

# プログラミング概論

第5回 2024年10月23日

プログラミングの基本概念(3)

繰り返し

# 今回の授業内容

- 繰り返し処理とは
- 迷路パズルとblocklyでは
- プログラムを作ってみる



**繰り返し処理とは**



# 繰り返し

- プログラムではある目的が達成されるまで動作を“繰り返す”ということがよく行われます
    - 変数や条件分岐と組み合わせることで様々な動作が実現できる
      - ○○回繰り返す
      - ...の間はずっと繰り返す
      - ...になるまではずっと繰り返す
- のように様々な繰り返し方がある
- 例： **15回**前期の講義を受ける（繰り返し受ける）  
**倒れるまで**走り続ける

# 「Blockly Games: 迷路」を クリック



プログラミング概論 - MATSUMOTO x +

保護されていない通信 | daisy.fwex.tohoku-gakuin.ac.jp/proginthro.html

Gmail YouTube マップ プログラミング概論 -...

東北学院大学 教養学部 情報科学科 松本章代 研究室

## MATSUMOTO Akiyo Laboratory

Tohoku Gakuin University

ホーム 研究紹介 講義 研究室メンバー 業績リスト 備忘録 リンク集

### プログラミング概論

#### 授業に必要なソフトのインストール・設定

- [NotePad++](#)
- [BlueStacks 5](#)
- [Google Chrome](#)
- [Google アカウント](#)

#### プログラミングサイト

- [Blockly Games: 迷路](#)
- [Blockly Games: Pond Tutor](#)
- [Blockly Code](#)
- [ピクトプログラミング](#)
- [MIT App Inventor](#)

# 迷路の問題 6 (復習)

Blockly Games : 迷路

6

10



▶ プログラムを実行

まっすぐ進む

左を向く ◻ ▾

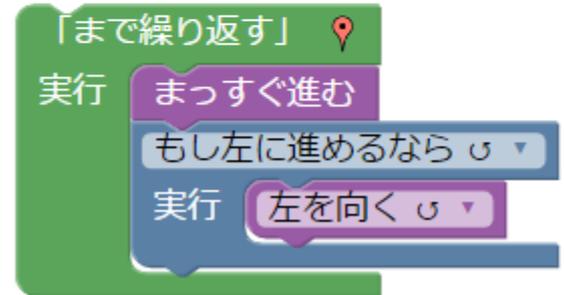
右を向く ◻ ▾

「まで繰り返す」 ◻

実行

もし左に進めるなら ◻ ▾

実行



# プログラムの基本：条件（分岐）と繰り返し

繰り返し（「～まで繰り返す」）



条件(分岐)（「もし～なら・・・する」）

# 繰り返し

---

---

「～まで繰り返す」のような同じことを繰り返し行う



例：迷路問題 6

もしまっすぐ進めるならばまっすぐ進む、  
そうでなければ(それ以外では) 左を向く  
ということを繰り返す

# Blockly Codeでは



プログラミング概論 - MATSUMOTO x +

保護されていない通信 | daisy.fwex.tohoku-gakuin.ac.jp/progintro.html

Gmail YouTube マップ プログラミング概論 -...

東北学院大学 教養学部 情報科学科 松本章代 研究室

## MATSUMOTO Akiyo Laboratory

Tohoku Gakuin University

ホーム 研究紹介 講義 研究室メンバー 業績リスト 備忘録 リンク集

### プログラミング概論

#### 授業に必要なソフトのインストール・設定

- [NotePad++](#)
- [BlueStacks 5](#)
- [Google Chrome](#)
- [Google アカウント](#)

#### プログラミングサイト

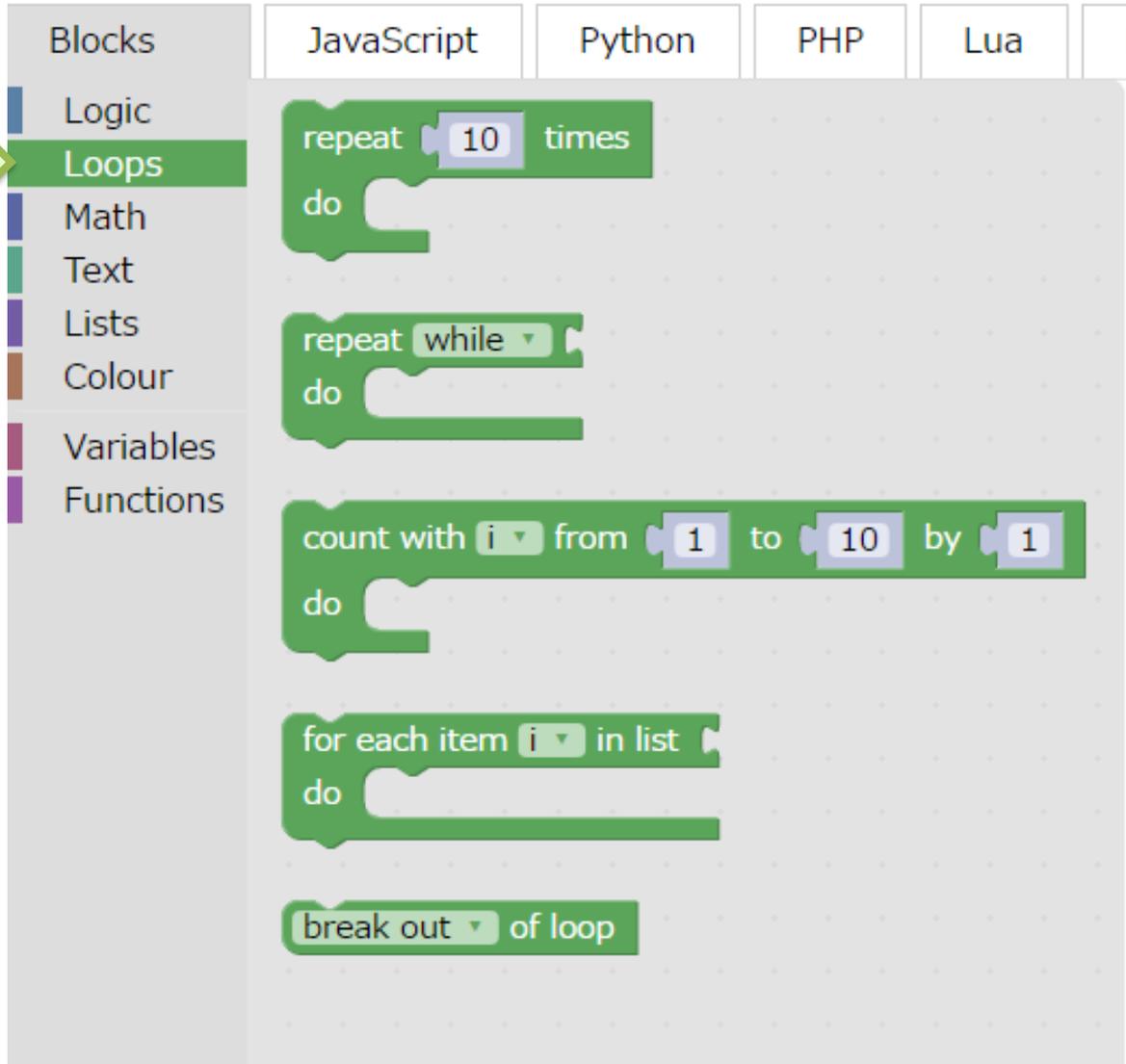
- [Blockly Games: 迷路](#)
- [Blockly Games: Pond Tutor](#)
- [Blockly Code](#)
- [ピクトプログラミング](#)
- [MIT App Inventor](#)

