micro:bit で学ぶプログラミング
～ブロックプログラム集～

目次

【注意事項】

ブロックプログラムの印刷は、「歯車」のアイコンから印刷（図 1）を選択すると、「プログラムを印刷する」のダイアログが表示されるので、プリンタ（図２）を選択すると印刷することができます。

なお、ここでは、1 章～5 章のプログラムを作成し、載せています。教科書のブロックのプログラムと比較してみてください。

図 1　印刷の選択

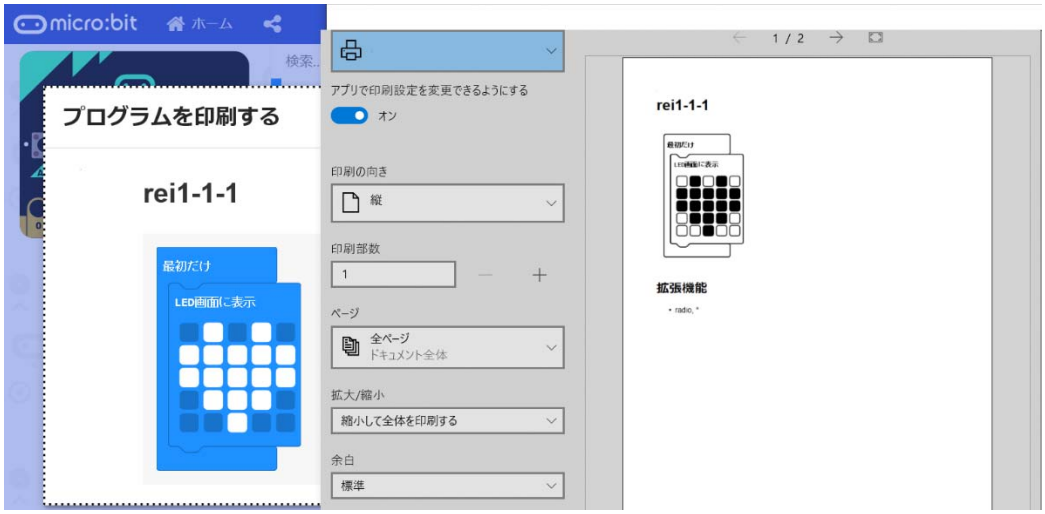


図 2　プリンタの選択

１．プログラミングの基礎

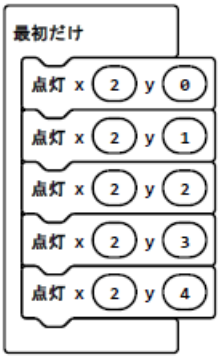
