

社会情報科学概論（川嶋-3/4） データとプログラム（実践編）

兵庫県立大学 社会情報科学部

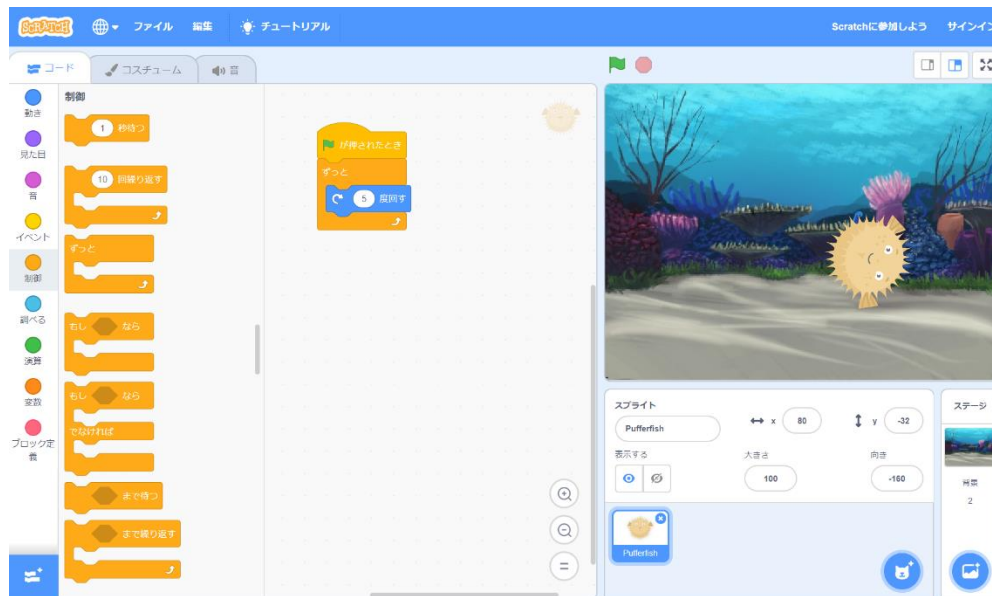
川嶋宏彰

kawashima@sis.u-hyogo.ac.jp

Scratch

2

- ScratchはMITメディアラボによるプログラミング学習環境
 - 現在Version 3.0
 - オンライン版 (ブラウザ上)
 - オフライン版 (インストール版)
- ビジュアルプログラミング
 - ブロックを組み合わせる
 - Scratch の他にも Blockly などいろいろ
- 子供から大人まで
- 結構複雑なものを開発可能



- Scratch は ゲームプログラミングに近い
 - **スプライト**：キャラクター（スプライトは複数あってよい）
 - **ステージ**：背景など（ステージは一つ）
 - 各「スプライト」もしくは「ステージ」にコードが紐づいている
 - それぞれのコードが並列に実行される
 - コード間ではメッセージや変数でやり取り・・・やや複雑
- 今日は一つのスプライトのコードだけを用いる
 - 並列処理なしが基本形（後期以降のプログラミング演習も）
 - いきなり中級クラス

行動分析？基礎

- 人の嗜好や行動には偏りがある
 - Amazonの推薦システム，行動ターゲティング広告
 - たとえじゃんけんであっても
- ランダムに手を出す相手にはじゃんけんの勝率は5割
 - 「グー」「チョキ」「パー」の確率がそれぞれ1/3
 - $\text{勝率} = \text{勝った回数} / (\text{勝った回数} + \text{負けた回数})$
- じゃんけんには癖がある
 - 人によって出す手に偏りがある
 - 状況にも依存する（過去の手が何だったか，過去の勝負結果）
- (例) 昭和～令和まで続いているアニメ
 - 1991年の秋に「んがぐぐ」から「じゃんけん」に変更

行動分析？基礎

- サザエさんじゃんけん研究所

- <http://park11.wakwak.com/~hkn/>
- 1991年 (平成3年)のじゃんけん開始時から記録

平成の記録といってもよい?! (約30年!)

- サザエさんじゃんけん白書 (2019年春版 4/6) によれば

- サザエさんじゃんけんにも中の人 (担当者) がいると思われる
- 各クールの初回には「チョキ」を出しやすい
- 3回続けて同じ手がでる確率がランダムに比べ極端に低い
- 傾向が過去に6通り (現在6人目の担当者?)
 - 過去2回に対し次に出た手の頻度を記録

社内人事の事情まで…

- 過去の傾向を分析して事前に予想 (現在はT

- 1996~2018年の勝率: 70%以上
- 平成最後の勝負も勝利でかざっている

クラスタリング, 変化点
検出, マルコフモデル…
(機械学習・時系列分析でも
おなじみの手法)

じゃんけんゲーム（機械学習版？）

6

1. じゃんけんをするゲームを作る

- 人 vs コンピュータ

2. 人の傾向を記録

- 「グー」「チョキ」「パー」の度数分布
- 最頻手の計算（最大値の計算アルゴリズム）

目標: 2 の途中

3. 統計分析: 偏り(癖)はあるのか？

- 帰無仮説: 等確率(全ての手が確率 $1/3$)である
- 対立仮説: 等確率ではない… カイ二乗検定

4. 機械学習による予測とコンピュータの手の決定

- 傾向に応じて出す手を変える (最頻手に勝つ手を一定割合入れる)
- 過去1, 2回や勝負結果等に応じた偏りも考慮して出す手を変える

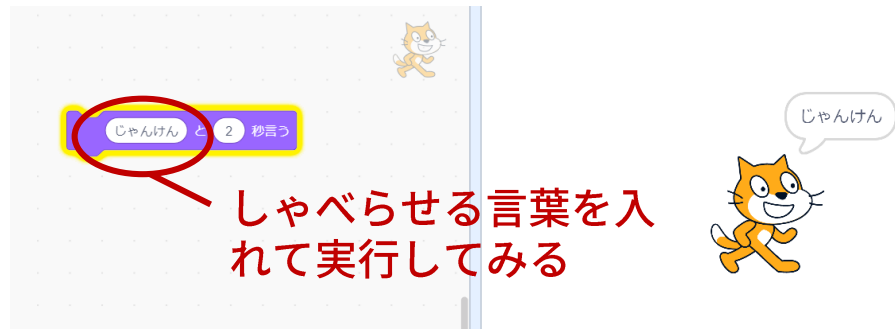
じゃんけんゲーム

7

・「じゃんけん」と言う



- ・ ブロックの色の部分をクリックすると実行される
- ・ つながったブロックの場合はまとめて実行される



・「ぽん」と聞いて待つ



- ⑤ 下に入力した文字列が表示されることを確認

- ② 2秒は長いので1に変えておく
(半角数字)