# プログラミング時の注意(再)

- 数字、アルファベットは“半角”
  - プログラミングでは半角が基本
  - 半角でないと数字として入力できない
  - 全角とまぜると条件分岐で間違えたりする
  - printで表示する部分は全角でもよい

(皆さんに定着するまで繰り返し注意する)

# 繰り返し : Loopsのなかをみてみよう



The image shows the Scratch Loops block palette. A green arrow points to the 'Loops' category. The palette includes the following blocks:

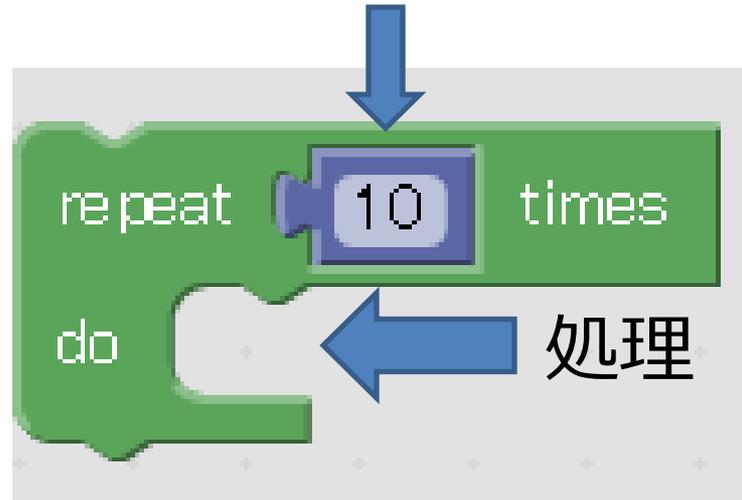
- repeat 10 times
- do
- repeat while
- do
- count with i from 1 to 10 by 1
- do
- for each item i in list
- do
- break out of loop

繰り返しの命令に  
使われる  
ブロック  
群

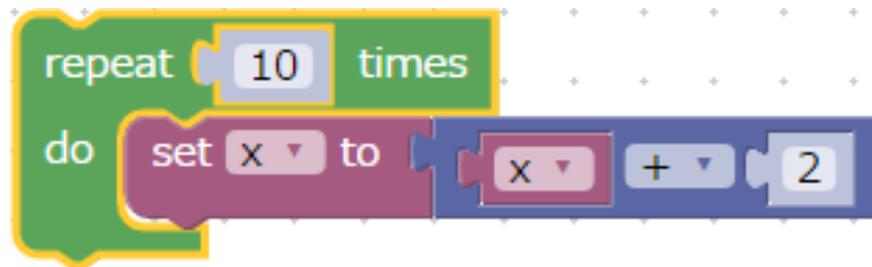
# 決まった動作を指定回数繰り返す : repeat ... times

---

ここに繰り返しの回数を入力

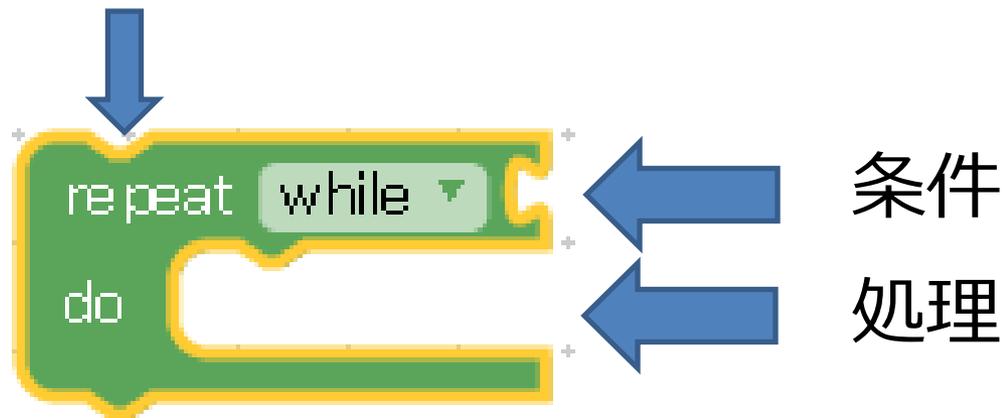


プログラム例：変数 x に 2 を 10 回加える



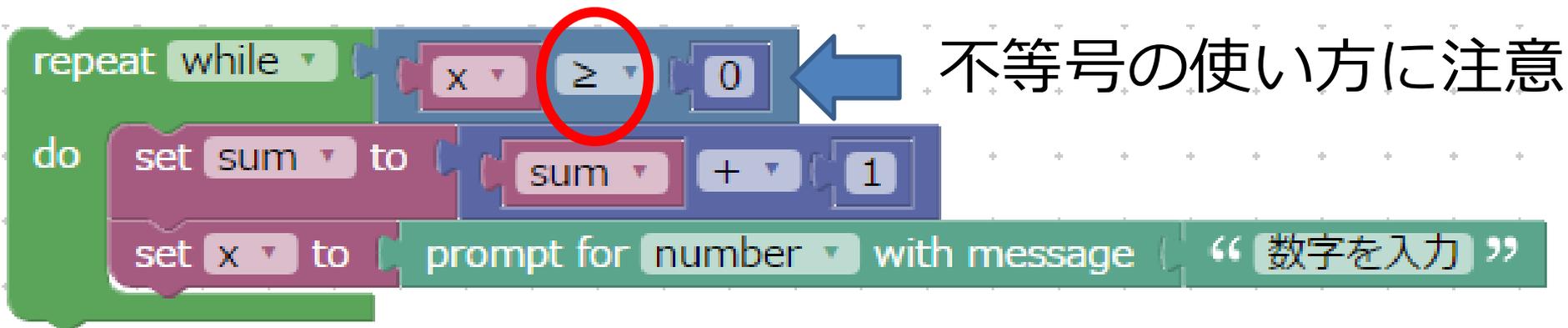
# 条件を満たす限り繰り返す : while

初期値



例 : xに負の数が入力されるまでsumに1を加える

⇒ 0以上の数が入力される限り



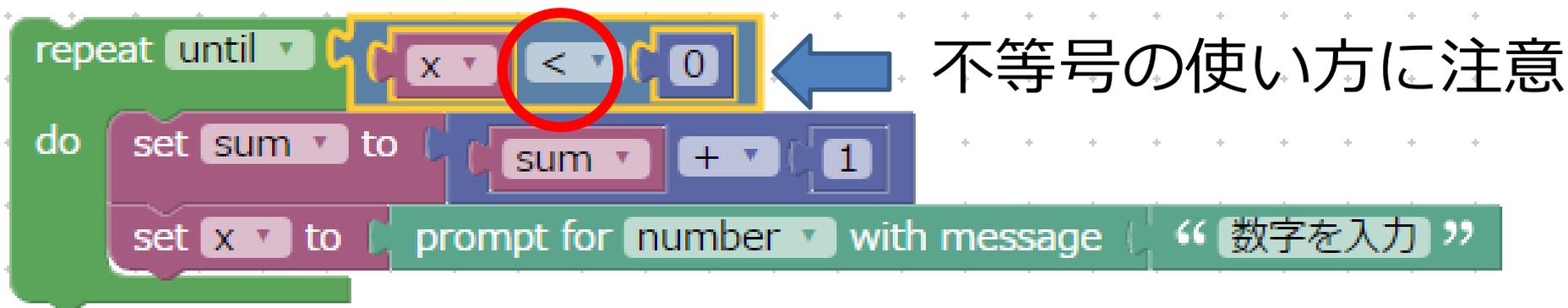
# 条件を満たすまで繰り返す(until)