1.1　micro:bit の基本操作

rei1-1-1

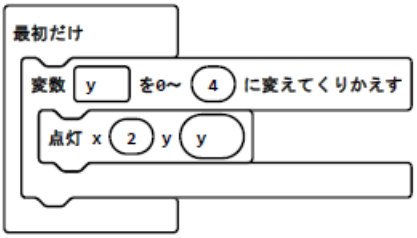
rei1-1-2

1.2　プログラムの基礎（順次，繰返し）

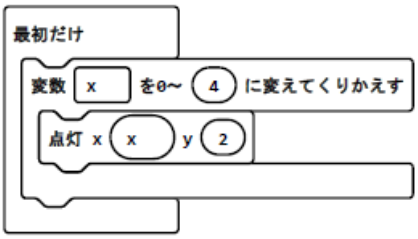
rei1-2



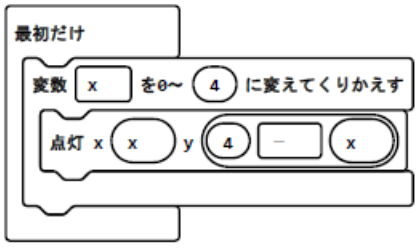
rei1-3



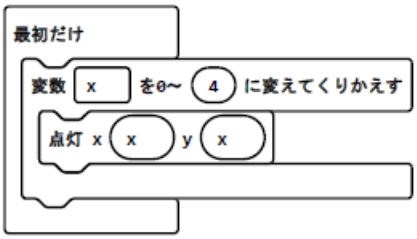
ren1-1



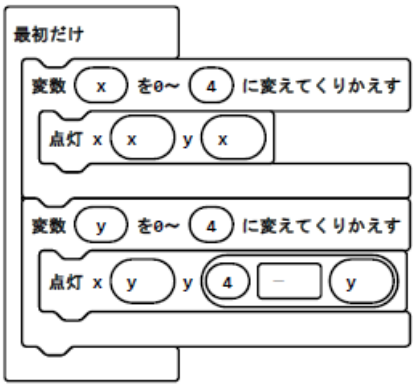
rei1-4



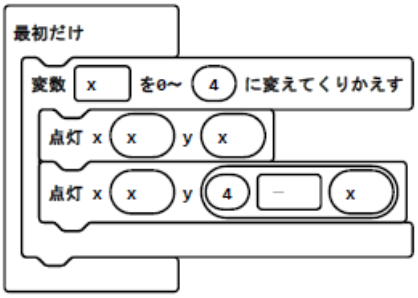