- ③ ブロックを実行

- ④ なにか入力してEnterする

- ① チェックしておくとも猫の左上に表示される



変数とリスト

8

- 仕様: 半角 g, c, p (グー, チョキ, パー) どれかを入力してもらう
 - $g \rightarrow 1, c \rightarrow 2, p \rightarrow 3$ に変換し, 内部では数値で扱う
 - 変換にはリストを使う. 変換後は「人の手」という名前の変数に入れる
- 変数**: 1つの値 (数値もしくは文字列)を記録しておける箱
- リスト**: 複数の値を記録しておける引き出し (はじめ0段, 追加削除可)

「リストを作る」
をクリック



リスト関係のブロックが追加される



変数とリスト

ここに入れた中身が何番目に入っているか(index)を教えてくれる
見つからないときは0になる

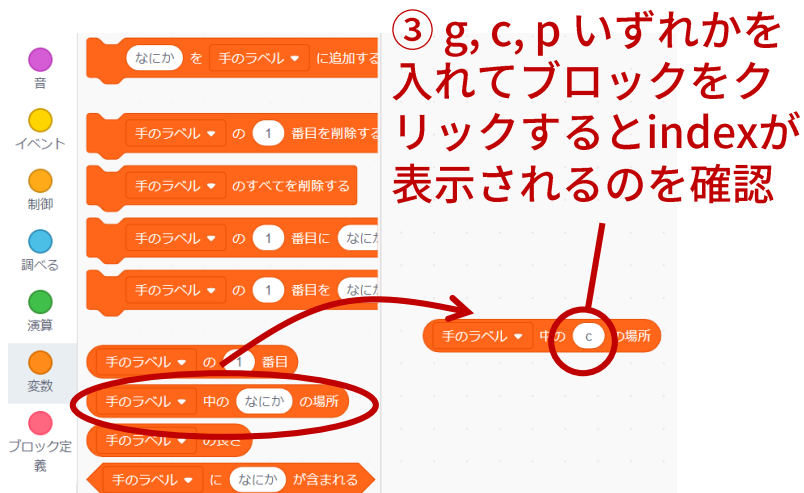
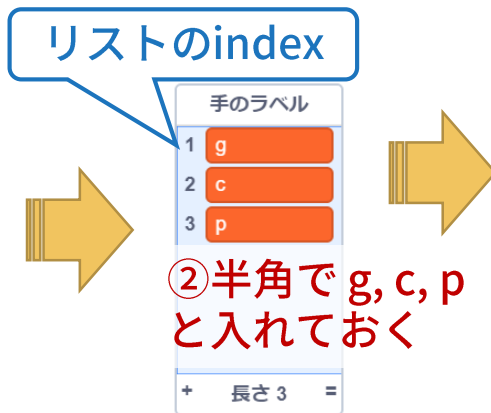
• $g \rightarrow 1, c \rightarrow 2, p \rightarrow 3$ に変換

• リストの添え字 (index) を返すブロック

リスト名

手のラベル ▾ 中の なにか の場所

を利用



⑥ 変換後の数値が表示
されるかクリックで確認

⑥ 変換後の数値が表示
されるかクリックで確認

変数とリスト

10

- 変換後の数値を変数に入れる
 - 人 (プレイヤー) の出した手を入れるので「人の手」という名前にしておく



① 「変数を作る」をクリックして「人の手」という名前の変数を作成

変数に関するブロックを試しておこう

☒ 人の手
☐ 変数

変数に値を入れるブロック

変数に値を加算するブロック (変数の中身が文字列のときは置き換えられる)



② リストで変換した数値を変数に入れる (代入)