whileをuntilに変更

初期値



例：x に負の数が入力されるまでsumに1を加える



# カウントしながら、決まった数だけ繰り返す

変数*i*が0から1ずつ増やし、10になるまで繰り返す

変数*i*

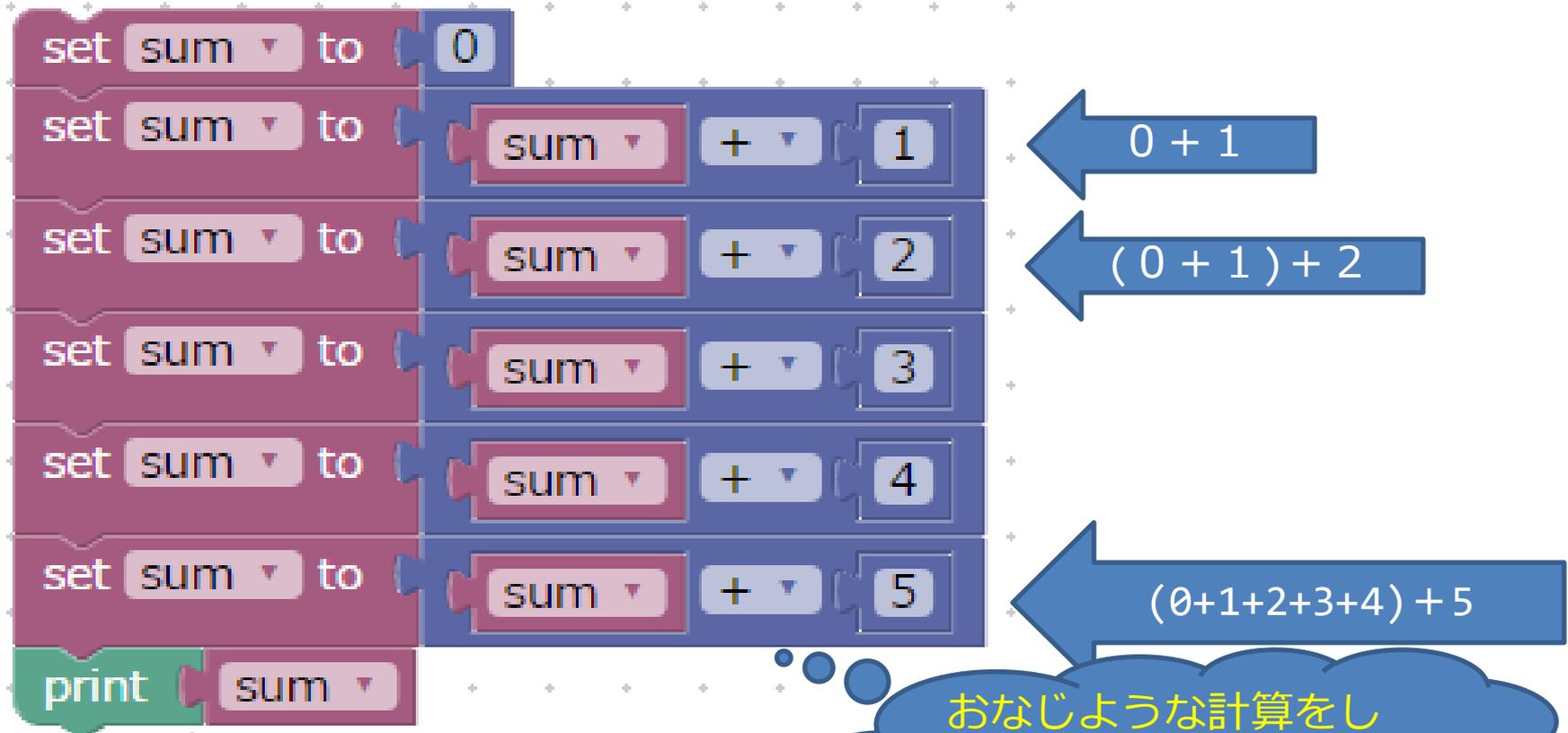


例：0から10までの数を表示する



# 繰り返しを使わないと

例題 変数sumを作りsum=0から始め、sumに1、2、…、5を次々に加えることによって、その合計を表示するプログラムを以下のように作ってください。



1 + 2 + 3 + 4 + 5の計算  
結果がsumに入っている

おなじような計算をしている⇒1からはじまり5まで繰り返し

# ブロックの複写（コピー）

同じようなブロックを作る際には複写が便利です。

- ① コピーしたいブロックの上で右クリック
- ② メニューからDuplicate（複写）を選択してクリック  
またはコピーしたいブロックをクリックしctrl+c, ctrl+v

例 set sum to sum+1 のブロックを複写

ブロック上で  
右クリック

Duplicate  
Add Comment  
Collapse Block  
Disable Block  
Delete 4 Blocks  
Help  
Create 'get sum'

Duplicate  
を選択

同じブロックがもう1つできた

# 実行結果



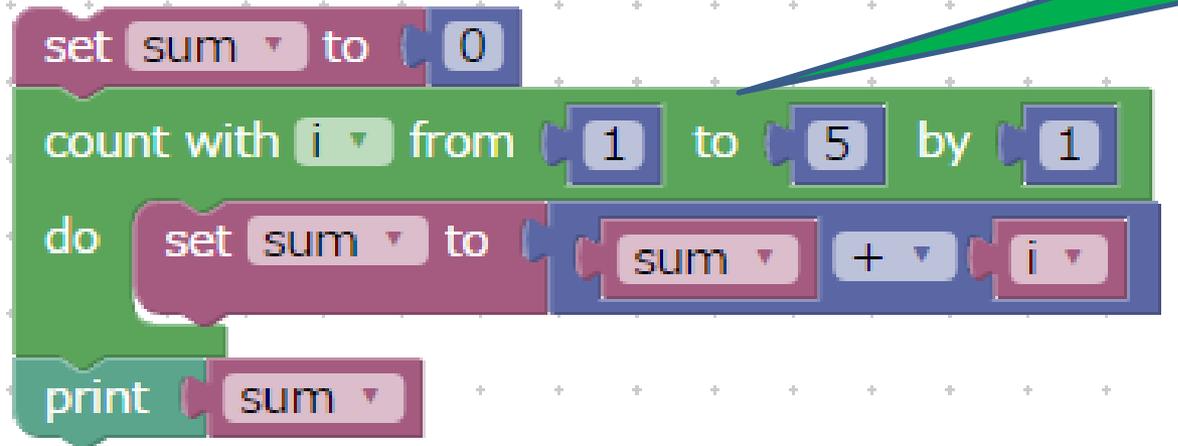
sumとして1+2+3+4+5の結果になっている

sum = 0 から始めて  
sum ← sum + 1 (= 0 + 1)  
sum ← sum + 2 (= 0 + 1 + 2)  
⋮  
sum ← sum + 5 (= 0 + 1 + 2 + ... 4 + 5)  
という計算はsum ← 1 + 2 + 3 + 4 + 5ということになる