ren1-2



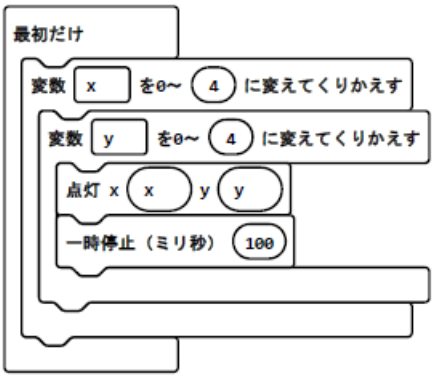
ren1-3-1



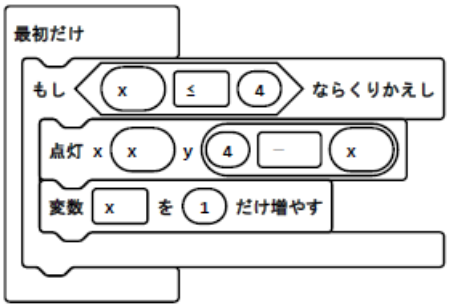
ren1-3-2



rei1-5

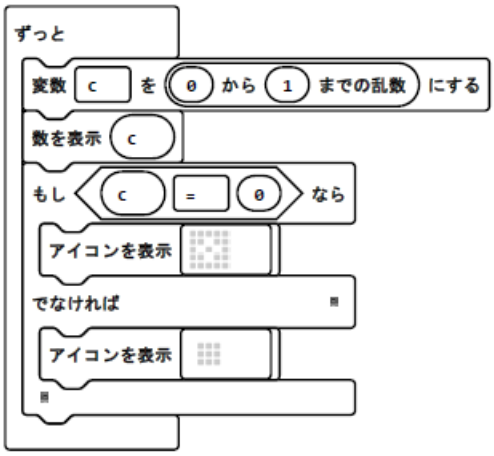


rei1-6

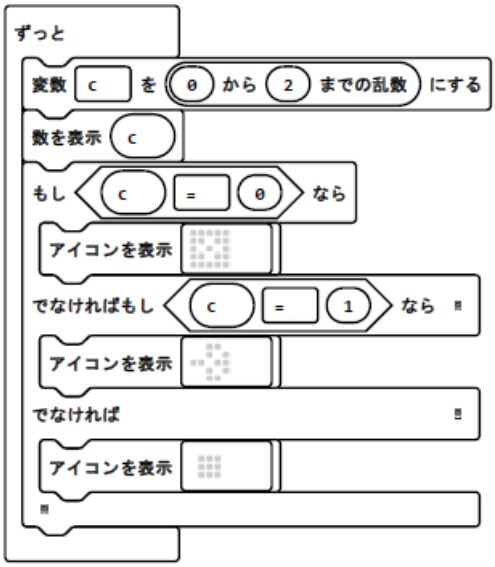


1.3　プログラムの基礎（分岐）

rei1-7

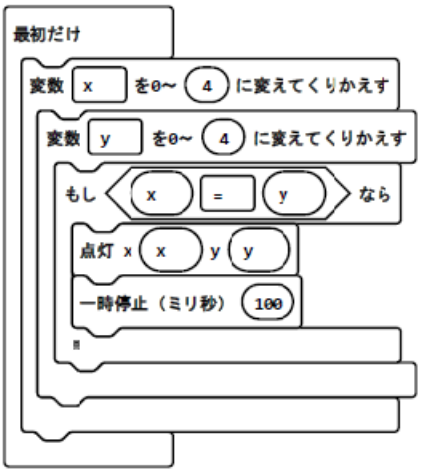


rei1-8

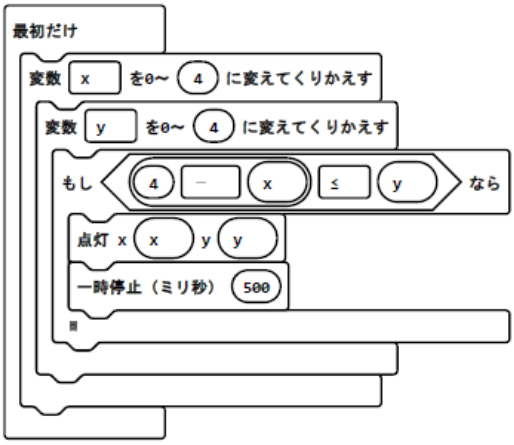


