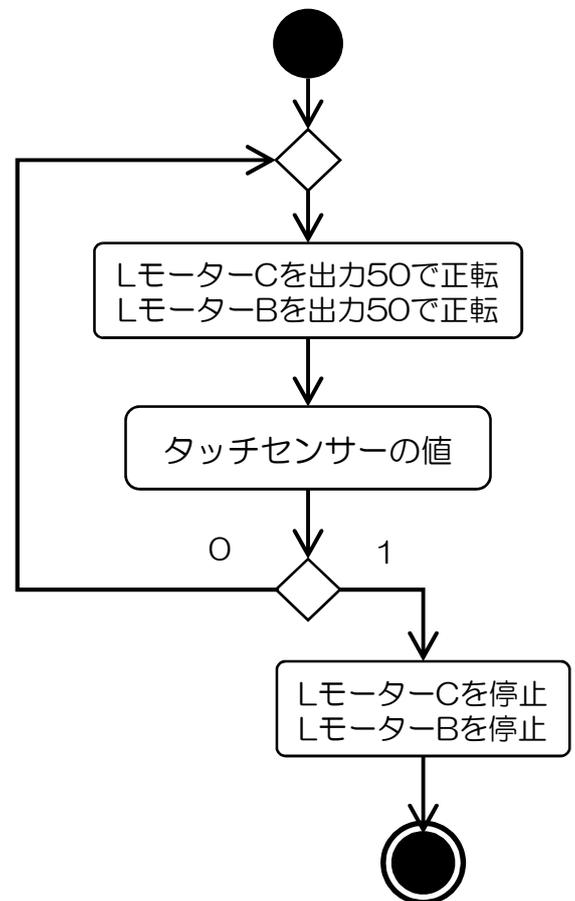
- 指導案付属のワークシート

「様々なセンサとアクチュエータを扱うシステムとコンピュータその1」

### 4. 動作の整理とアクティビティ図

- アクティビティ図の書き方は中学校2年生編の4/20時間目を参照

順番	動作	プログラム
①	タッチセンサーが押されるまで繰り返す	タッチセンサー(ポート1)の値 = 1 まで繰り返す
②	前進する	LモーターCを出力50で正転 LモーターBを出力50で正転
③	停止する	LモーターCを停止 LモーターBを停止



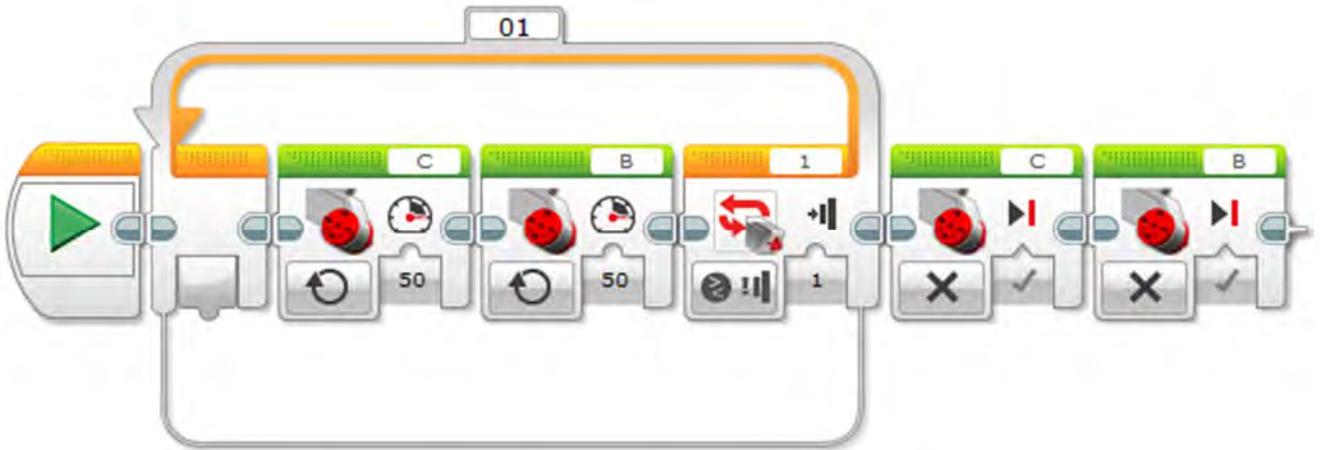
②-4

プログラミング

● 指導案付属のワークシート

「様々なセンサとアクチュエータを扱うシステムとコンピュータその1」

5. プログラミング



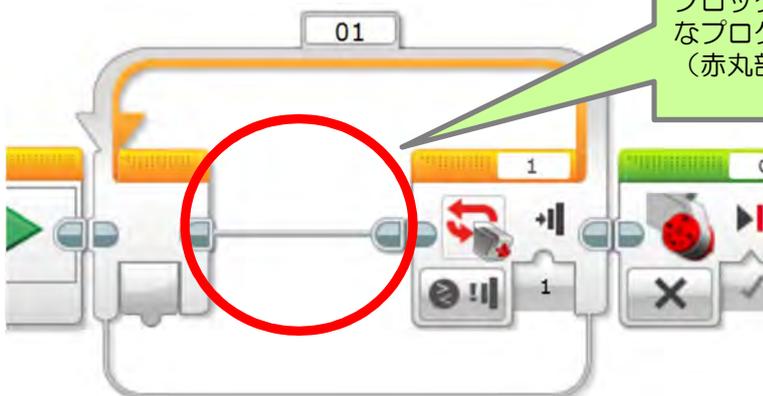
②-5

参考

「…まで待つ」ブロックを使用するプログラムも考えられる。  
ただし、待っている間は、さまざまなプログラムを実行することができない。(ただ待つだけ)



②-4 で示したように「…まで繰り返す」ブロックを使うことで、待っている間もさまざまなプログラムを実行することができる。(赤丸部分にプログラムを追加できる)