# 繰り返しをつかおう

例題 変数sumを作りsum=0から始め、sumに1、2、…、5を次々に加えることによって、その合計を表示するプログラムを繰り返し処理を用いて以下のように作ってください。

i=1,2,3,4,5として繰り返し



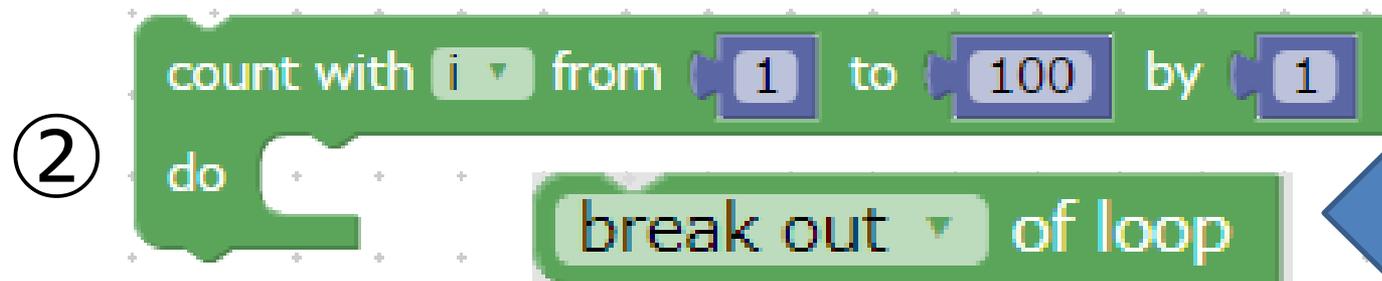
sum+1  
sum+2  
:  
sum+5

1 + 2 + 3 + 4 + 5の計算  
結果がsumに入る

# プログラムを作ってみよう 1

## 問題 1

・負の数が入力されるまで数字を入力し、その合計（変数名sum）を表示するプログラムを以下の2通り作成してください。まずwhileを使うプログラムを作成し、次にcountを使うプログラムを作成してください。（入力回数は100回以下とし、繰り返し処理を抜ける（=終わる）際にはif 文とbreak out of loopを使います）



繰り返し処理  
を抜ける命令

# 結果例

入力

Four stacked dialog boxes, each titled "ページ https://blockly-demo.appspot.com の記述: ✕". Each dialog contains the text "数字を入力" and a text input field. The input fields contain the values 2, 3, 5, and -1 respectively. The bottom-most dialog also features "OK" and "キャンセル" buttons.

出力

A single dialog box titled "ページ https://blockly-demo.appspot.com の記述: ✕". It displays the number "10" in the center. At the bottom right, there is an "OK" button.

# 解答ヒント①repeatを使って

The image shows a Scratch script on a grid background. The script consists of the following blocks from top to bottom:

- set `x` to 0
- set `sum` to 0
- repeat loop (containing two blocks):
  - do loop (containing two blocks):
    - set `sum` to [ ] (with annotation: sumに `x` を加える)
    - set `x` to [ ] (with annotation: `x` へキーボードからの数字の入力と代入)
  - Annotation: `x` が0以上である限り繰り返す
- say [ ] for 2 seconds (with annotation: 合計の表示)

# 解答のヒント②COUNTを使って

The image shows a Scratch script on a grid background. The script consists of the following blocks:

- set x to 0
- set sum to 0
- count with i from 1 to 100 by 1
- do loop containing:
  - sumの計算と代入 (Annotation)
  - xへキーボードからの数字の入力と代入 (Annotation)
  - if loop containing:
    - やめる際のxの条件 (Annotation)
    - do loop containing:
      - 繰り返しから抜ける (Annotation)
- 合計の表示 (Annotation)

# 解答例 ①repeatを使って

```
set x to 0
set sum to 0
repeat while x >= 0
do
  set sum to sum + x
  set x to prompt for number with message "数字を入力"
print sum
```

The image shows a Scratch script on a grid background. It starts with two 'set' blocks: 'set x to 0' and 'set sum to 0'. This is followed by a 'repeat while' loop. The condition for the loop is 'x >= 0'. Inside the loop, there are two 'do' blocks: 'set sum to sum + x' and 'set x to prompt for number with message "数字を入力"'. The 'prompt for number' block has a speech bubble icon and the message text is in Japanese. After the loop, there is a 'print' block for 'sum'.

# 解答例 ②COUNTを使って

```
set x to 0
set sum to 0
count with i from 1 to 100 by 1
do
  set sum to sum + x
  set x to prompt for number with message "数字を入力"
  if x < 0
  do
    break out of loop
print sum
```

The image shows a Scratch script on a grid background. It starts with two 'set' blocks: 'set x to 0' and 'set sum to 0'. A green 'count with i from 1 to 100 by 1' block is followed by a 'do' loop. Inside the loop, there are three blocks: 'set sum to sum + x', 'set x to prompt for number with message "数字を入力"', and an 'if' block with a gear icon. The 'if' block contains 'x < 0' and a 'do' loop with 'break out of loop'. After the 'do' loop, there is a 'print sum' block.

# プログラムを作ってみよう 2

## 問題 2

- 「点数入力」と表示させて、キーボードから点数を繰り返し入力させ（負の点数が入力されたら終了）、60点以上の回数と未満の回数を計算し表示させよう。

たとえば以下のように数値を繰り返し入力すると、

76, 55, 90, 30, 87, -10

ページ <https://blockly-demo.appspot.com> の記述: ×

3,2

結果として以下のように表示

3,2

（この場合60点以上が3回、未満が2回、-10は数えない）

# 問題 2 解答例

```
set a to 0
set b to 0
set x to prompt for number with message "点数を入力"
repeat while x >= 0
do
  if x >= 60
  do
    set a to a + 1
  else
    set b to b + 1
  set x to prompt for number with message "点数を入力"
print create text with a, b
```

xが0以上で繰り返す（xが負で処理からぬける）

もしxが60以上なら  
aに1を加え  
そうでないなら  
bに1を加える

# プログラムを作ってみよう 3

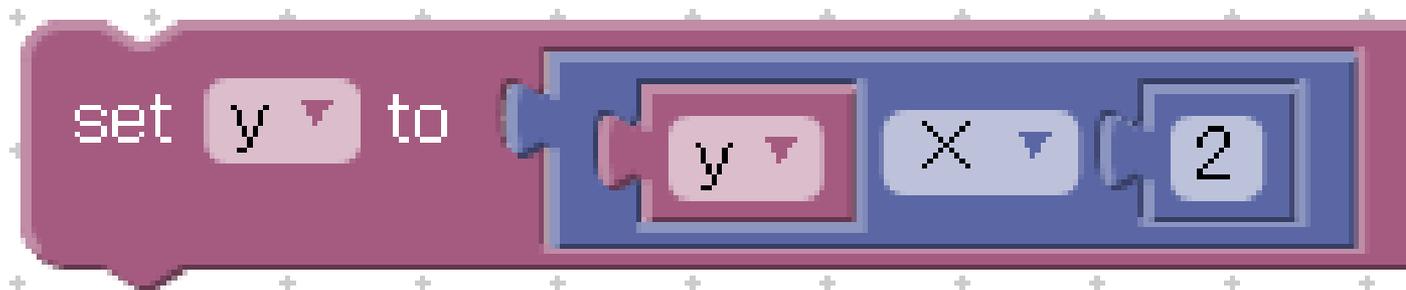
## 問題 3

・「2を何乗しますか」という表示とともに数字を入力させる。その数を変数nに代入し、2のn乗を計算した結果を表示させよう。

2のn乗とは

例 2の3乗の場合  $2 \times 2 \times 2 = 8$

Hint : 以下の操作は変数 y に 2 を 1 回かけることに対応する



↑変数yに2をかけて、変数yに代入する

# 結果例

入力

blockly-demo.appspot.com の内容: ×

2を何乗しますか

blockly-demo.appspot.com の内容: ×

16

blockly-demo.appspot.com の内容: ×

2を何乗しますか

出力



blockly-demo.appspot.com の内容: ×

256

# 問題 3 解答例



```
set y to 1
set n to prompt for number with message "2を何乗しますか"
repeat n times
do
  set y to y x 2
print y
```