演習問題

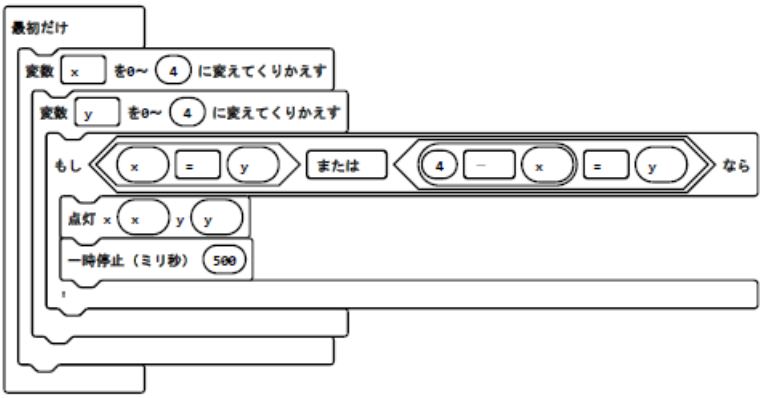
ens1-1



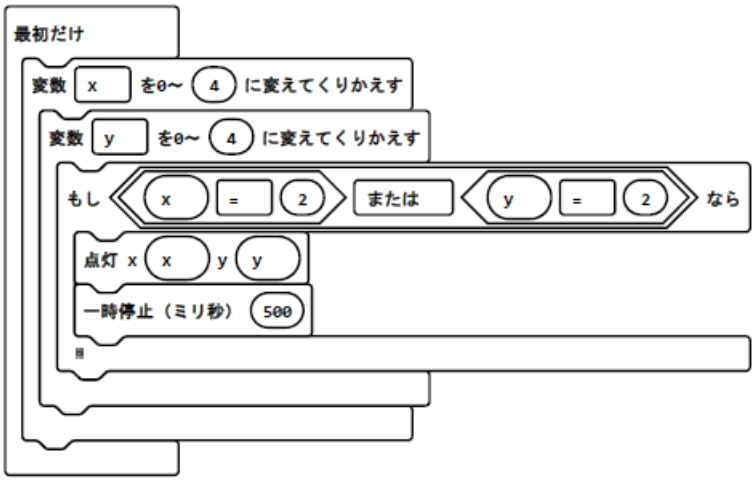
ens1-2-1



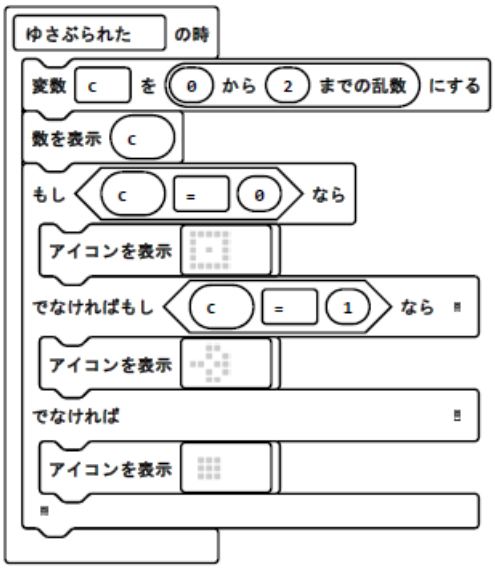
ens1-2-2



ens1-2-3



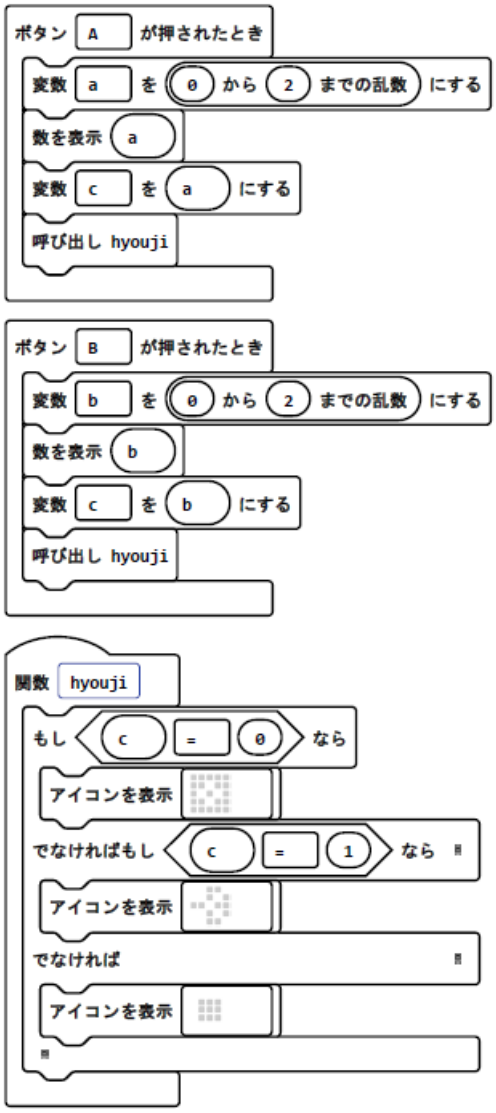
ens1-3


ゆさぶられた
の時
変数
c
を
0
から
2
までの乱数
にする
数を表示
c
もし
c
=
0
なら
アイコンを表示
でなければもし
c
=
1
なら
アイコンを表示
でなければ
アイコンを表示