### ③超音波センサーの利用（指導過程5、6）

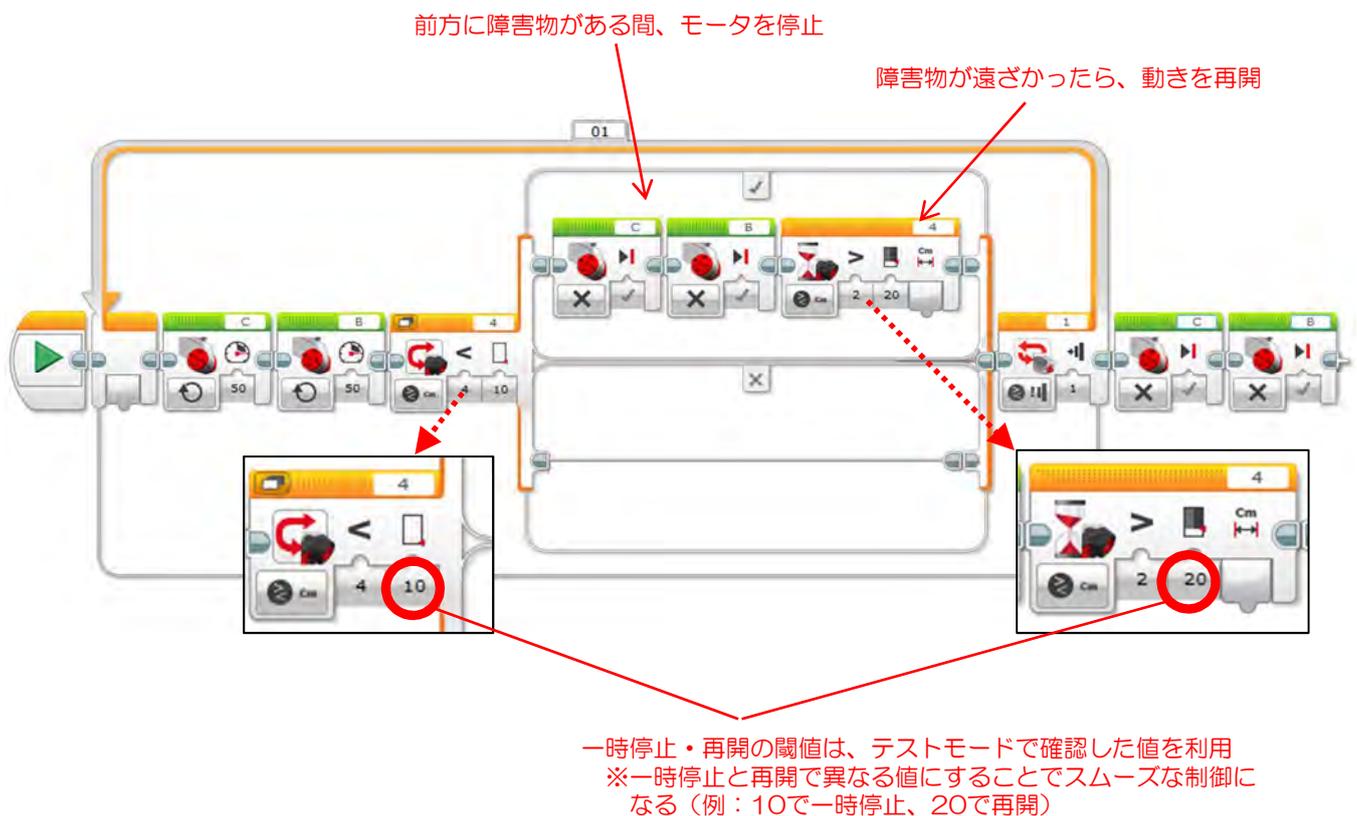
#### ③-1 閾値の決定

- 「プログラムによる計測・制御プログラミング」生徒用ワークブック P30参照

#### ③-2 衝突回避カーのプログラム例

- ・先に作成したタッチでストップするプログラムに、障害物検知で一時停止するプログラムを追加

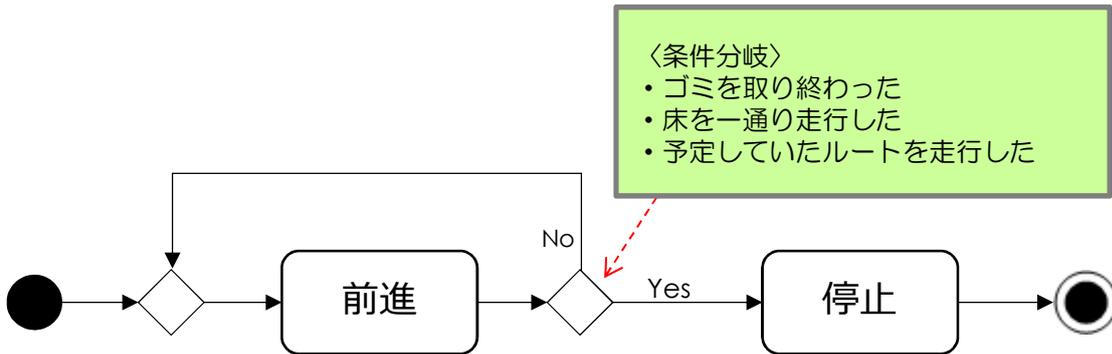
※超音波センサーをポート4に接続した場合



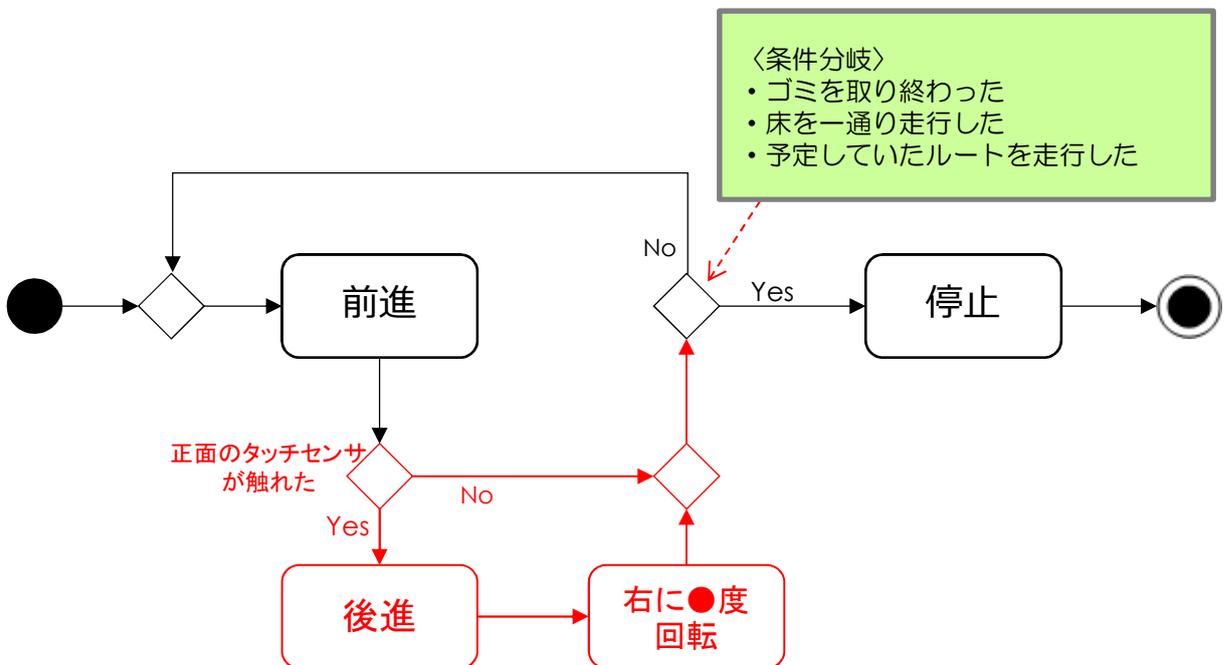
①業務用掃除ロボに求められる課題を図示（指導過程2）

- アクティビティ図の書き方は中学校2年生編の4/20時間目を参照

〈掃除ロボの基本動作〉



〈壁にぶつかったら後進、旋回、を追加〉 ※赤色部分を追加



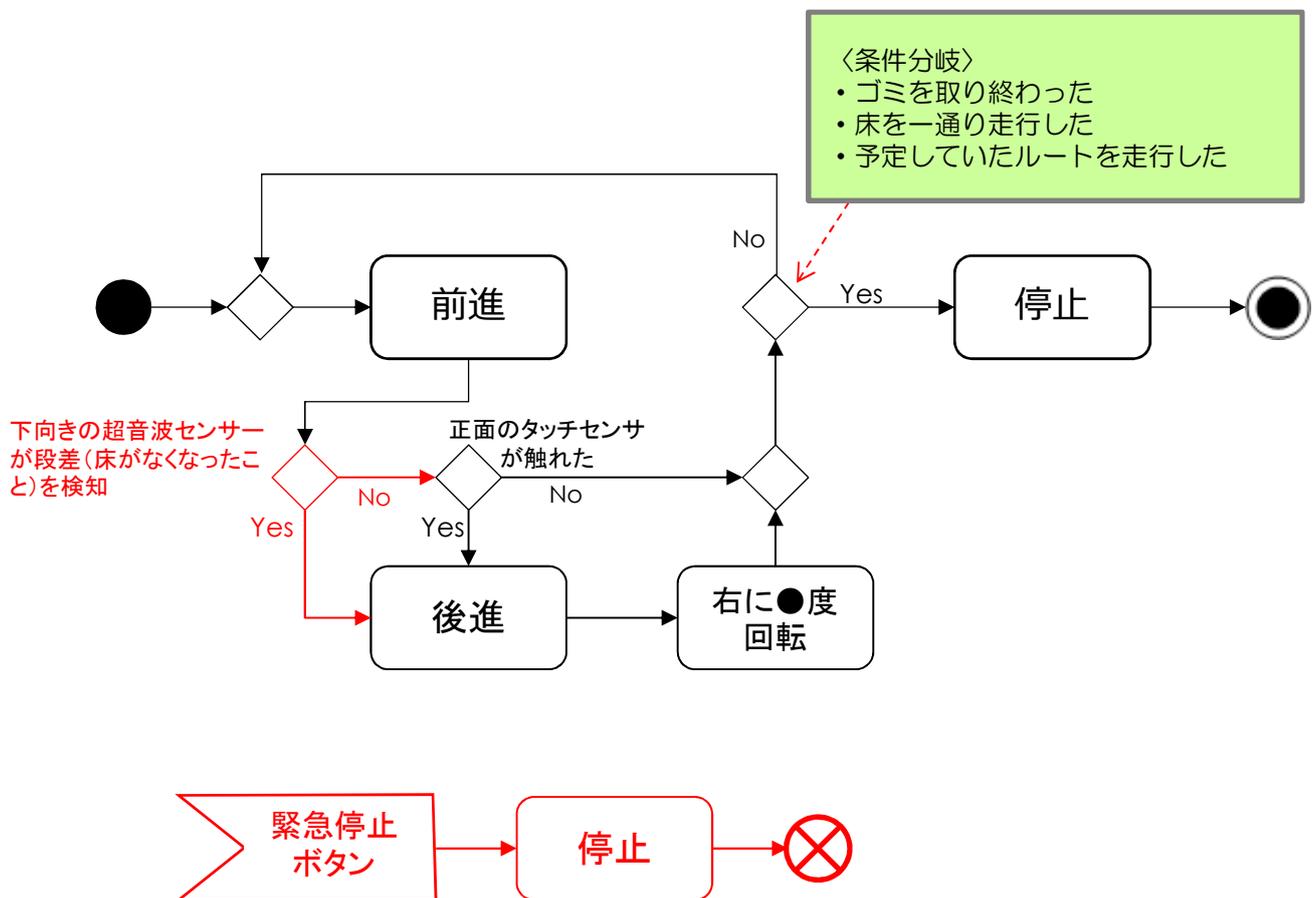
## ②どのような課題があるか（指導過程3）

※教育用MINDSTORMS EV3に含まれているセンサ

- タッチセンサー
- 超音波センサー
- カラーセンサー
- ジャイロセンサー

## ③手順分解やアルゴリズム（指導過程4）

〈凹みの段差検知、緊急停止ボタンを追加〉 ※赤色部分を追加



## 7,8時間目：プログラミング、制作

### ①プログラム例（カラーセンサーやタッチセンサーの利用）

- ライントレースカー

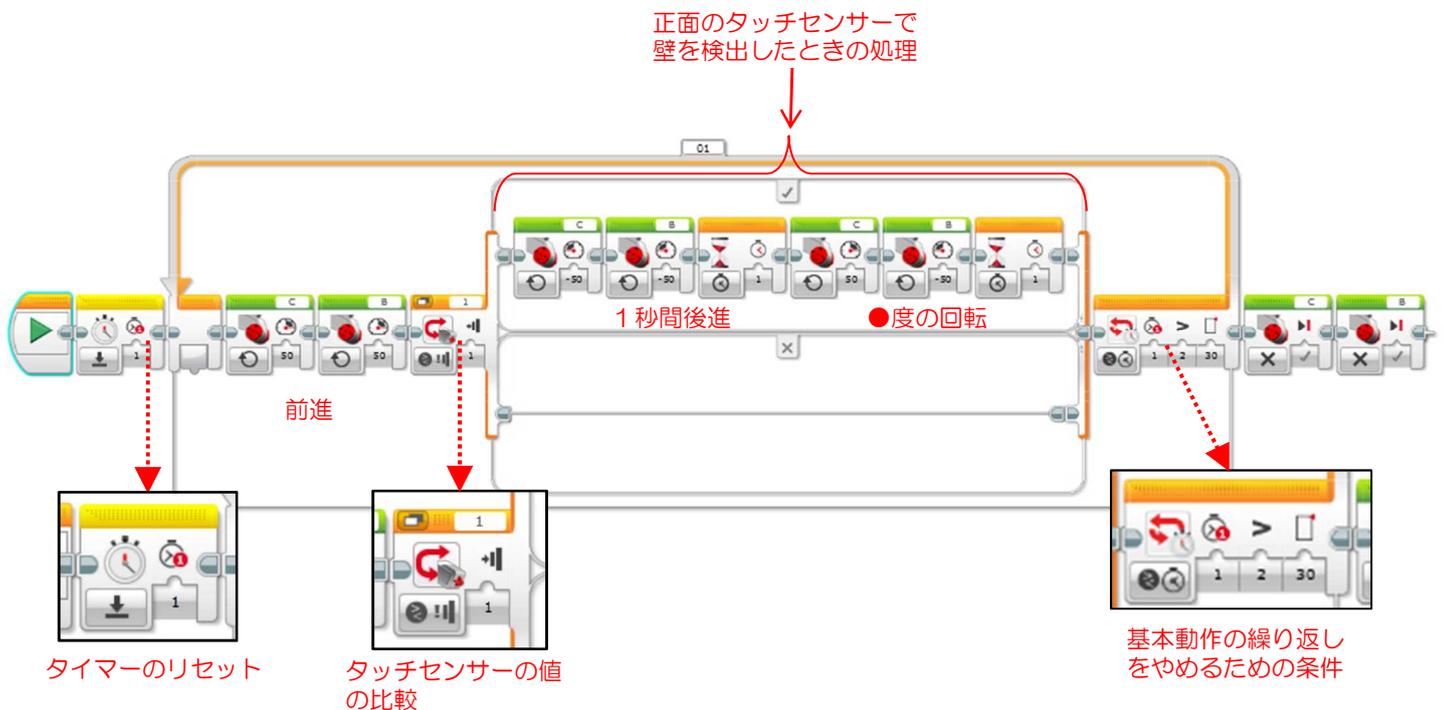
→ 生徒用ワークブックP24～29参照

- 掃除ロボ

※基本動作（ここでは30秒前進して止まる）の中で、正面のタッチセンサー（ポート1）で壁を検知したら後進・旋回