リスト関係のレポーターブロックとブーリアンブロックも試してみよう

手のラベル の 1 番目

手のラベル 中の なにか の場所

手のラベル の長さ

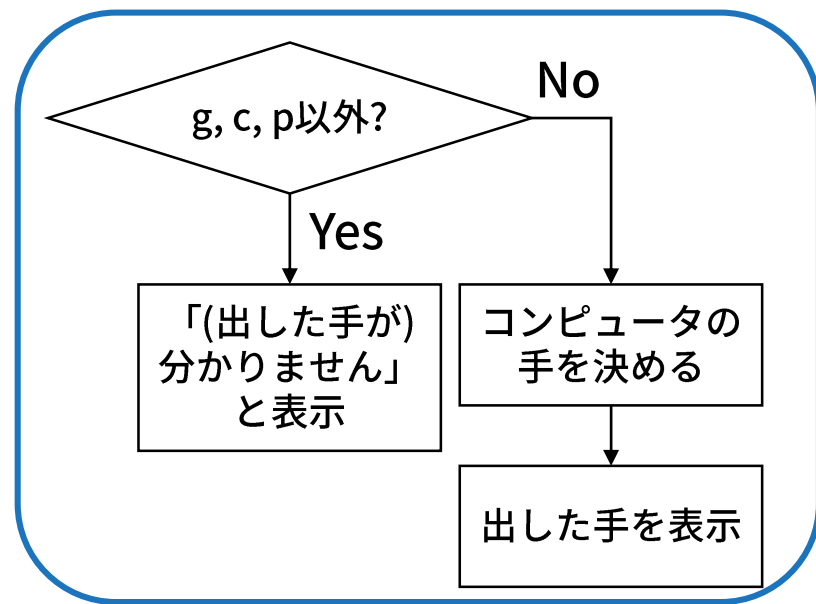
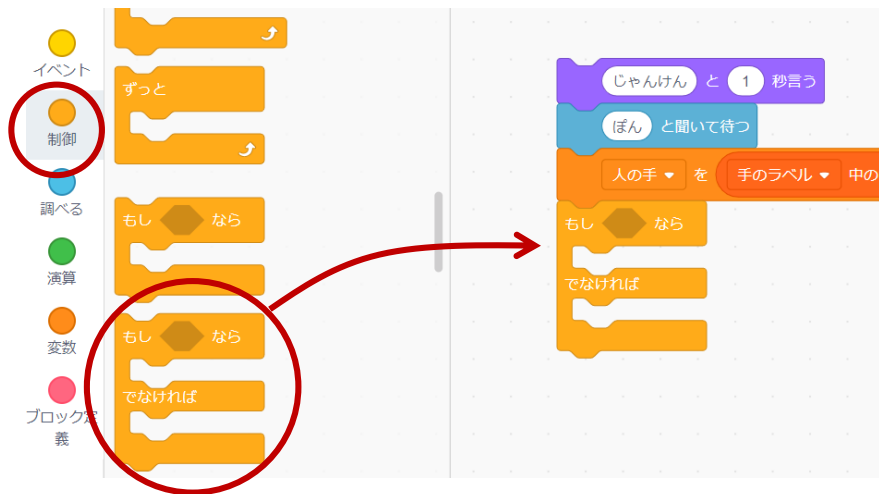
手のラベル に なにか が含まれる

boolean: 真偽

条件分岐

11

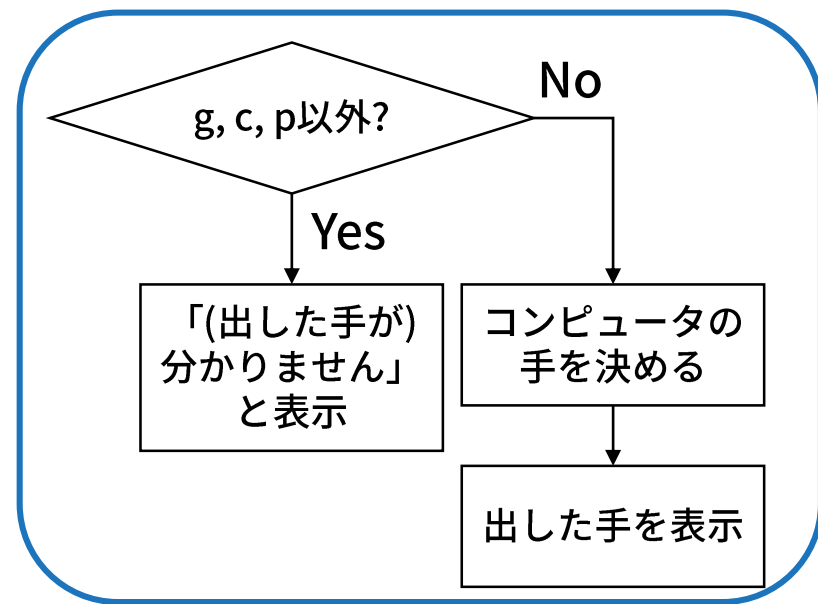
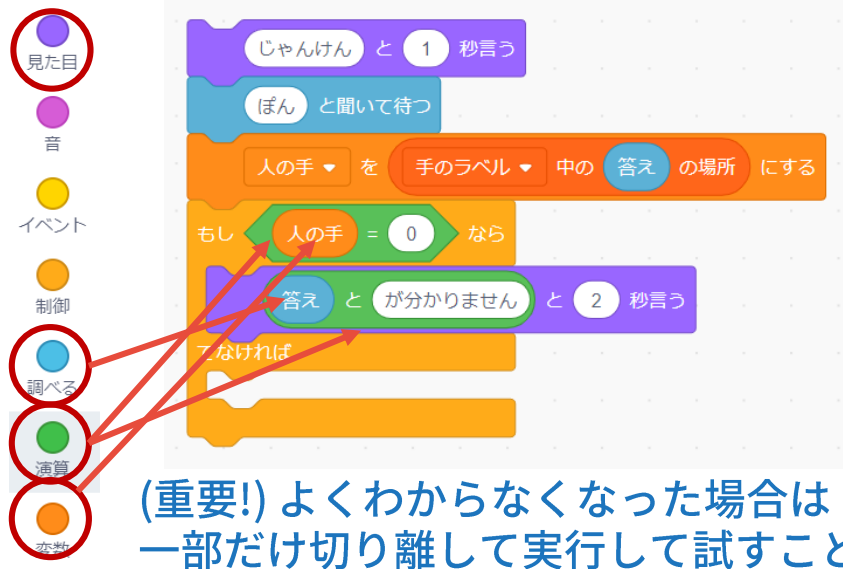
- プレイヤーの手が有効か否かを判断して条件分岐する
 - 無効な入力 (g, c, p 以外) ならメッセージを出そう
 - 有効な入力 (g, c, p いずれか) なら「コンピュータの手」を決めよう



条件分岐

12

- プレイヤーの手が有効か否かを判断して条件分岐する
 - 無効な入力 (g, c, p 以外) ならメッセージを出そう
 - 有効な入力 (g, c, p いずれか) なら「コンピュータの手」を決めよう