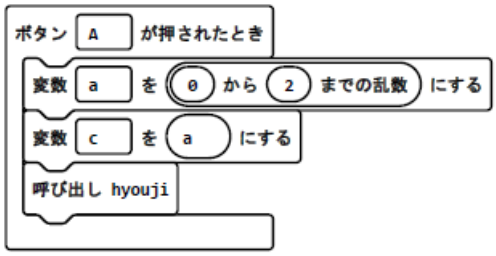
２．プログラミングの応用（関数，配列）

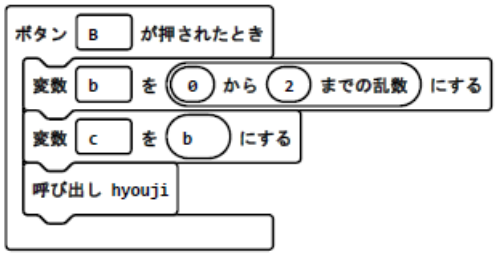
2.1　じゃんけんゲーム

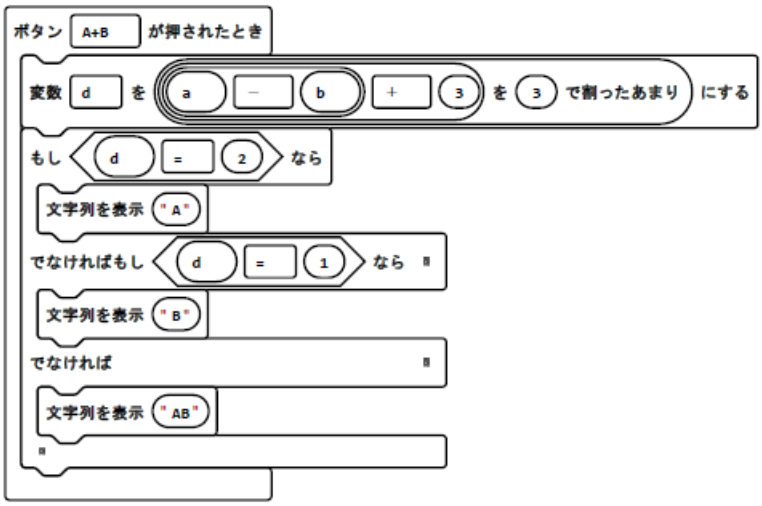
rei2-1

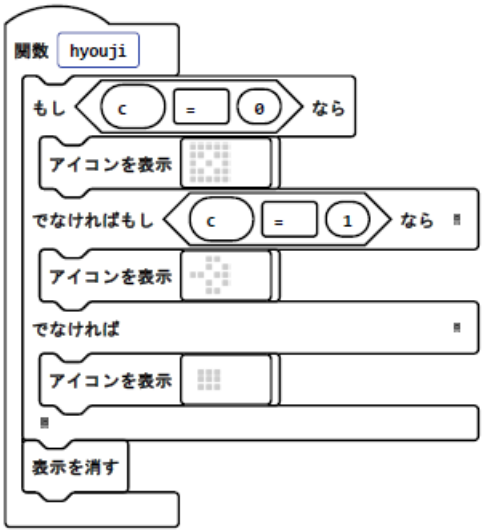


rei2-2

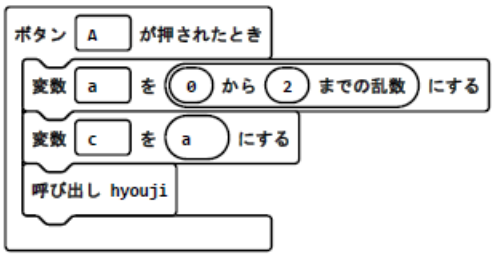

ボタン A が押されたとき
変数 a を 0 から 2 までの乱数 にする
変数 c を a にする
呼び出し hyouji

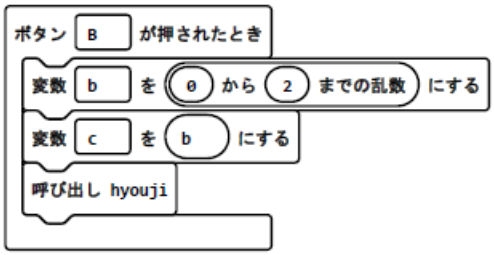

ボタン B が押されたとき
変数 b を 0 から 2 までの乱数 にする
変数 c を b にする
呼び出し hyouji

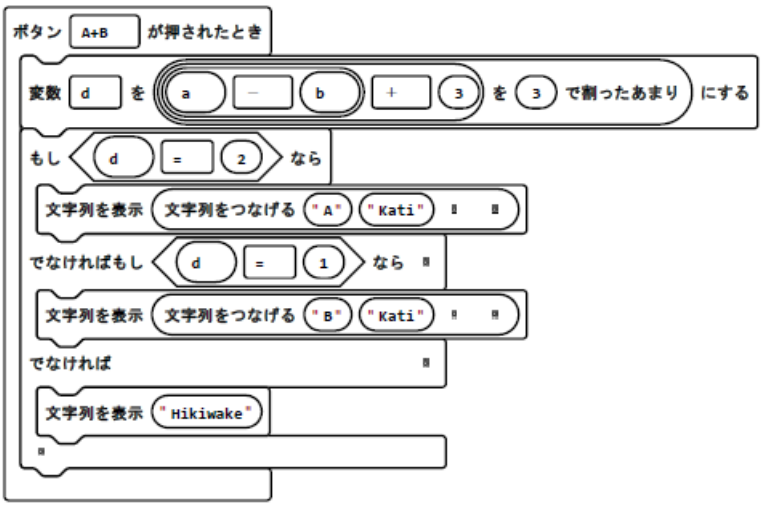

ボタン A+B が押されたとき
変数 d を a − b + 3 を 3 で割ったあまり にする
もし d = 2 なら
文字列を表示 "A"
でなければもし d = 1 なら
文字列を表示 "B"
でなければ
文字列を表示 "AB"