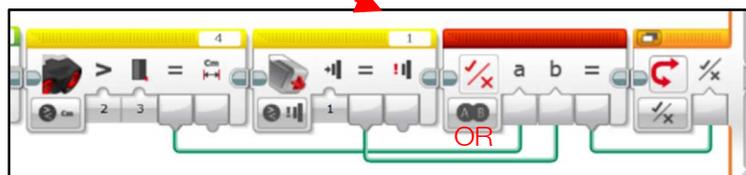
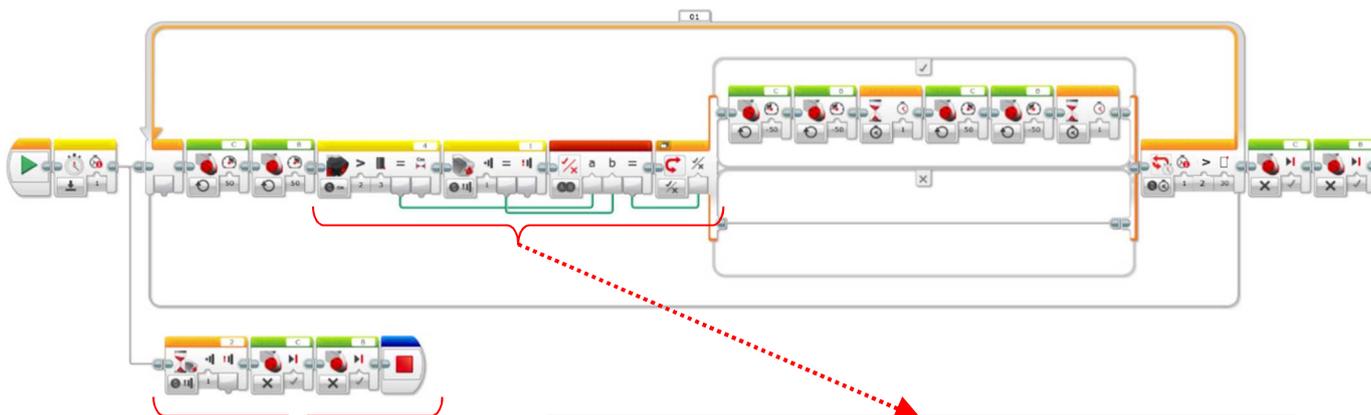
「プログラムによる計測・制御」  
生徒用ワークブック



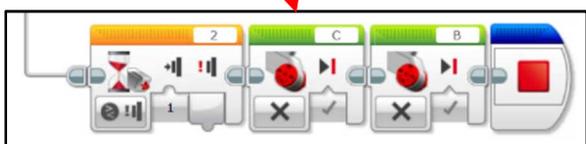
## ②プログラム例（超音波センサー + タッチセンサー×2）

- 前ページの掃除ロボに、段差検知機能と緊急停止ボタンを追加

段差検知：下向きにつけた超音波センサー（ポート4）の値が閾値より大きい  
緊急停止ボタン：タッチセンサー（ポート2）が押されたら停止



段差を検知したときにも後進・旋回を行うように、ロジック操作ブロックを使って、  
（超音波センサーが閾値より大きい）または（タッチセンサーが1）を表している



緊急停止ボタンが押されていたときに  
モーターを停止し、プログラムの実行  
をやめる  
基本動作と並列に実行

## 9時間目：より効率の良いプログラミングの技術（テキストプログラミングの良さ）

### ①課題把握（指導過程1）

①-1

世界で最も使用されているプログラミング言語

- <https://www.benfrederickson.com/ranking-programming-languages-by-github-users/>

#### 〈主なプログラミング言語〉

- ・ JavaScript (ジャバスクリプト)
- ・ Python (パイソン)
- ・ Java (ジャバ)
- ・ C++ (シーplusplus)
- ・ C (シー)
- ・ PHP (ピーエイチピー)
- ・ C# (シーシャープ)
- ・ Shell (シェル)
- ・ Go (ゴー)
- ・ TypeScript (タイプスクリプト)