条件分岐

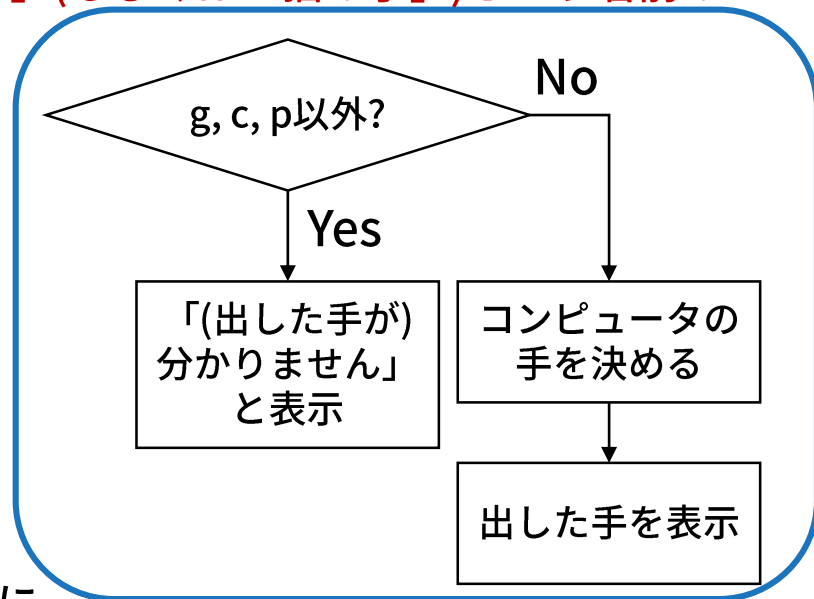
13

- プレイヤーの手が有効か否かを判断して条件分岐する
 - 無効な入力 (g, c, p 以外) ならメッセージを出そう
 - 有効な入力 (g, c, p いずれか) なら「コンピュータの手」を決めよう

「コンピュータの手」(もしくは「猫の手」)という名前の
変数を作成しておく



リストのレポーターブロック



1→g, 2→c, 3→p へ変換されて表示 (日本語にしたければ別途リストを作成しておけばよい)

勝負の判定

14

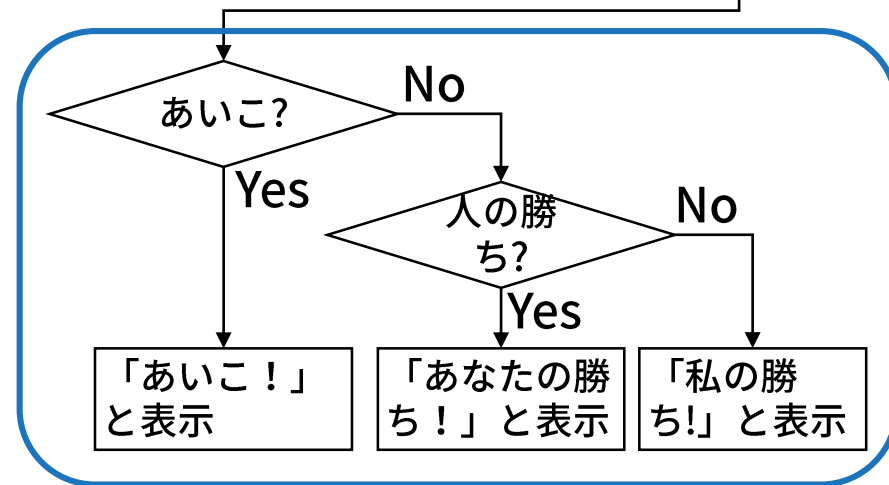
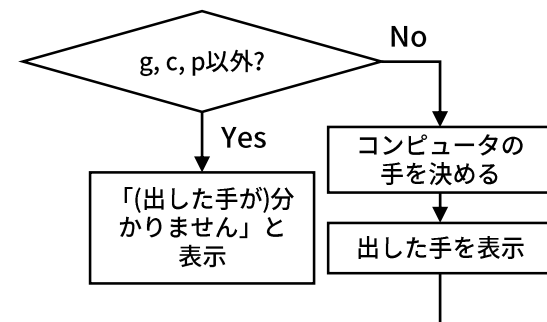
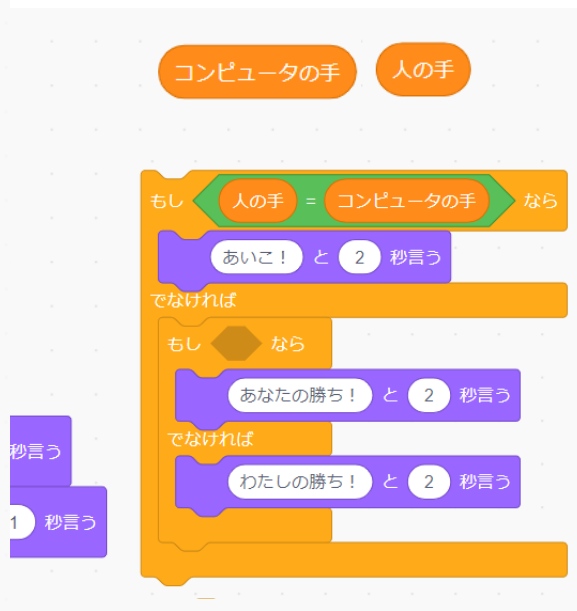
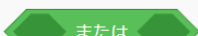
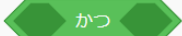
- 勝負の判定も条件分岐

- あいこか?
- あいこでないなら人(プレイヤー)の勝ちか?