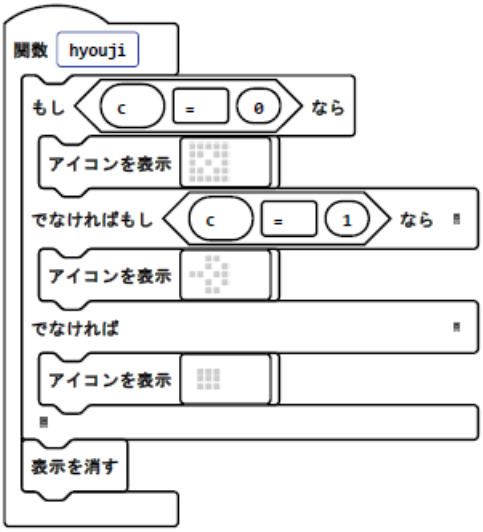

関数 hyouji
もし c = 0 なら
アイコンを表示
でなければもし c = 1 なら
アイコンを表示
でなければ
アイコンを表示
表示を消す

ren2-1


ボタン A が押されたとき
変数 a を 0 から 2 までの乱数 にする
変数 c を a にする
呼び出し hyouji

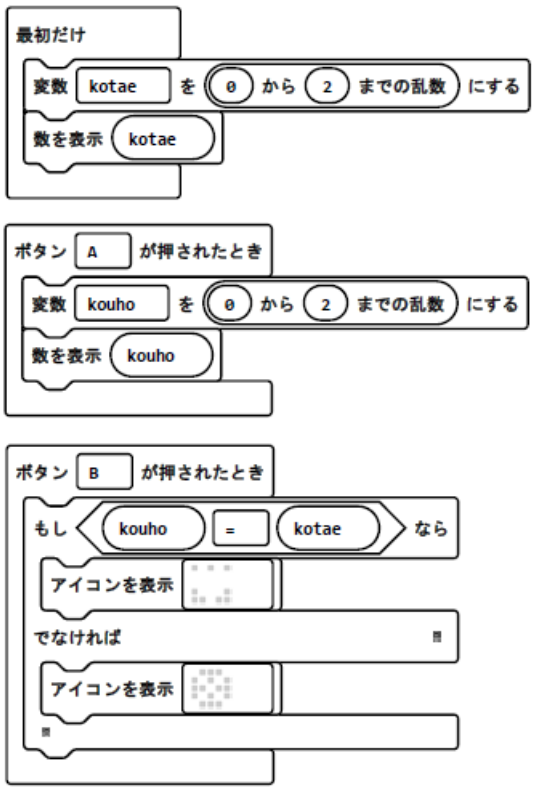

ボタン B が押されたとき
変数 b を 0 から 2 までの乱数 にする
変数 c を b にする
呼び出し hyouji


ボタン A+B が押されたとき
変数 d を a − b + 3 を 3 で割ったあまり にする
もし d = 2 なら
文字列を表示 文字列をつなげる "A" "Kati"
でなければもし d = 1 なら
文字列を表示 文字列をつなげる "B" "Kati"
でなければ
文字列を表示 "Hikiwake"