上位幾つかの言語を紹介し、その言語名を検索エンジンで画像検索し、全てが文字で入力されている言語であることを見せる。

### ②マイクロビットのエディタの起動（指導過程2）

②-1

ブラウザ（インターネットエクスプローラなど）を立ち上げる



インターネット  
エクスプローラ



グーグル  
クローム



マイクロソフト  
エッジ

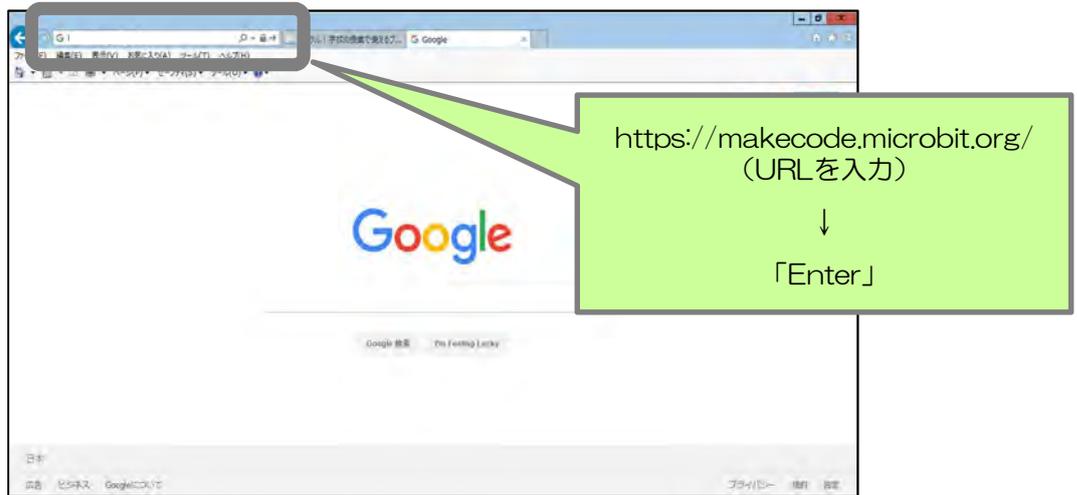


※アイパッドの場合は  
マイクロビットのアプリの  
インストールが必要

②-2

マイクロビットのエディタへ

※Googleの例



②-3

プログラムを組む



### ③テキスト言語との対比（指導過程3）

③-1

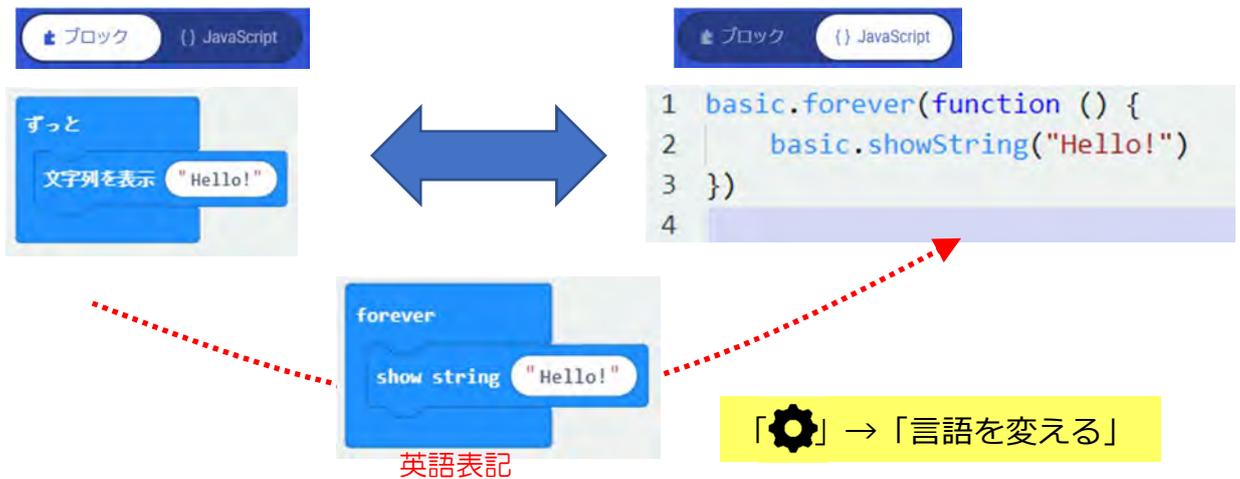
エディタを「 {} JavaScript」（ジャバスクリプト）に切り替える



②プログラムがテキスト型言語に切り替わる

```
1 basic.forever(function () {
2   basic.showString("Hello!")
3 })
4
```

※英語のブロックを経由しても良い



#### ④ サンプルプログラムの実行 (指導過程 4)

サンプルプログラムは、教育センター学びの丘の「きのくにeラーニング」システムの「きのくにICT教育（中学校）」に掲載

中3年9時間目演習配布テキストプログ...

① サンプルプログラムをパソコンに保存しておく（保存場所は任意）

開く

```
let y = 0
let pi = 0
let a = 0
input.onButtonPressed(Button.AB, function () {
  basic.showString("")
  y = a * pi
  basic.showNumber(y)
})
input.onButtonPressed(Button.A, function () {
  a += 1
  basic.showNumber(a)
})
input.onButtonPressed(Button.B, function () {
  a += -1
  basic.showNumber(a)
})
a = 5
pi = 3.14
basic.showNumber(a)
```

② 全選択 (Ctrl+A) して  
コピー (Ctrl+C)

③ エディターをJavaScriptモードにする

④ BackSpaceキーですべて削除

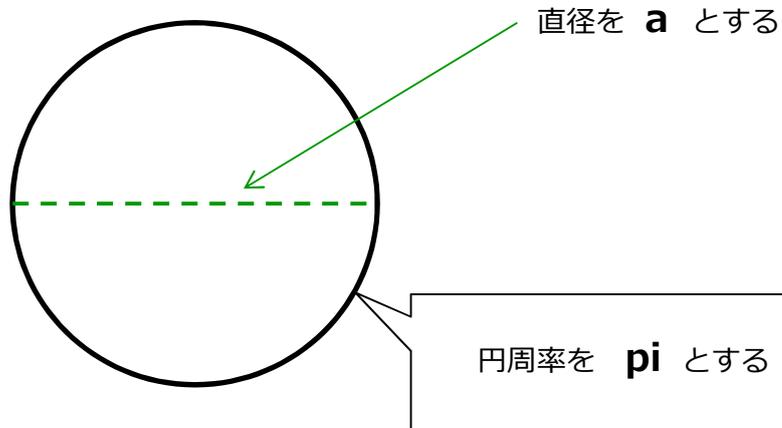
⑤ 貼り付け (Ctrl+V)

※生徒のパソコン操作の習熟度に合わせて、コピー・貼り付けを以下の方法としても良い  
マウスでプログラムをドラッグ ⇒ 右クリック「コピー」⇒「貼り付け」

#### ④テキストプログラミングで数式を変更（指導過程4①）

④-1

円周を求めるプログラム（ブロックプログラミング）



最初だけ

- 変数 **a** を **5** にする ← a (直径) を 5 にする
- 変数 **pi** を **3.14** にする ← pi (円周率) を 3.14 にする
- 数を表示 **a** ← a (直径) を表示

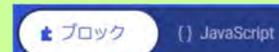
ボタン **A** が押されたとき

- 変数 **a** を **1** だけ増やす ← Aボタンが押されたら、a (直径) を 1 増やす
- 数を表示 **a** ← 1 増えた後の a (直径) を表示

ボタン **B** が押されたとき

- 変数 **a** を **-1** だけ増やす ← Bボタンが押されたら、a (直径) を 1 減らす
- 数を表示 **a** ← 1 減った後の a (直径) を表示

エディターをブロックモードにするとブロックプログラムが表示される



ボタン **A+B** が押されたとき

- 文字列を表示 **〇** ← 何もしない状態では、a(直径)が表示されているので、一度表示を消す
- 変数 **y** を **a** × **pi** にする ← 円周を求める計算結果 [ a(直径) × 円周率 ] を変数 **y** に入れる
- 数を表示 **y** ← 変数 **y** (円周) を表示

```
1 let y = 0
2 let pi = 0
3 let a = 0
4 input.onButtonPressed(Button.A, function () {
5     a += 1
6     basic.showNumber(a)
7 })
8 input.onButtonPressed(Button.B, function () {
9     a += -1
10    basic.showNumber(a)
11 })
12 input.onButtonPressed(Button.AB, function () {
13    basic.showString("")
14    y = a * pi ← 円周を求める計算式
15    basic.showNumber(y)
16 })
17 a = 5
18 pi = 3.14
19 basic.showNumber(a)
20
```