動き
見た目
音
イベント
制御
調べる
演算
変数
ブロック定義



1 から 10 までの乱数



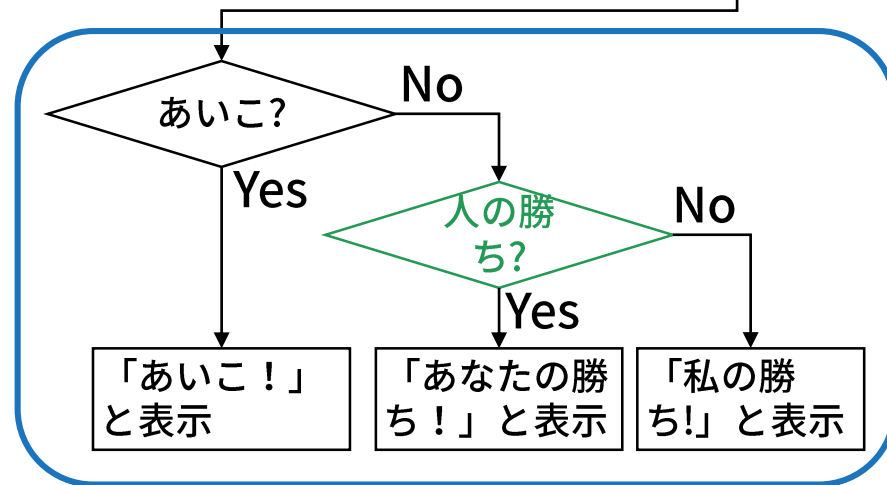
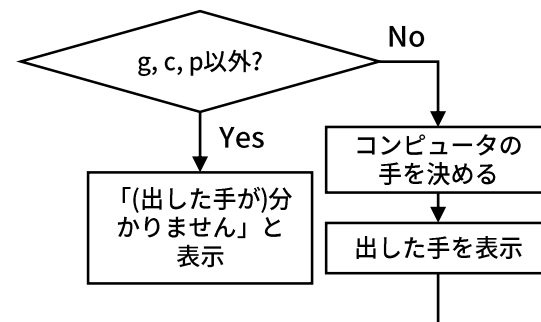
(重要!) いったん勝負の判定部だけ分けて作り、動作をテストする (単体テスト)

勝負の判定

15

- 人(プレイヤー)の勝ちはどう判定できる?

| 人の手 | コの手 |
|-------|-------|
| g (1) | c (2) |
| c (2) | p (3) |
| p (3) | g (1) |



その1



勝負の判定

16

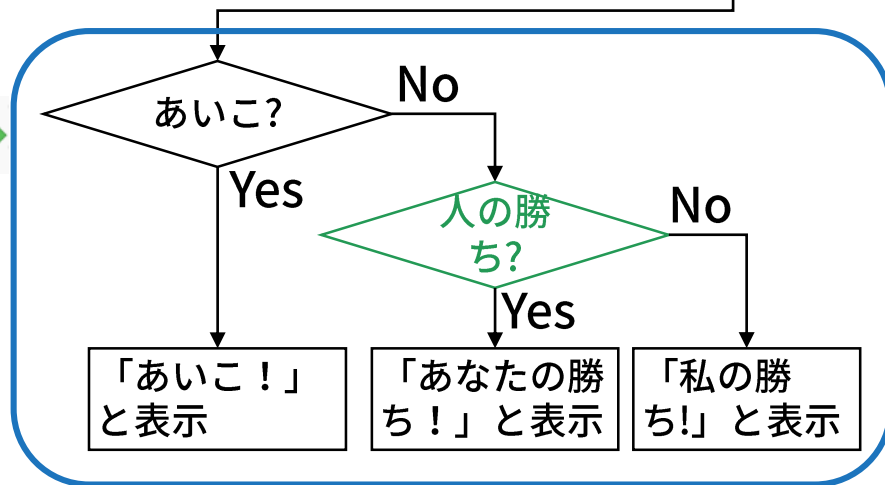
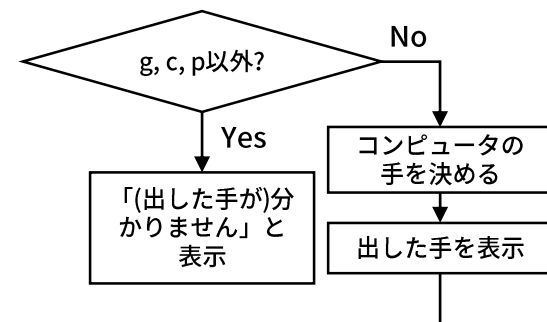
- もっと短く判定できる？

| 人の手 | コの手 |
|-------|-------|
| g (1) | c (2) |
| c (2) | p (3) |
| p (3) | g (1) |

その2



その3



定義 (独自ブロック, 関数)