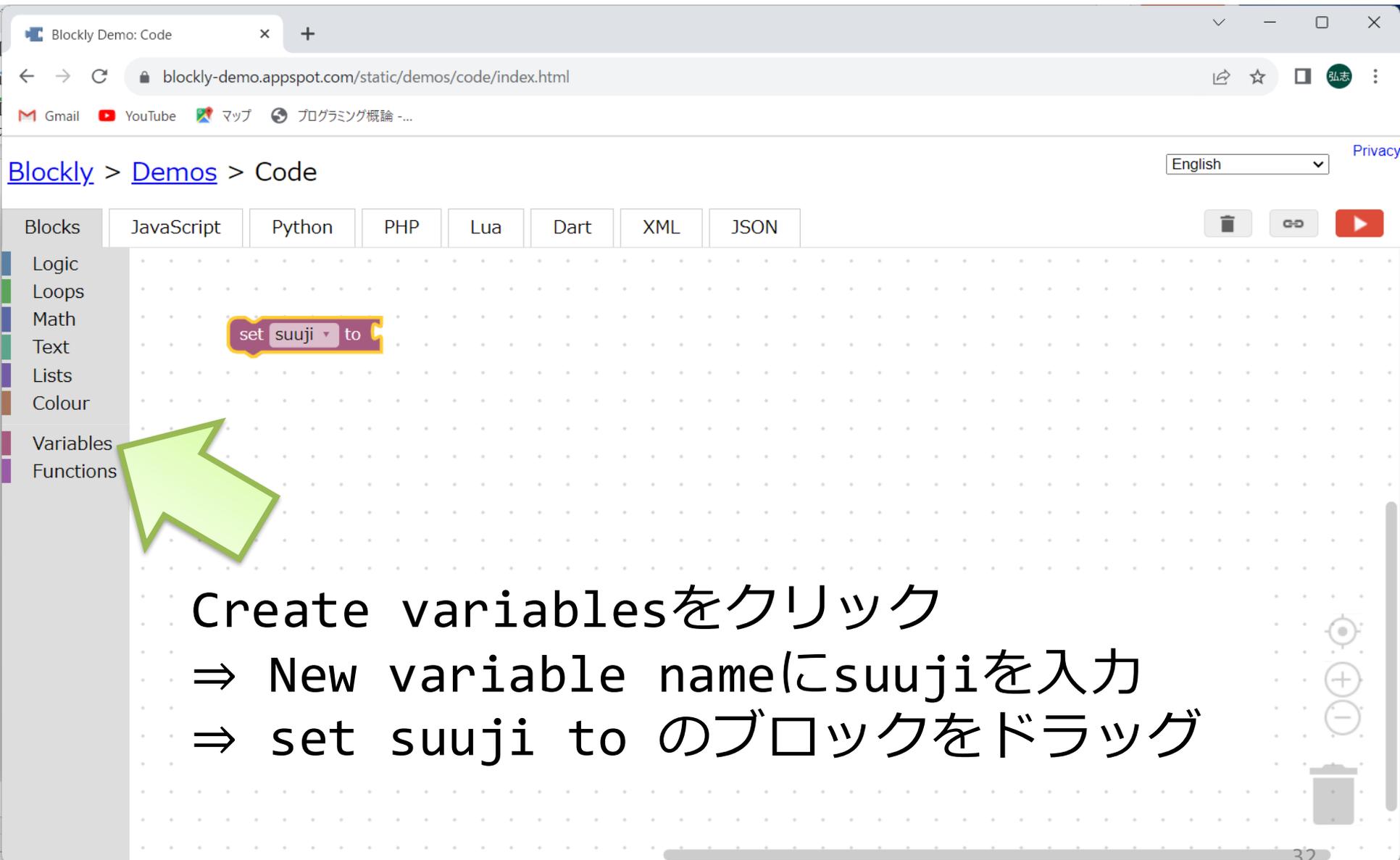
The image shows a Scratch script on a grid background. It consists of five code blocks: 1. A purple 'set' block for variable 'y' with the value '1'. 2. A green 'set' block for variable 'n' with the text 'prompt for number with message' and a speech bubble containing '2を何乗しますか'. 3. A green 'repeat' block with 'n' in the 'times' field. 4. A purple 'do' block containing a 'set' block for 'y' with 'y' in the 'to' field and 'x' in the 'operator' field, followed by a blue 'multiply' block with '2' in the 'number' field. 5. A green 'print' block with 'y' in the 'text' field.