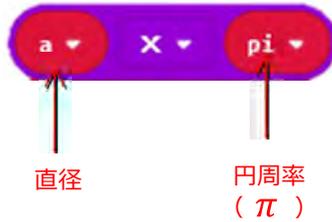
テキスト型プログラミングで“円の面積”を求める計算式に変更する

$$y = (a / 2) * (a / 2) * pi$$

## ⑤ブロックプログラミングとテキストプログラミングを比較（指導過程4②）

### A 円周を求める計算式

$$y = a * \pi$$

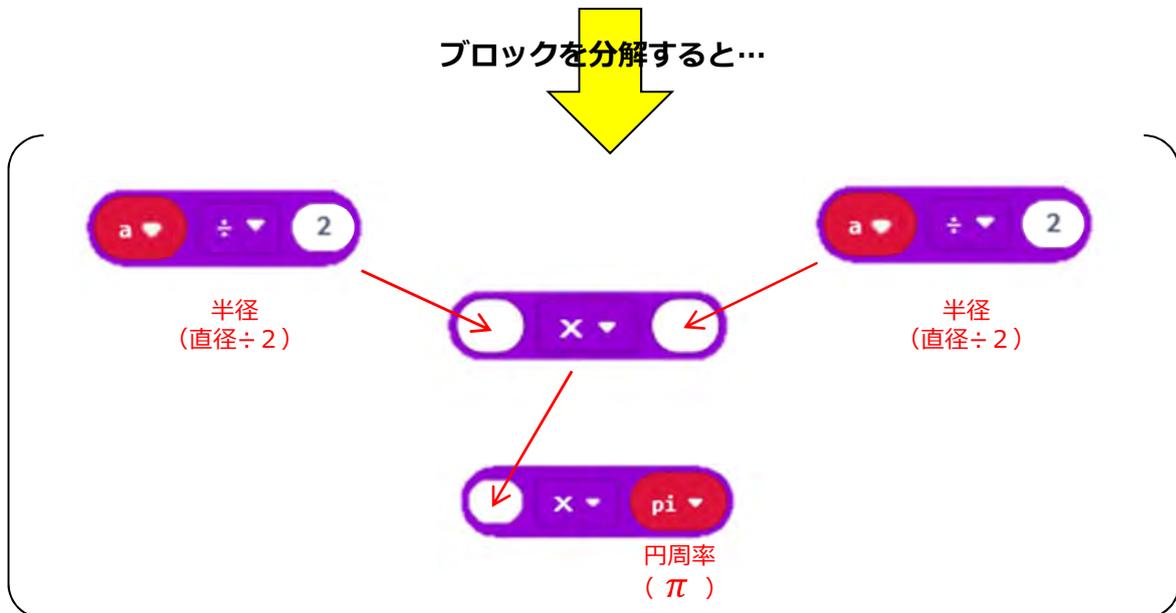


### B 円の面積を求める計算式

$$y = (a / 2) * (a / 2) * \pi$$

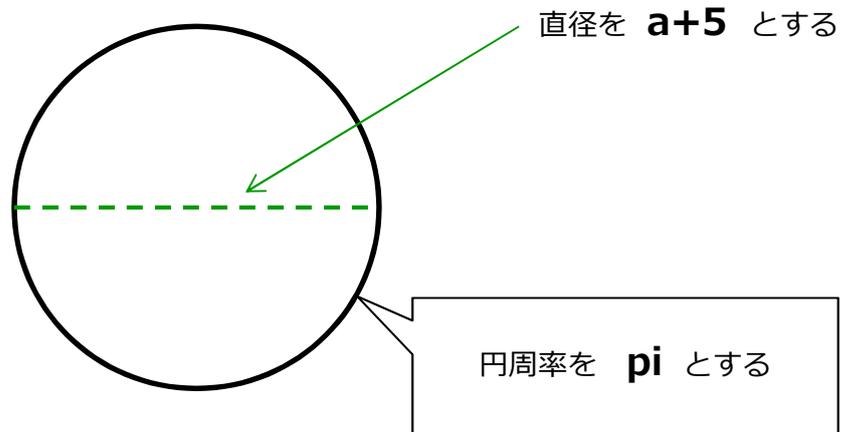


ブロックを分解すると…



ブロックプログラミングでA（円周を求める計算式）からB（円の面積を求める計算式）に変更するには、時間と手間がかかることが分かる

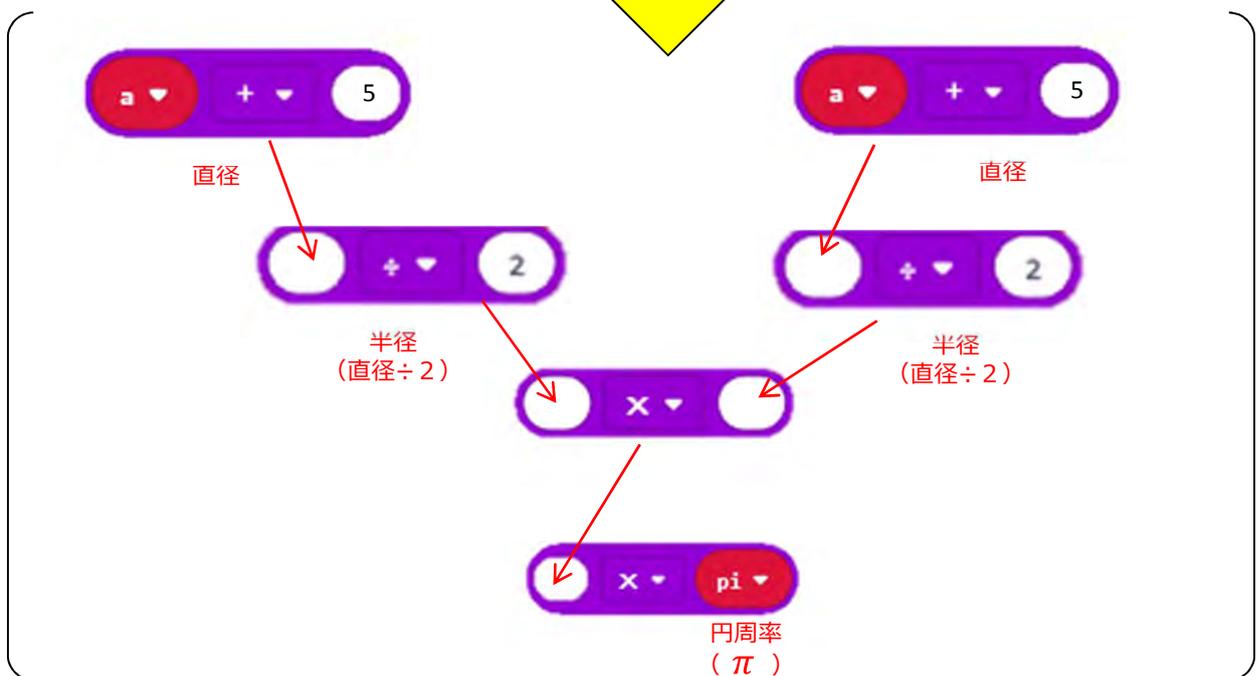
## ⑥ブロックプログラミングで数式を変更（指導過程4③）



### ● 「直径 (a+5)の円」の面積を求める計算式



ブロックを分解すると…



## ⑦ブラウザのソース表示（指導過程 6）

① Webサイトを開き「表示」→「ソース」

② ソースが表示される

## ⑧ロボット制御のプログラム（参考）

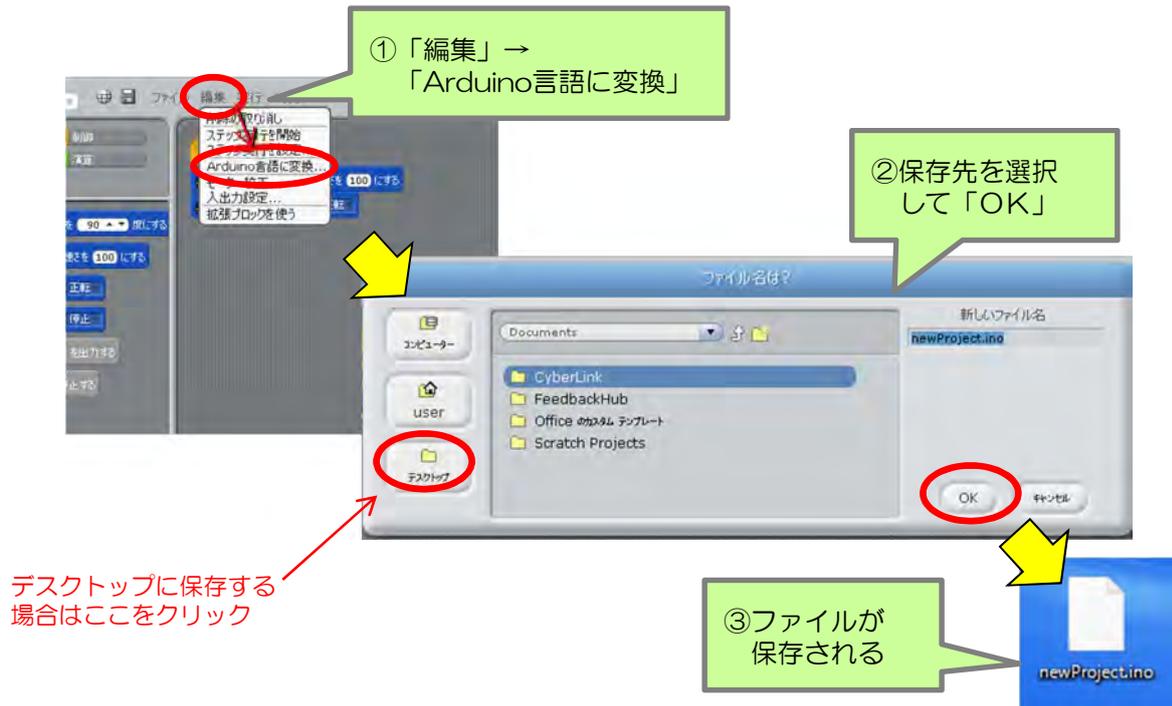
⑧-1 studuino（スタディーノ）をロボットモードで起動

あらかじめインストール

⑧-2 プログラムを作成

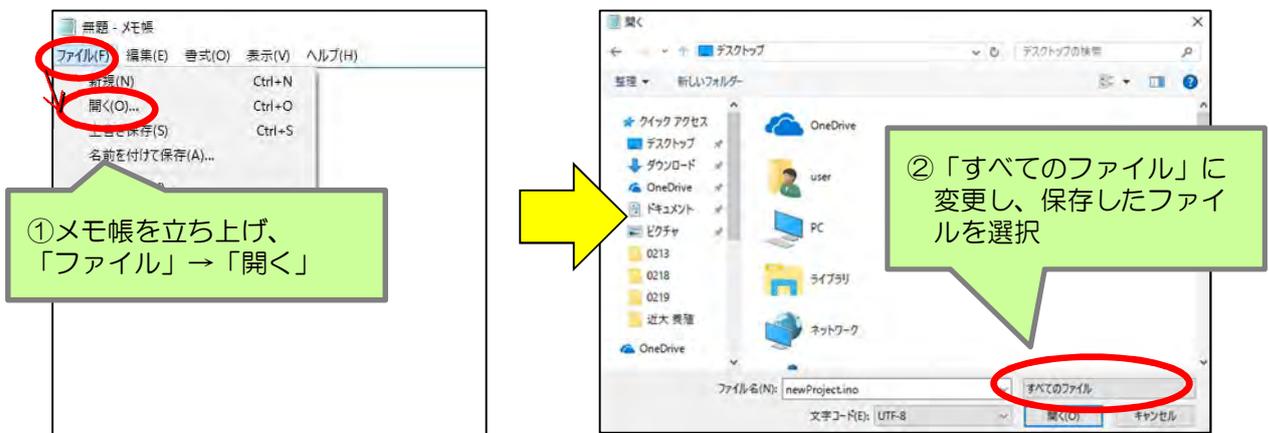
⑧-3

テキスト型言語にして保存



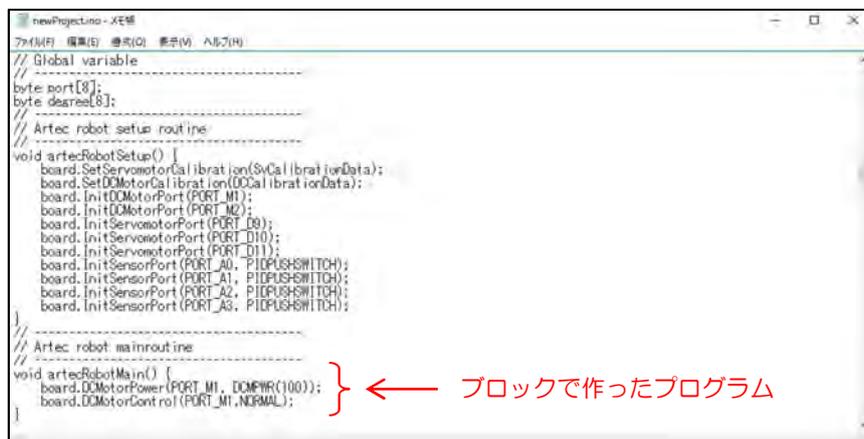
⑧-4

メモ帳で開く



⑧-5

プログラムを見る