17

- 一言で言い表せる処理のまとまりは新たなブロックとする
 - (例) 「判定結果を言う」の定義を作る



判定結果を言う部分

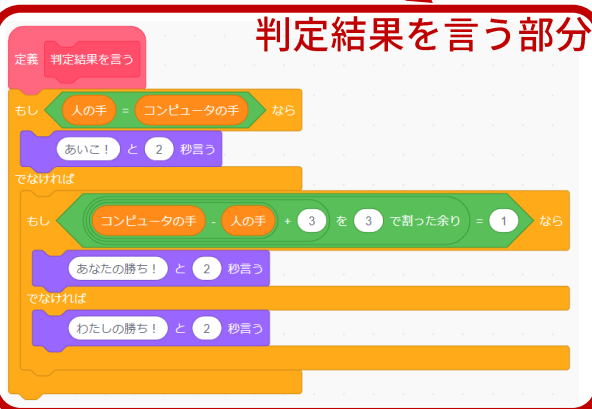


ブロックを作る

判定結果を言う



判定結果を言う部分



イベントでゲームを開始

18

- 緑の旗クリック時にゲームを開始させる
 - ちなみに赤丸で実行終了



イベント

動き

見た目

音

イベント

制御

調べる

演算

変数

ブロック定義

このスプライトが押されたとき

背景が 背景1 になったとき

音量 > 10 のとき

メッセージ1 を受け取ったとき

メッセージ1 を送る

メッセージ1 を送って待つ

が押されたとき

じゃんけん と 1 秒言う

ぼん と聞いて待つ

人の手 を 手のラベル 中の 答え の場所 にする

もし 人の手 = 0 なら

答え と が分かりません と 2 秒言う

でなければ

コンピュータの手 を 1 から 3 までの乱数 にする

あなたの手は と 手のラベル の 人の手 番目 と 1 秒言う

私の手 と 手のラベル の コンピュータの手 番目 と 1 秒言う

判定結果を言う

定義 判定結果を言う

もし 人の手 = コンピュータの手 なら

あいこ! と 2 秒言う

でなければ

もし $\text{コンピュータの手} - \text{人の手} + 3$ を 3 で割った余り = 1 なら

あなたの勝ち! と 2 秒言う

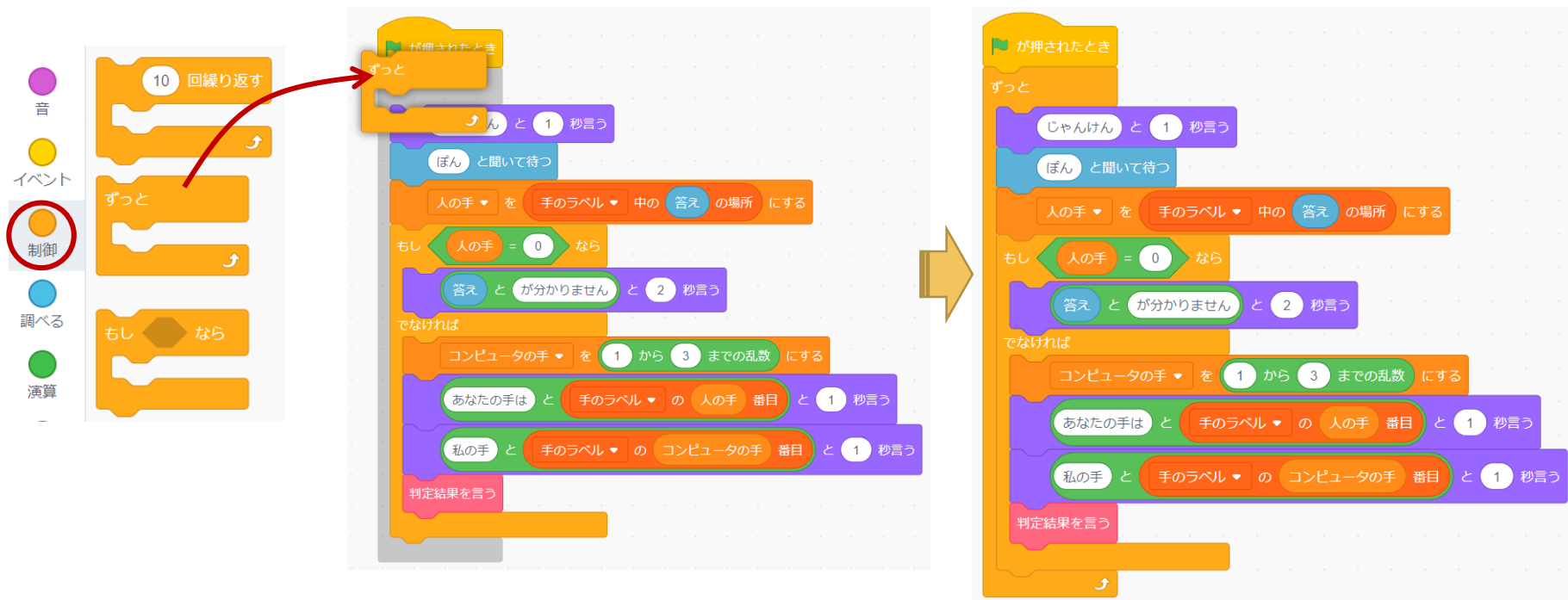
でなければ

わたしの勝ち! と 2 秒言う

ループ

19

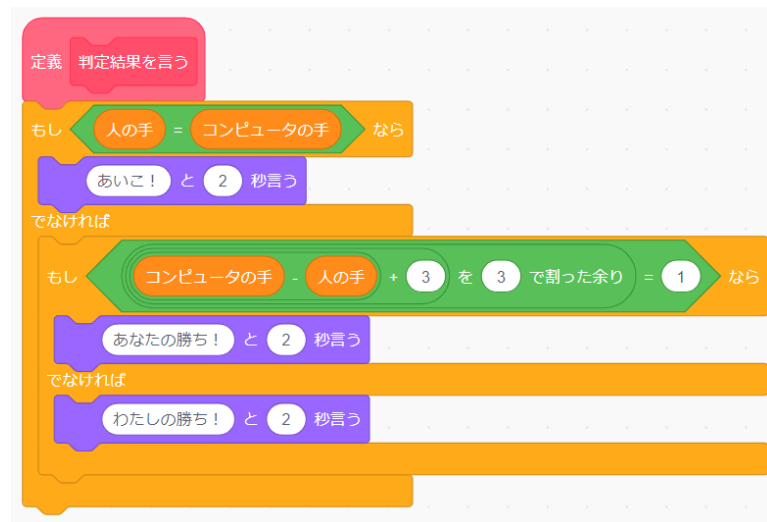
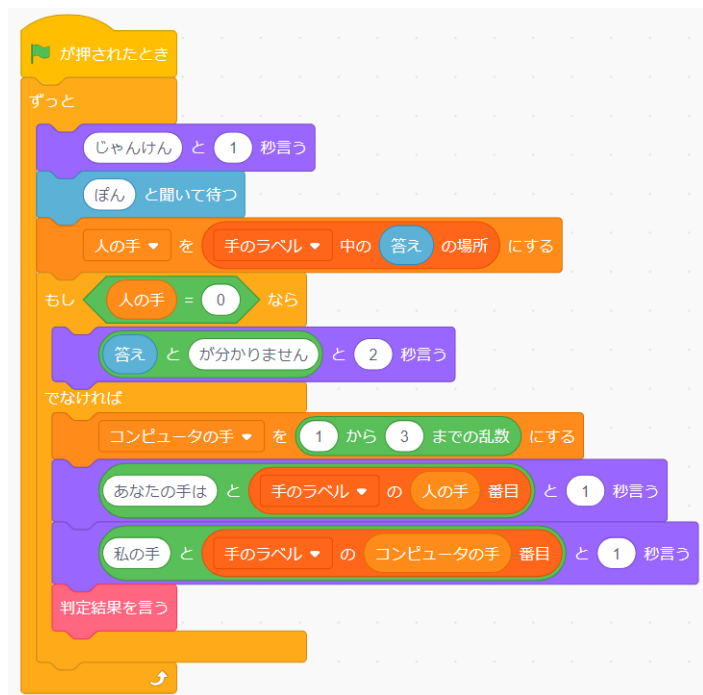
- ・ 緑の旗を毎回押さずともじゃんけんを繰り返すにはループで
 - ・ 無限ループを使ってみる（回数指定のループもある）