# 応用

- “リスト”で変数を便利に使う
  - いくつもの変数を一つのリストとして作成できる
  - 繰り返しと組み合わせると便利

第3回と同じことを繰り返しを使ってやる

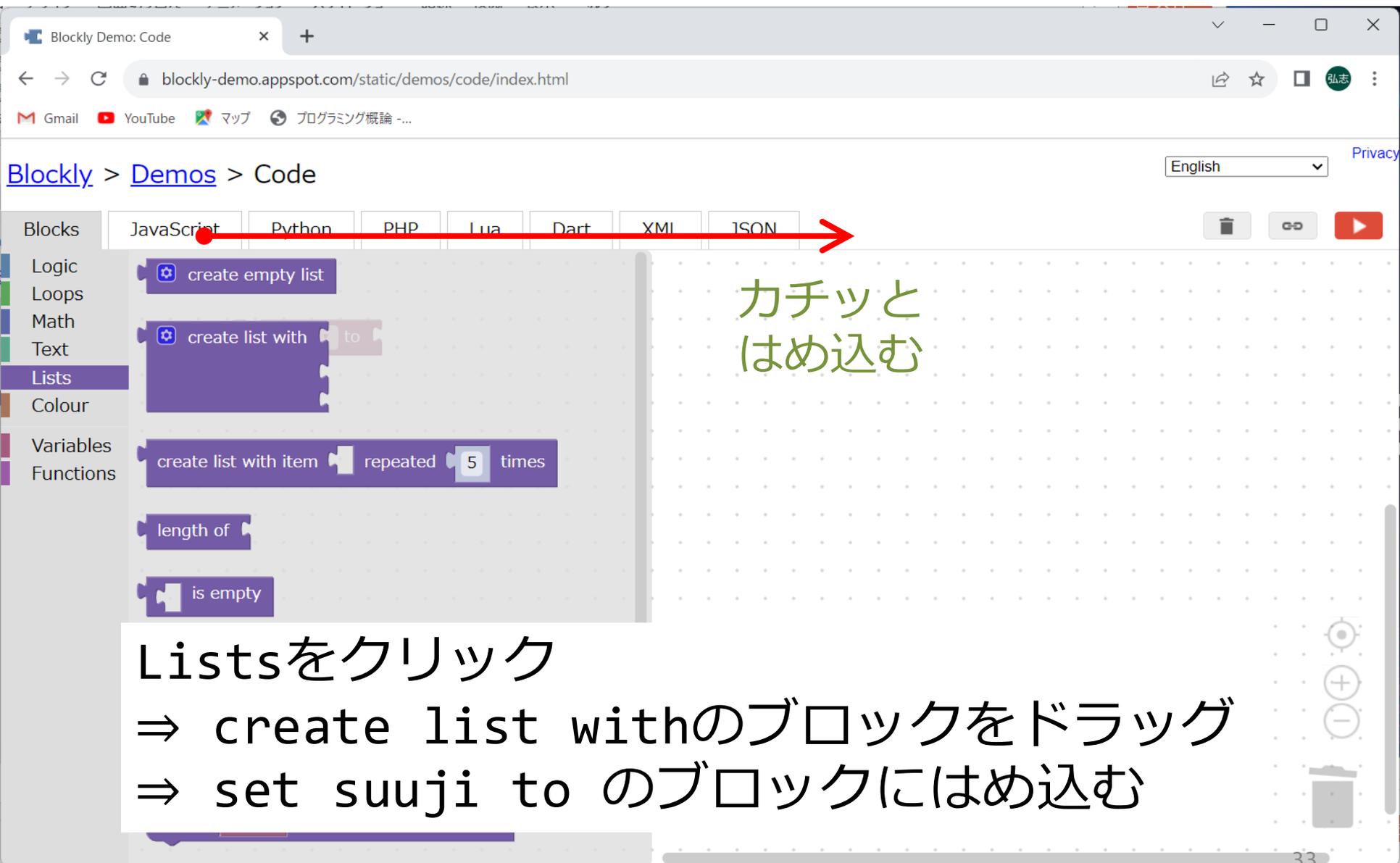
# suujiという変数を作成する



The screenshot shows the Blockly Demo Code editor interface. The browser address bar displays 'blockly-demo.appspot.com/static/demos/code/index.html'. The page title is 'Blockly > Demos > Code'. The left sidebar lists various block categories: Logic, Loops, Math, Text, Lists, Colour, Variables, and Functions. The 'Variables' category is highlighted, and a green arrow points to it. The main workspace contains a single block labeled 'set suuji to', which is highlighted with a yellow border. The workspace background is a light gray grid.

Create variablesをクリック  
⇒ New variable nameにsuujiを入力  
⇒ set suuji to のブロックをドラッグ

# 「suuji」という名前のリストを作成

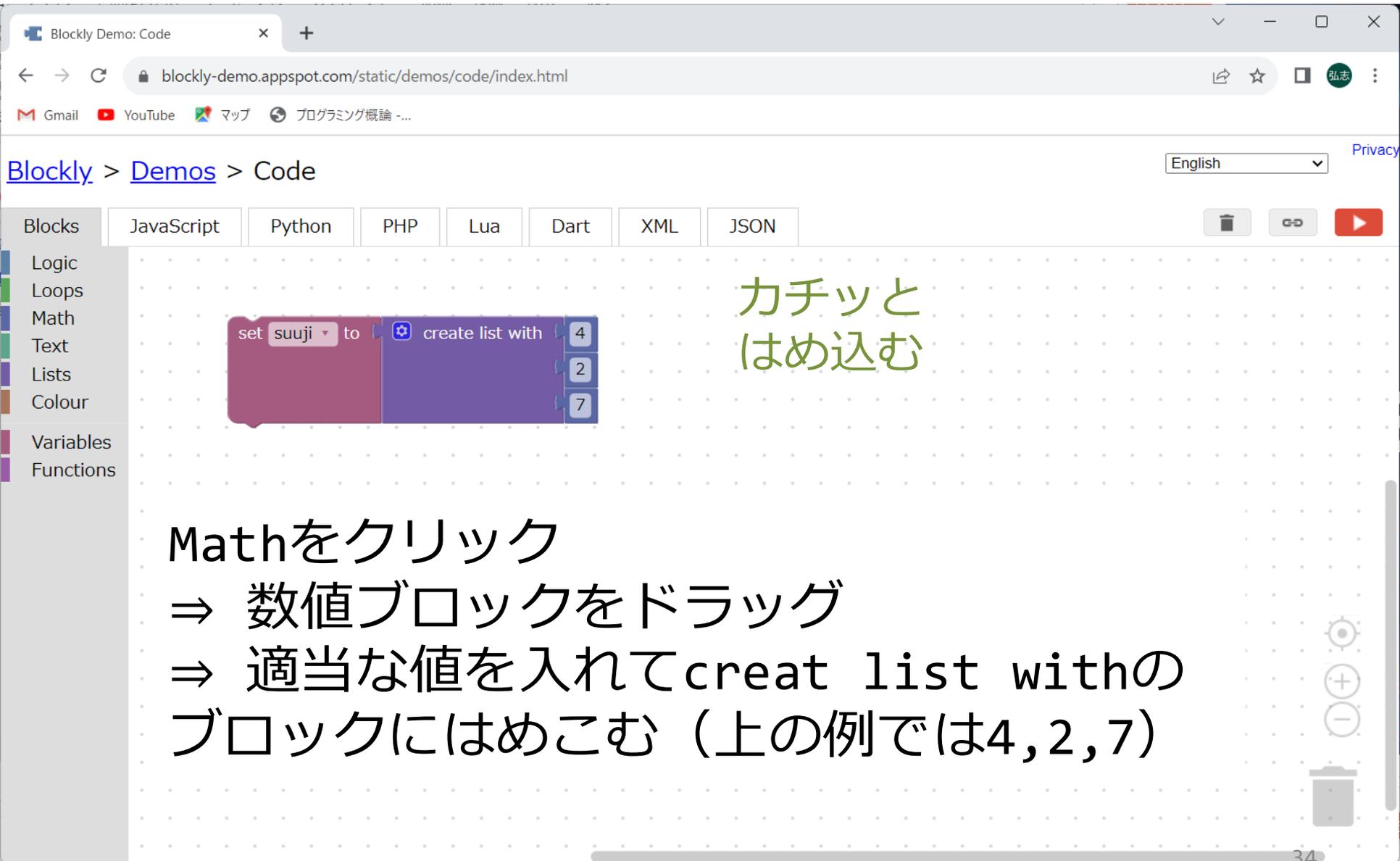


The screenshot shows the Blockly Demo: Code interface. The 'JavaScript' tab is selected, and the 'Lists' category is highlighted in the left sidebar. The 'create list with' block is being dragged from the sidebar into the workspace. A red arrow points from the 'JavaScript' tab to the 'create list with' block. The workspace contains a 'set suuji to' block with the value 'suuji'.