# 県内事例集

● 技術分野では、単に何かをつくるという活動ではなく、例えば、技術に関する原理や法則、基礎的な技術の仕組みを理解した上で、生活や社会の中から技術に関わる問題を見いだして課題を設定し、解決方策が最適なものとなるよう設計・計画し、製作・制作を行い、その解決結果や解決過程を評価・改善し、さらにこれらの経験を基に、今後の社会における技術の在り方について考えるといった学習過程を行うことが大切です。

● この事例集では、本県の先進的な事例を6つ集めました。それぞれの学校や地域の実態に応じ、参考にしつつ授業を行ってください。

## 双方向

〔 ネットワークを活用した双方向性のあるコンテンツのプログラミングによる問題の解決 〕

- 事例1：遠隔操作による鳥獣被害対策
- 事例2：遠隔医療（テレビ会議システムを利用した遠隔カンファレンス）

## 計測・制御

〔 計測・制御のプログラミングによる問題の解決 〕

- 事例3：地震・津波観測監視システム DONET（ドゥーネット）
- 事例4：農業ハウス環境の自動制御システム
- 事例5：航空レーザ測量による森林資源情報の整備
- 事例6：稚魚自動供給システム

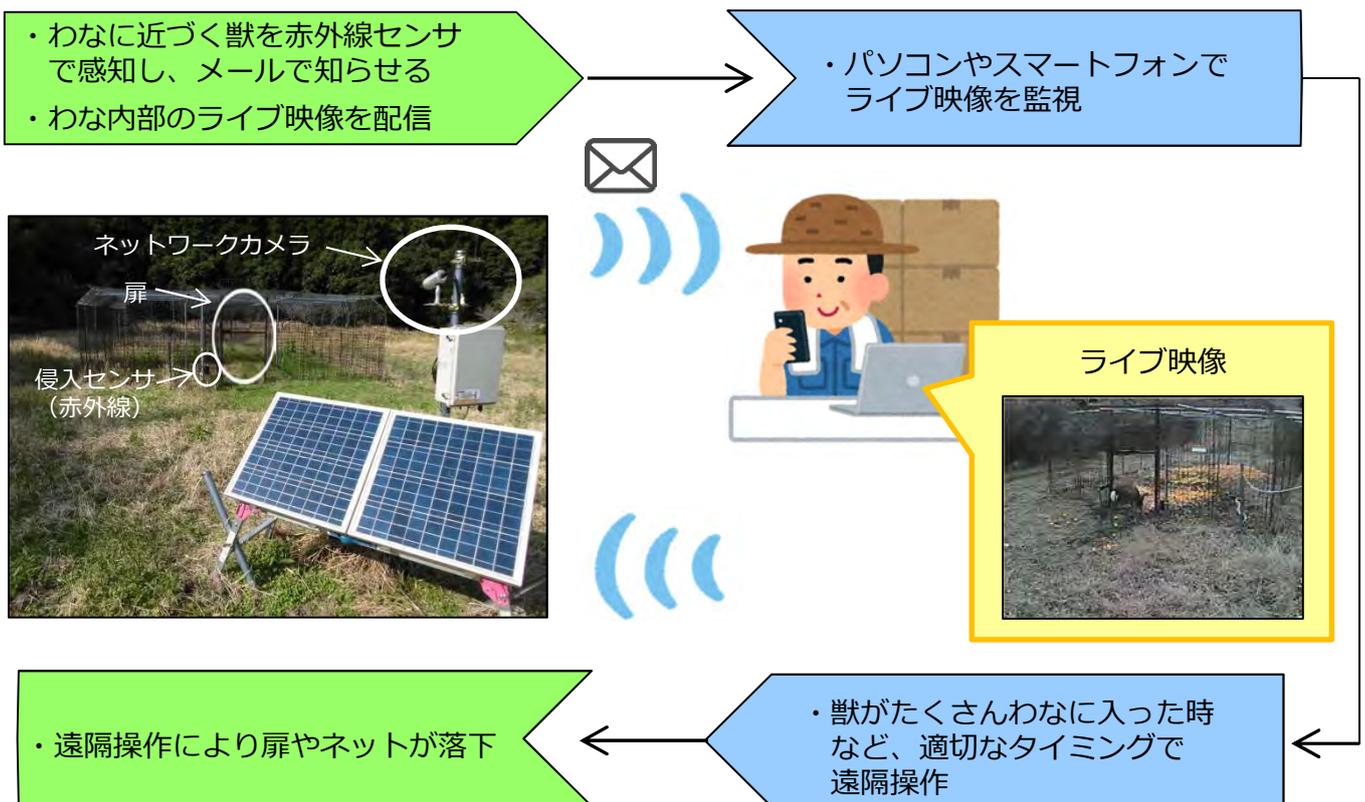
①課題

- 本県では、イノシシ、シカ、サル、アライグマなどにより、年間3億円を超える農作物被害が継続して発生しています。
- 網、わな、銃猟などにより対策を図っていますが、見回り・監視による体力的負担や大量捕獲が困難なことなど、課題があります。

②解決策

- ICTを活用した大型捕獲機の導入

センサーやカメラを搭載するとともに、離れた場所にあるパソコンやスマートフォンと通信できる大型檻や罠が導入されています。



③効果

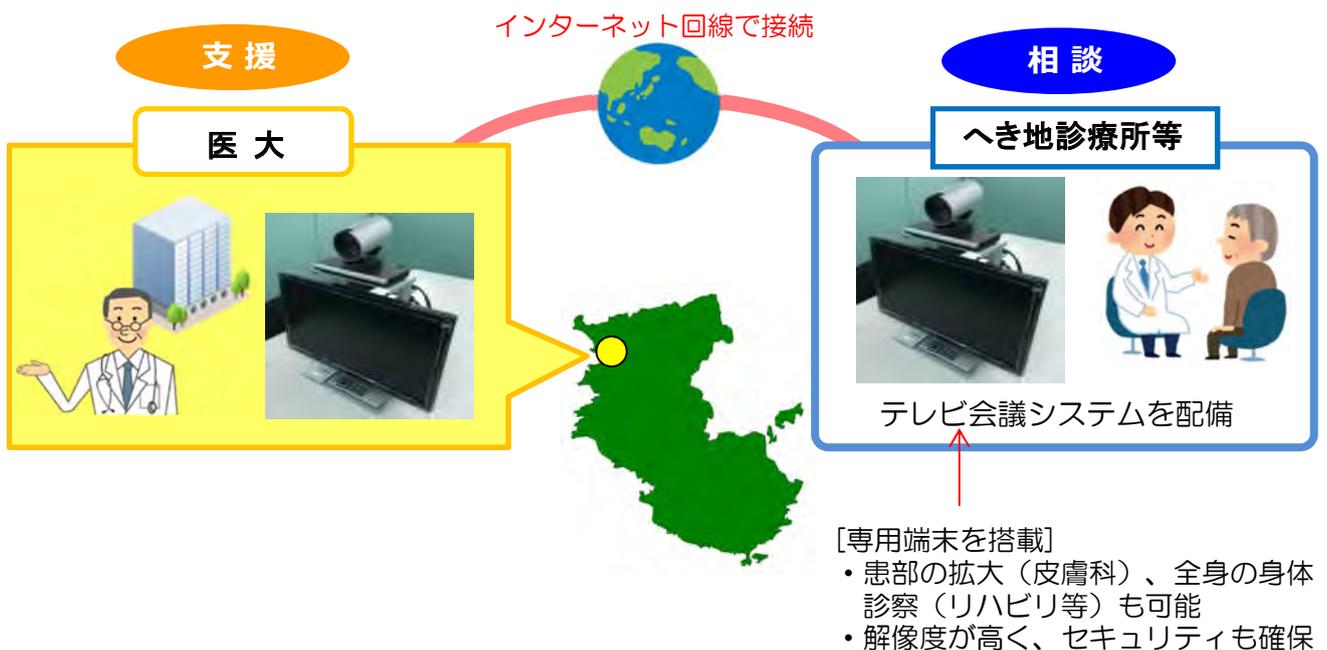
- 獣の映像を遠隔地で見ながら捕獲することで、監視などの体力的負担の軽減や効率的(大量)な捕獲が出来るようになりました。また、エサの置き方などの改善も可能になりました。