じゃんけんプログラム完成

20

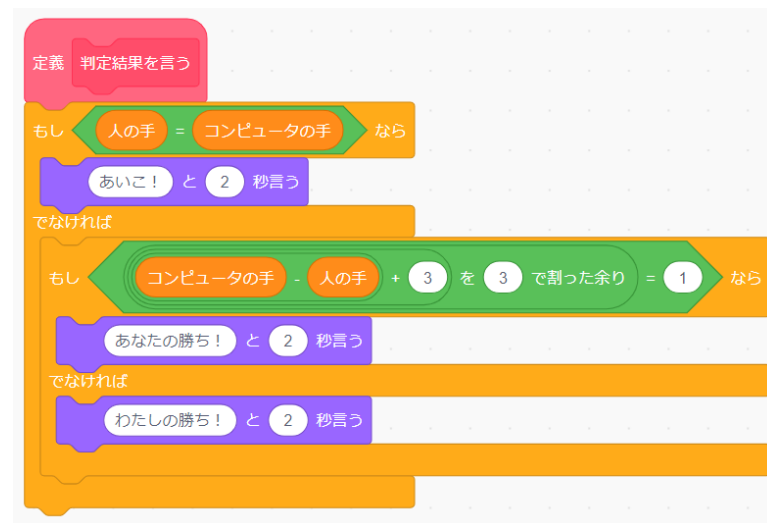
- 条件分岐，ループ (繰り返し)，変数，リスト，乱数，四則・剰余演算，定義，入出力など，重要な要素を含むプログラム
 - 「グー」「チョキ」「パー」の手の絵を表示するなどの拡張も容易



- 人(プレイヤー)の手の傾向を見るために、過去に出した手の記録 (ログ) を取る

どの箇所でログを取ればよいだろう？
どうやって取ればよいだろう？

リストを利用可能



リストの操作

22

- Scratchのリストは連結リストなので追加/挿入/削除が簡単

「テスト」という名前のリストを作りブロックの動作を確認しておこう

- 100, 200, 300, 400 までブロックを使って順に追加
- 1番目を123に置き換え
- 2番目を削除
- 2番目に789を挿入
- いったん全て削除
- 1から3の乱数を200個追加
- いったん全て削除
- 2から200までの偶数を追加
- リストの要素の総和を計算

操作対象のリストを選ぶのを忘れずに

The image shows five Scratch code blocks for list operations on a list named 'テスト' (Test):

- Block 1: 'なにか' (Anything) を 'テスト' に追加する (Add to 'Test').
- Block 2: 'テスト' の 1 番目を削除する (Delete the 1st item of 'Test').
- Block 3: 'テスト' のすべてを削除する (Delete all of 'Test').
- Block 4: 'テスト' の 1 番目に 'なにか' を挿入する (Insert 'Anything' at the 1st position of 'Test').
- Block 5: 'テスト' の 1 番目を 'なにか' で置き換える (Replace the 1st item of 'Test' with 'Anything').

| index | 要素 |
|-------|-----|
| 1 | 100 |
| 2 | 200 |
| 3 | 300 |
| 4 | 400 |

リストの操作

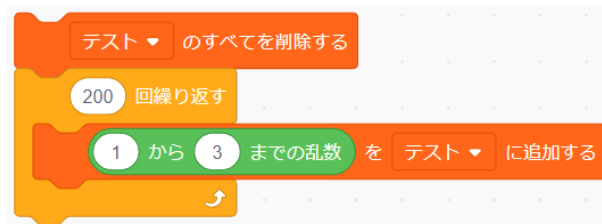
23

- リストはループ (繰り返し) と相性がいい

練習続き...