カチッと  
はめ込む

Listsをクリック  
⇒ create list withのブロックをドラッグ  
⇒ set suuji to のブロックにはめ込む

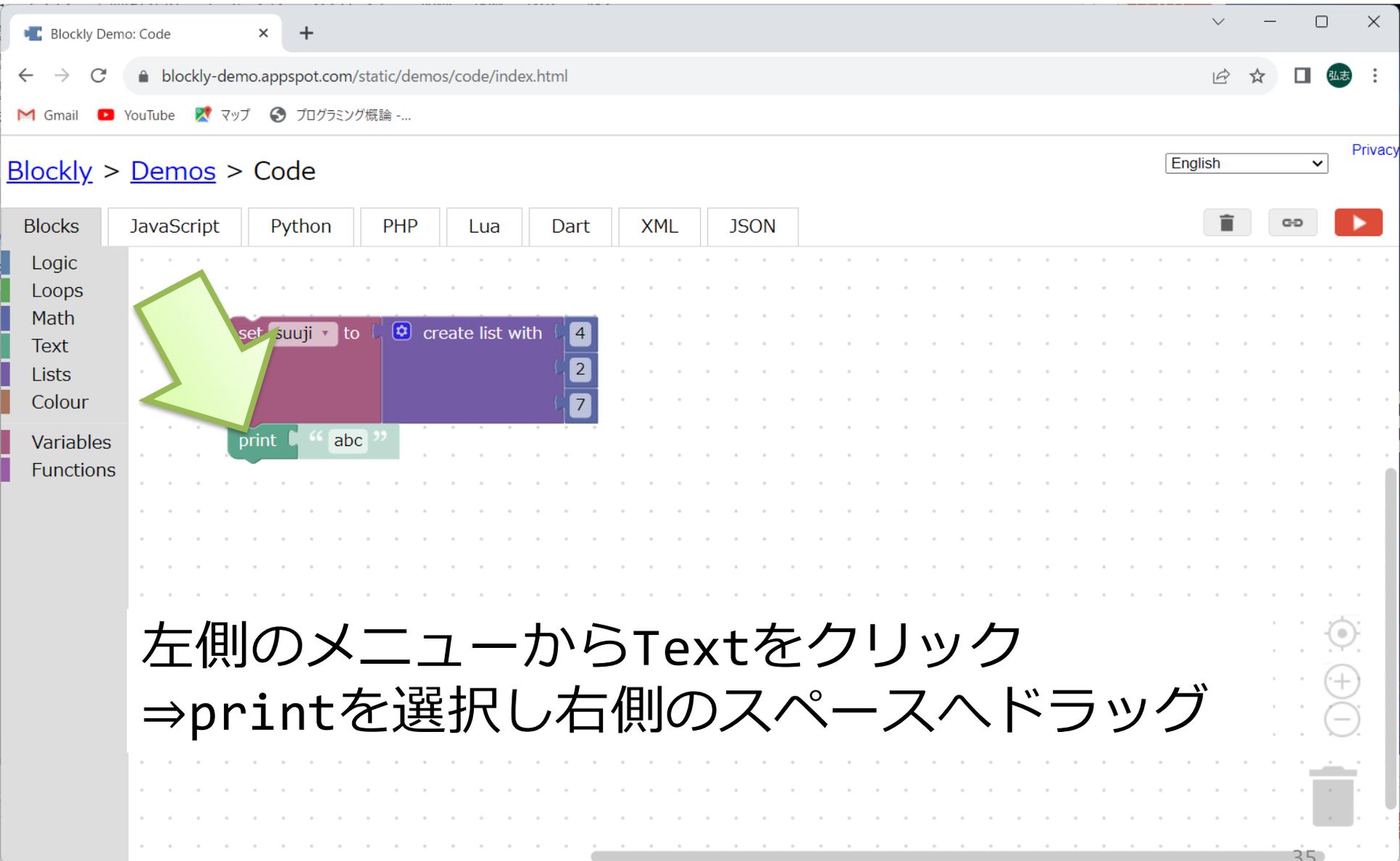
# 「suuji」に値を入れる



カチッと  
はめ込む

Mathをクリック  
⇒ 数値ブロックをドラッグ  
⇒ 適当な値を入れてcreat list withの  
ブロックにはめこむ（上の例では4,2,7）

# リストの中身を表示する (1)



The screenshot shows the Blockly Demo: Code interface. The browser address bar displays `blockly-demo.appspot.com/static/demos/code/index.html`. The page title is `Blockly > Demos > Code`. The language is set to `English`. The left sidebar shows the `Blocks` menu with categories: `Logic`, `Loops`, `Math`, `Text`, `Lists`, `Colour`, `Variables`, and `Functions`. The main workspace contains a script with the following blocks: `set suuji to` (purple), `create list with` (purple) with values `4`, `2`, and `7`, and `print` (green) with the text `" abc "`. A green arrow points to the `print` block.

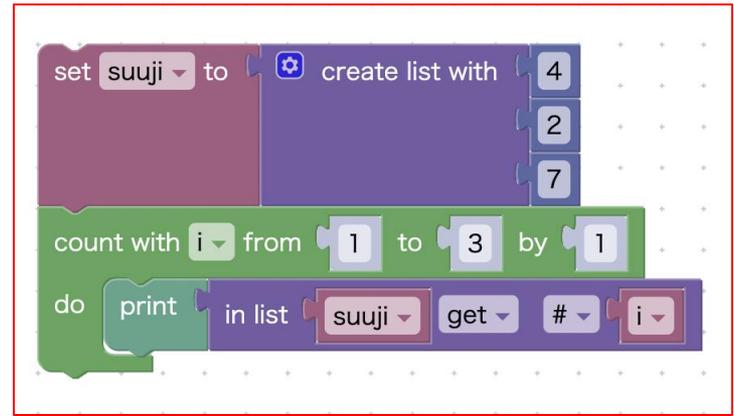
左側のメニューからTextをクリック  
⇒printを選択し右側のスペースヘドラッグ



## リストの中身を表示する (3)



```
set suuji to create list with 4 2 7
print in list suuji get # 1
print in list suuji get # 2
print in list suuji get # 3
```



```
set suuji to create list with 4 2 7
count with i from 1 to 3 by 1
do print in list suuji get # i
```

⇒ 繰り返しを使うことでprintのブロックをまとめられる  
好きな部分だけ表示するなどにも楽にできる

# 実行結果

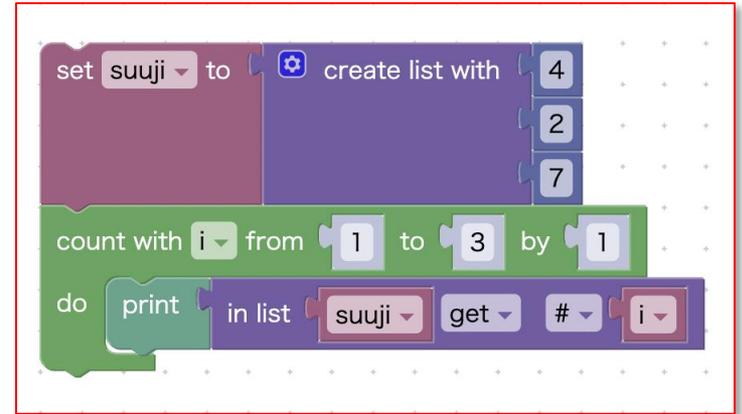
リストの  
1番目



リストの  
2番目



リストの  
3番目



リストの値が順番に表示される

- 一つの変数名で多くの数を扱うことができる
- リストの一つ一つを変数と同様に値を入れたり参照したりできる

作業終了後（終了しなくても16:45まで）

# manabaで振り返りを提出

- manabaにログイン
- プログラミング概論の小テストから第5回振り返り課題を選択し各問題のプログラムを作成し正しく動作できたかを回答する
- 感想や分からなかった点などもあれば記入

# 終了：今日のまとめと次回予告

## ■今日の授業

- repeat文で繰り返しの処理が指定できる
- repeat whileで条件を満たす限り繰り返す
- repeat untilで条件を満たすまで繰り返す
- repeat x timesでx回繰り返す
- count with i from x to y step zは変数iがxからzずつ増やし、yになるまで繰り返す
- break out of loopで繰り返し処理を抜ける

## ■次回はApp Inventorについて

# 時間が余った人は

- 迷路、Pondやほかのゲームに挑戦してみよう  
(繰り返しや条件分岐も使います)

ブラウザのタブ: ブロックリーゲーム: 迷路, プログラミング概論 - MATSUMOTO

アドレスバー: 保護されていない通信 | daisy.fwex.tohoku-gakuin.ac.jp/progintro.html

ページタイトル: プログラミング概論 -...