①課題

- 地方の人口減少などを背景に、医師が都市部に偏在するとともに診療科別の偏在も深刻になるなど、多くの地域で必要な医師の確保ができていません。
- 患者からすると、いつでも良質な医療サービスを受けられるという信頼感が揺らいでいます。

②解決策

- 遠隔医療体制の構築
  - ・ 平成26年7月から、県立医科大学附属病院（医大）と県内中核病院において、テレビ会議システムを活用した遠隔カンファレンスが開始されています。
  - ・ 平成29年度からは、へき地診療所にも配備し、医大専門医への相談体制を構築することにより、へき地診療所でも診療支援を行っています。



- 診断の質の向上に向けた取り組み
  - ・ 現在、2020年以降の実用化を目指して、次世代通信規格「5G」を活用した遠隔医療の実証実験を行っています。鮮明なエコー動画や患部画像の送受信により、診断の質の向上が期待されています。

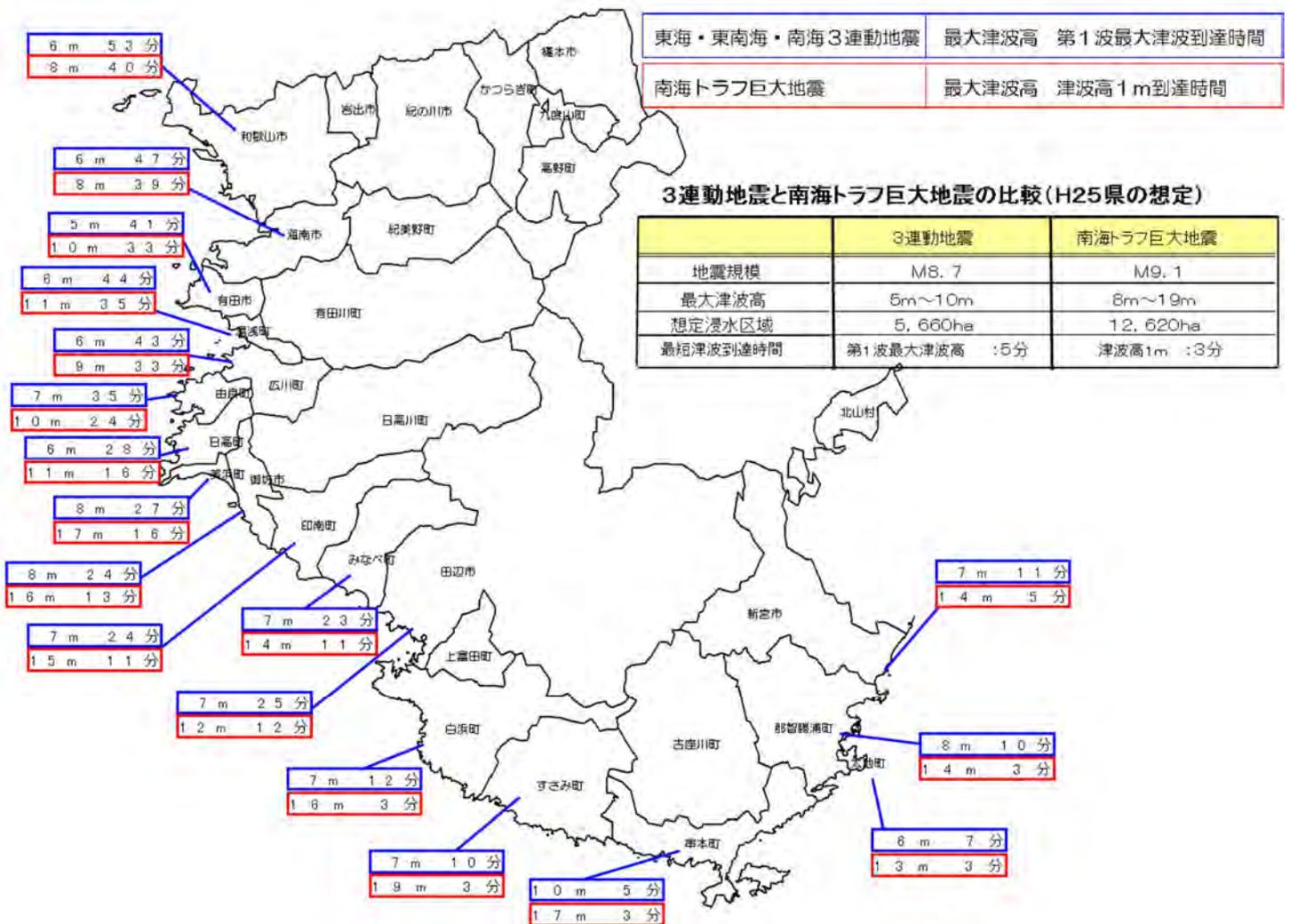
③効果

- 患者にとって、住み慣れた地域で専門医の助言を受けることができ、遠方の医療機関への通院負担の軽減につながっています。  
(皮膚科や神経内科（認知症）、栄養指導や言語リハビリテーションなどで活用されています)

①課題

- 近い将来、発生が懸念されている南海トラフ地震の震源域は紀伊半島に近く、激しい揺れと短時間で津波が到達することが想定されています。
- 津波到達までの時間が短いことから、津波から逃げ切ることが難しい地域（津波避難困難地域）が存在します。

和歌山県の津波浸水想定（最大津波高・平均浸水深・到達時間）



## ② 解決策

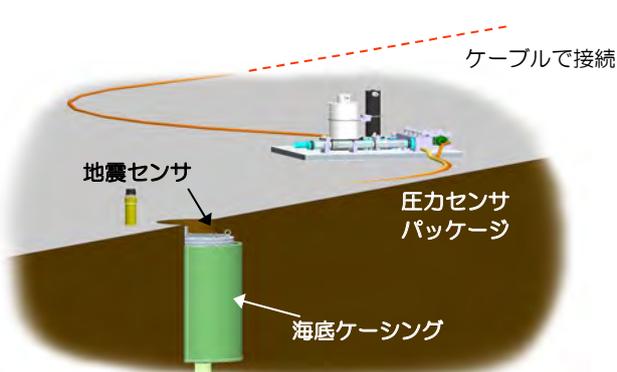
### ● 津波予測システムの開発

- 和歌山県では、国立研究開発法人「防災科学技術研究所（NIED）」所管の「DONET（ドゥーネット）」による地震津波観測網を活用して、津波の規模などを予測できる津波予測システムを国立研究開発法人「海洋研究開発機構（JAMSTEC）」と共同開発しました。

### DONET（ドゥーネット）

- 紀伊半島沖合の海底に設置された地震・津波観測監視システム

#### <DONETイメージ（海底）>



#### <DONET観測点>



#### <DONETのセンサ>

##### <地震センサ>

- ◇強震計：強い地震動を記録
- ◇広帯域地震計：地震の速い振動から非常にゆっくりとした振動まで広い周波数範囲にわたって地震動を記録

##### <圧力センサ>

- ◇水圧計等
- ◇ハイドロフォン：水中の音を記録
- ◇温度計

### 津波予測システム

- DONET観測点からの水圧計などのデータをリアルタイムで伝送し、データ解析ソフトを使って予測対象区域の第一波到達予想時刻と最大予想津波高、浸水エリア予測、浸水深予測を即時に表示