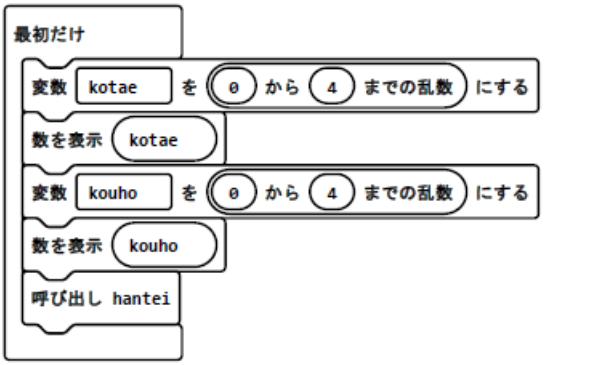

関数 hyouji
もし c = 0 なら
アイコンを表示
でなければもし c = 1 なら
アイコンを表示
でなければ
アイコンを表示
表示を消す

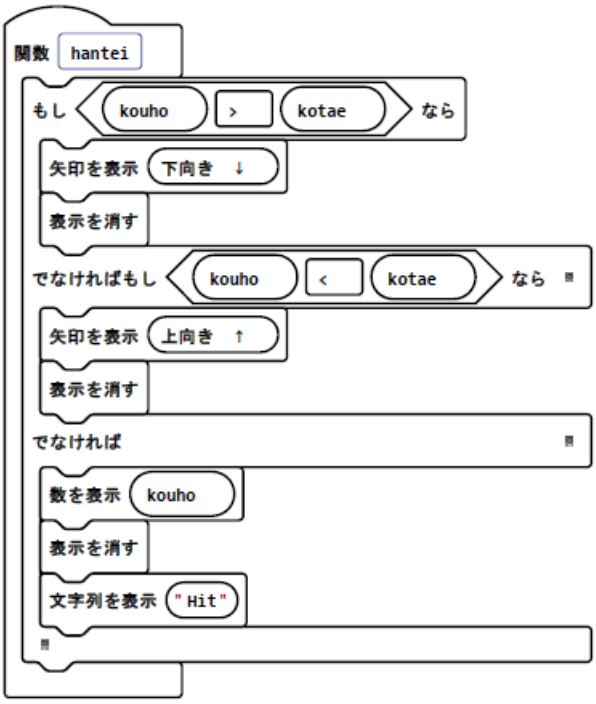
2.2　数あてゲーム

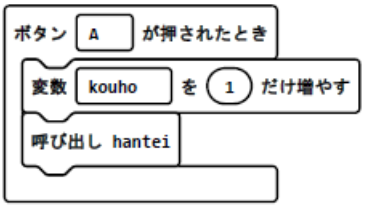
rei2-3

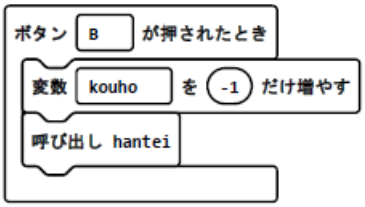


rei2-4

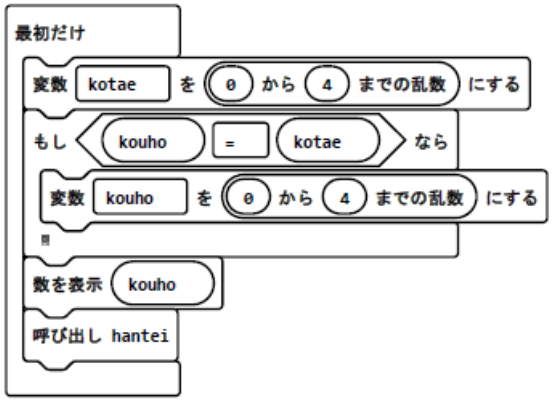

最初だけ
変数 kotae を 0 から 4 までの乱数 にする
数を表示 kotae
変数 kouho を 0 から 4 までの乱数 にする
数を表示 kouho
呼び出し hantei

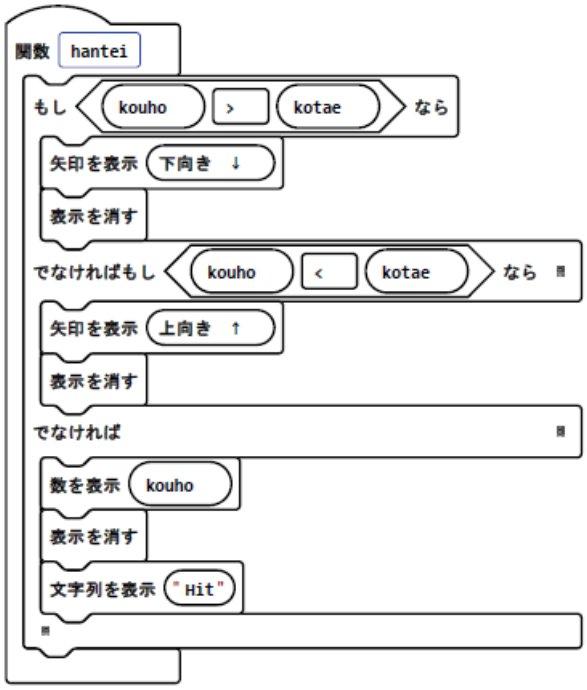

関数 hantei
もし kouho > kotae なら
矢印を表示 下向き ↓
表示を消す
でなければもし kouho < kotae なら
矢印を表示 上向き ↑
表示を消す
でなければ
数を表示 kouho
表示を消す
文字列を表示 "Hit"

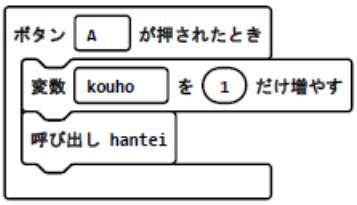

ボタン A が押されたとき
変数 kouho を 1 だけ増やす
呼び出し hantei


ボタン B が押されたとき
変数 kouho を -1 だけ増やす
呼び出し hantei

ren2-2


最初だけ
変数 kotae を 0 から 4 までの乱数 にする
もし kouho = kotae なら
変数 kouho を 0 から 4 までの乱数 にする
数を表示 kouho
呼び出し hantei