東北学院大学 教養学部 情報科学科 松本章代 研究室

## MATSUMOTO Akiyo Laboratory

Tohoku Gakuin University

ホーム 研究紹介 講義 研究室メンバー 業績リスト 備忘録 リンク集

### プログラミング概論

#### 授業に必要なソフトのインストール・設定

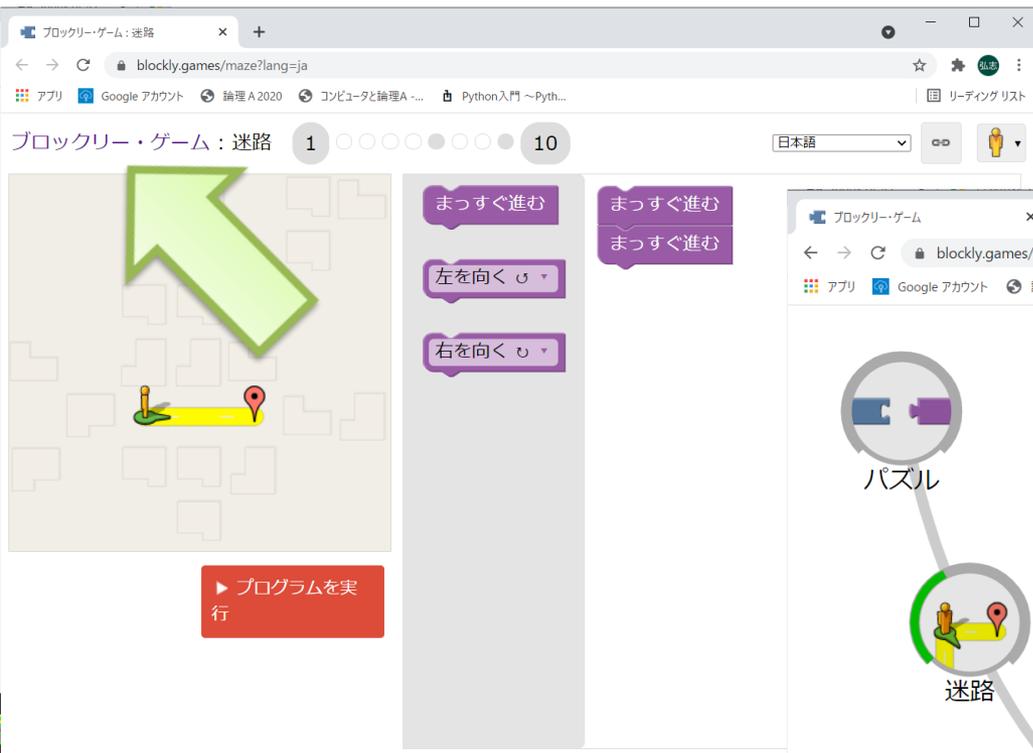
- [NotePad++](#)
- [BlueStacks 5](#)
- [Google Chrome](#)
- [Google アカウント](#)

#### プログラミングサイト

- [Blockly Games: 迷路](#)
- [Blockly Games: Pond Tutor](#)
- [Blockly Code](#)
- [ピクトプログラミング](#)
- [MIT App Inventor](#)

← Pondに挑戦するときはここをクリック

# ほかのゲームもやってみる



# さらに興味のある人へ

ブロックリーゲーム: 迷路 x プログラミング概論 - MATSUMOTO x +

← → ↻ ▲ 保護されていない通信 | daisy.fwex.tohoku-gakuin.ac.jp/progintro.html

📍 プログラミング概論 -...

東北学院大学 教養学部 情報科学科 松本章代 研究室  
**MATSUMOTO Akiyo Laboratory**  
Tohoku Gakuin University

ホーム 研究紹介 講義 研究室メンバー 業績リスト 備忘録 リンク集

## プログラミング概論

### 授業に必要なソフトのインストール・設定

- [NotePad++](#)
- [BlueStacks 5](#)
- [Google Chrome](#)
- [Google アカウント](#)

### プログラミングサイト

- [Blockly Games: 迷路](#)
- [Blockly Games: Pond Tutor](#)
- [Blockly Code](#)
- [ピクトプログラミング](#)
- [MIT App Inventor](#)

← ピクトプログラミングをクリック

# ピクトグラミング

日本語 English

## PICTOGRAMMING

### ピクトグラミング

「ピクトグラミング」は、ピクトグラムの作成を通じて、プログラミングの基礎概念や情報デザインについて学習できる統合型アプリケーションです。「ピクトグラミング」には、「Pictoch(ピクトチ)」、「PICTO」、「Pichon(ピクソン)」、「JavaScript(ジャバスクリプト)」、「Piby(ピビー)」の派生アプリケーションもあり、これらをまとめてピクトグラミングシリーズと呼んでいます。検索利用にもぜひお役立てください。誰でも自由に利用できます。

PICTOGRAMMING シリーズ

### PICTOGRAMMING (ピクトグラミング)

はじめる  
使い方

ピクトグラミング(Pictogramming)は、スペース区切りのテキスト文字列を入力してプログラミングします。命令は、英語表記、日本語表記、日本語ひらがな表記をサポートしています。

PICTOGRAMMING シリーズ

ーズの「含む」タグと法工「新向け

下にスクロール

PICTOGRAMMING シリーズ

## PICTO (Pseudocode for Informatics of Common Test Optimizer)

はじめる  
使い方

PICTO ... ピクトグラミングシリーズは、Python風なブロックプログラミングとPython風なテキストプログラミングに準じたPython風なブロックプログラミングとPython風なテキストプログラミングを組み合わせることで、Python風なプログラミングを実現します。できる処理はピクトグラミングやPictochと同様です。人型ピクトグラムとの対話を通じたわかりやすい体験を通じて、Python風の記法に慣れ親しむことができます。高等学校情報科の実習に最適です。

PICTOGRAMMING シリーズ

## Python (ピクソン)

Pictogramming

pictogramming.org/editor/block.html

アプリ Google アカウント 論理 A 2020 コンピュータと論理A ... Python入門 ~Pyth...

リーディングリスト

(181, -279)



動き  
図形  
論理  
繰り返し  
数学  
テキスト  
リスト  
変数  
関数  
イベント  
その他  
タイトル

実行ボタンが押されたとき

<https://pictogramming.org/>