## ③ 効果

### ● 避難のための情報提供

- 和歌山県では津波予測システムを使って、津波の規模や到達予測、浸水予測をいち早く把握し、避難のための情報提供や被害予測を行っています。

#### 県民・県内滞在者



津波からの避難を促すエリアメール・緊急速報メールを配信

#### 市町村・消防本部

予報データを「リアルタイム地震・津波関係情報表示システム」へ表示



※気象庁でも津波到達予報を行っています。これは、地震の観測データから津波の到達時刻などを予測したもので、浸水予測はありません。

①課題

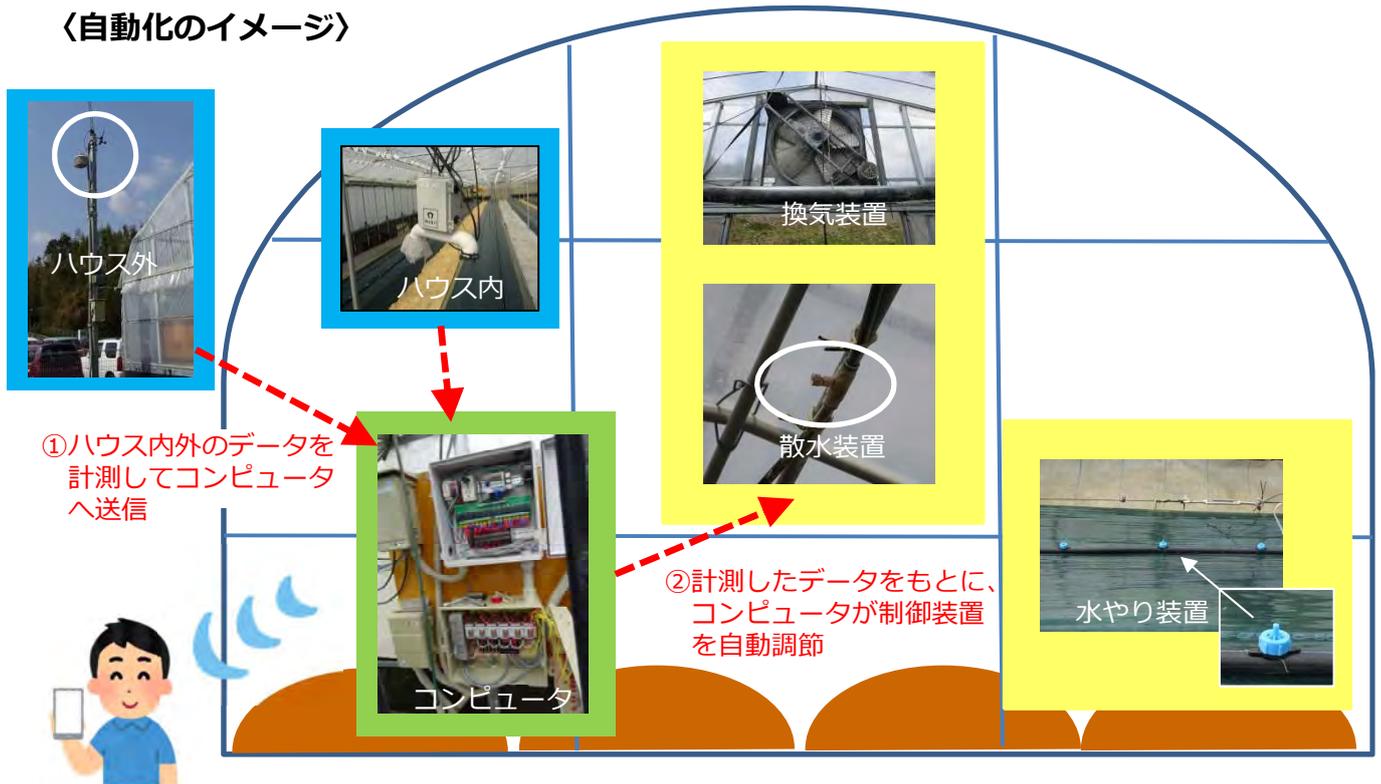
- 本県では、果樹や野菜、花きなど多くの農産物が生産されています。
- ビニルハウスでも盛んに生産されていますが、ハウス内の環境を適切に保つために、温度計・湿度計のチェックや換気・水やりなどに労力を要したり、人によってハウス内環境にばらつきがあるなど課題があります。



②解決策

- ハウス内環境の自動調節
  - ①ハウス内外のさまざまなデータを計測します。  
(ハウス内の温度・湿度、ハウス外の温度・湿度・雨量・日射量など)
  - ②計測したデータをもとに、コンピュータによって水やりや温度調節などが自動で行われ、ハウス内の温度や湿度などが適切に保たれています。

〈自動化のイメージ〉



- 自動調節の状況はスマートフォンやパソコンで確認することができます。換気装置や散水装置などを遠隔操作することも可能です。

③効果

- ハウス内環境調節の自動化により、体力的負担が軽減されるとともに、きめ細かな環境調節が可能になり、品質が良くなったり、育成不良が減少しています。
- ハウス内環境の「見える化」により、今後の対策が立てやすくなっています。

## ①課題

- 本県では林業が盛んですが、林業に必要な森林の情報は人が直接山に入って調べています。
  - ・ 生えている木の種類、太さ、高さなど
- 人が調べる方法では、調査する人によって精度にバラつきが出てしまいます。また、全ての森林を測ることができないので、サンプル調査になってしまいます。



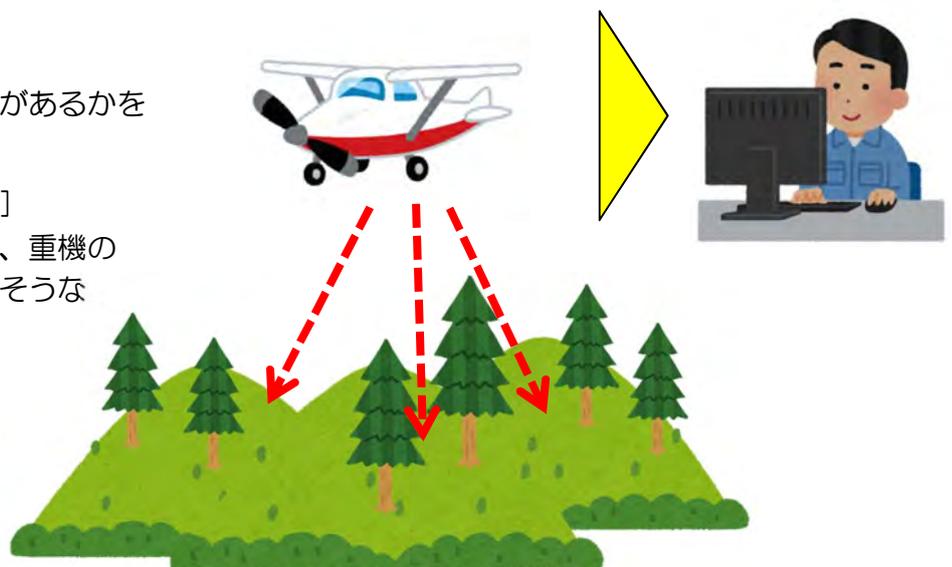
## ②解決策

- 航空レーザー測量※による新たな森林調査手法の導入が進められているところです。
  - ・ 全ての森林の調査が可能
  - ・ 精度の高い調査が可能
  - ・ 土地の細かな形状も同時に計測可能

※航空機に搭載したレーザースキャナから地上に向けてレーザ光を照射し、森林の上層と地上からの反射波との時間差より得られる情報などから、木の頂点の数や高さを割り出したり、地上の標高や地形の形状を求める測量方法

- 収集したデータの解析
  - [効率的な林業]
    - ・ どこにどれくらい森林資源があるかを解析して「見える化」
  - [治山・林道計画の作成支援]
    - ・ 地形（傾斜等）データから、重機の入りやすさや災害が起こりそうな危険な箇所などを予測

## データ解析



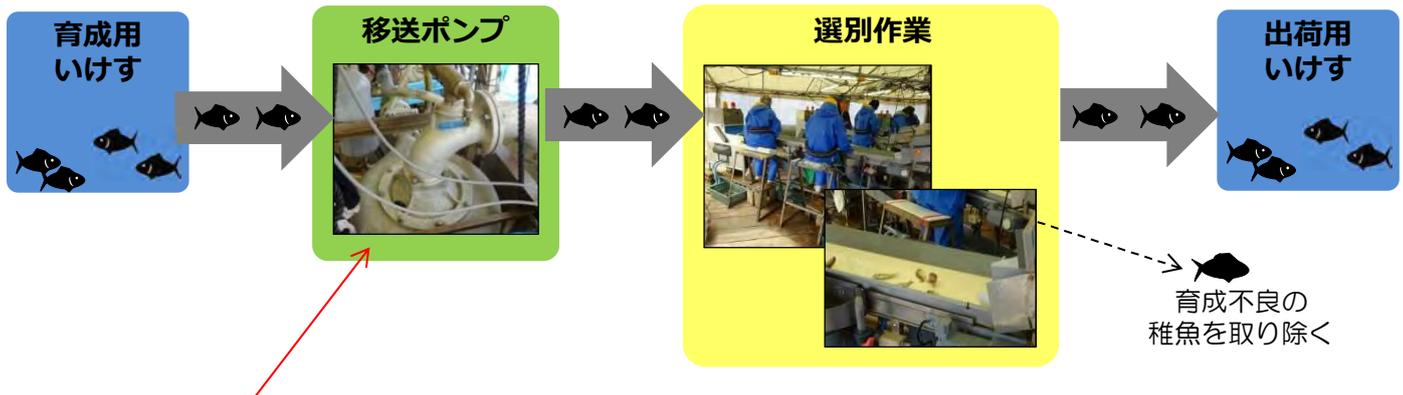
## ③効果

- 森林調査をする人の負担が少なくなるとともに、コストも抑えられることが期待されます。
- 解析したデータを活用して、効率的な林業を行うことができるようになります。

①課題

- 近畿大学では、養殖稚魚を全国の養殖業者へ出荷する前に、専門作業員による選別作業を行い、育成不良のものを取り除くなど基準を満たす魚だけを選び分けています。

〈稚魚選別の流れ〉



- 作業を効率的に行うためには、選別作業員の前を流れる稚魚が多すぎたり少なすぎたりしないように、**ポンプの流量調節**を行う必要があります。
- 流量調節は、**流れる魚の量を絶えず目視して手作業で行う**ため、経験と集中力が要求され、作業員への体力的負担が大きく、自動化が課題となっていました。



ポンプの流量調節を行う担当者

②解決策

- **ポンプ制御の自動化**
  - ・ 白浜町にあるマダイのいけすでは、AIやIoTを活用してポンプ制御を自動化した供給作業を行っています。



③効果

- ポンプの流量調節を行っていた作業員への体力的負担がなくなり、選別作業の効率化が図られています。



レゴ® マインドストーム® EV3版

**きのくにICT教育  
中学校プログラミング教育  
学習指導案 補助資料**

発行:2019(平成31)年3月

和歌山県教育庁学校教育局義務教育課  
〒640-8585

和歌山県和歌山市小松原通一丁目1番地  
TEL.073-441-3651/FAX.073-424-8877