5. いったん全て削除
6. 1から3の乱数を200個追加
7. いったん全て削除
8. 2から200までの偶数を追加
9. リストの要素の総和を計算

5, 6



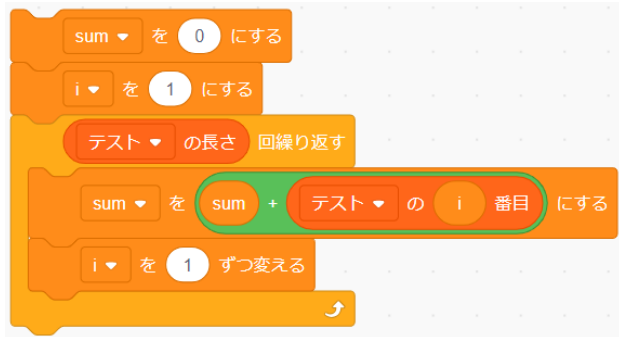
7, 8

変換「x」を
作成しておく



9

変換「i」「sum」
を作成しておく

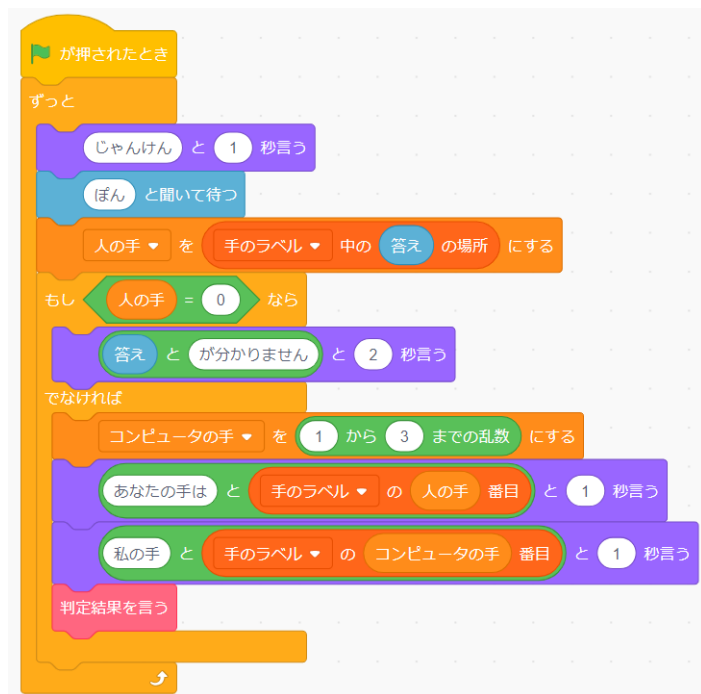


7, 8の別解



- 人(プレイヤー)の手の傾向を見るために、過去に出した手の記録(ログ)を取る

どの箇所でログを取ればよいだろう?
どうやって取ればよいだろう?



- ① 「手のログ」という名前のリストを作成



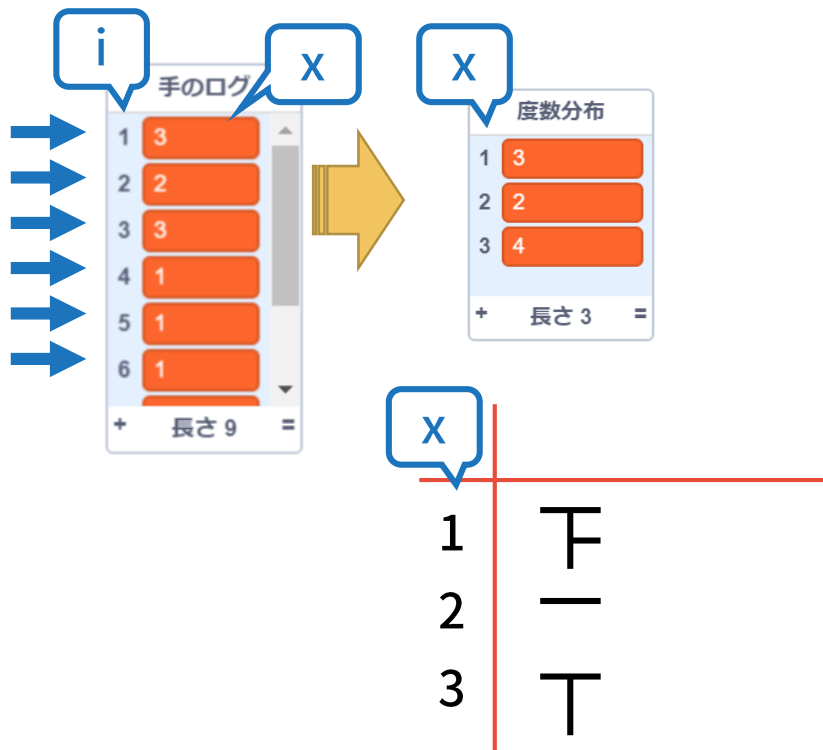
最初にログの初期化をするならこれを入れる(今は入れなくても可)

- ② 人の出した手をログに追加(リスト名注意)

度数分布

25

- 人(プレイヤー)の各手の出現回数を度数分布にまとめる
 - ログを順にたどって各手の回数を1ずつ増やしていく



最頻手 (最大値) を求める

26

- 度数分布の最大値をとる手が最頻手
- 最大値を見つけるアルゴリズム (手順)
 - 変数max を(リストの値より小さい)-1 とする
 - リスト (度数分布)の要素を順にたどり
もし要素の値が max よりも大きければ
max にその値を入れる (その時点での最大値)
 - リストのすべての要素をたどれば終了



様々なアルゴリズムが「アルゴリズムとデータ構造」で紹介

仮説検定を行う

27

- 適合度検定で想定と異なる分布を見分ける
 - (例) 「グー: $1/3$, チョキ: $1/3$, パー: $1/3$ 」
 - 帰無仮説: 全て同じ確率 $1/3$, 対立仮説: 偏りがある

「期待度数」「カイ二乗値」という名前の変数を追加しておく



この時だけ偏りを考慮して手を決める?

仮説検定の詳細は「確率・統計」(および「統計学」)にて

- 簡単な拡張
 - じゃんけんの手をアイコンや絵で表示
 - 勝ち/あいこ/負けの回数を変数で記録
 - 勝率を表示
- やや高度な拡張 (このあたりから Scratch だとやや面倒)
 - 可視化
 - 度数分布をヒストグラムで表示
 - 1つ前の手と現在の手の相関を散布図で表示
 - 機械学習
 - 過去2～3回の手に対して次の手はどう決まるか傾向を学習
 - 勝負結果によって出す手はどう変わるかを学習

プログラムの保存方法

29

- ファイル＞コンピュータに保存する
 - 「Scratchのプロジェクト.sb3」というファイルに保存されます
- 次回続けるとき：ファイル＞コンピュータから読み込む
 - 保存したファイルを選択してください
- もし、オンライン版を「サインイン」して使っている場合は、オンライン上に保存することも可能です