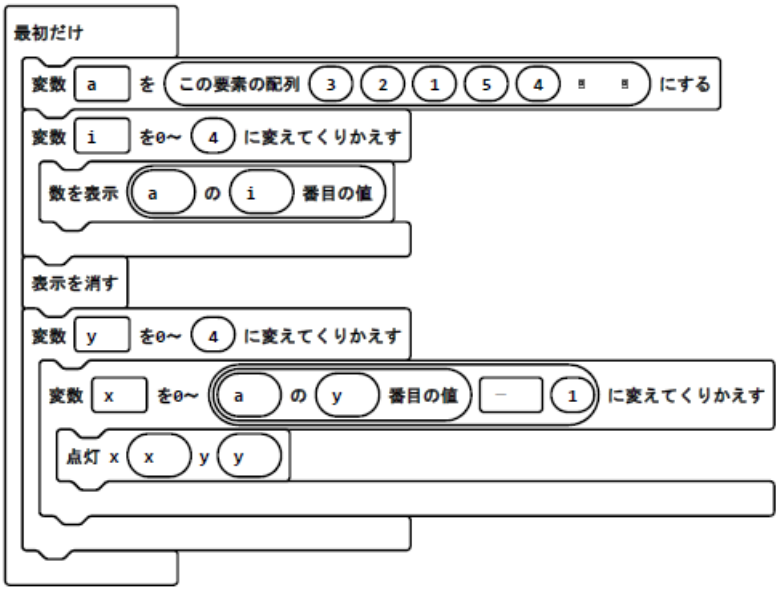

関数 hantei
もし kouho > kotae なら
矢印を表示 下向き ↓
表示を消す
でなければもし kouho < kotae なら
矢印を表示 上向き ↑
表示を消す
でなければ
数を表示 kouho
表示を消す
文字列を表示 "Hit"


ボタン A が押されたとき
変数 kouho を 1 だけ増やす
呼び出し hantei

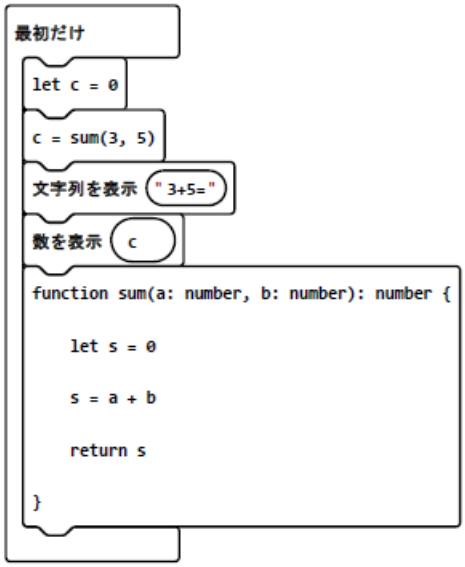

ボタン B が押されたとき
変数 kouho を -1 だけ増やす
呼び出し hantei

2.3　グラフの作成

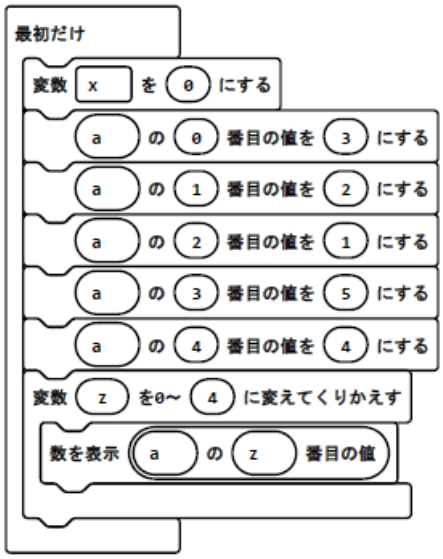
rei2-5



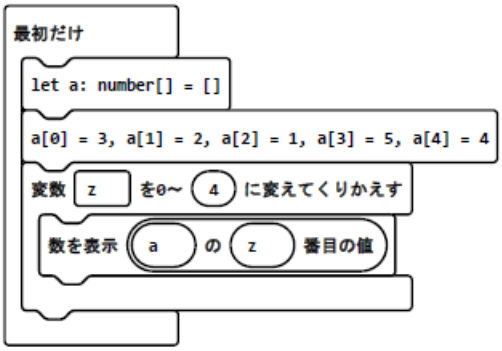
c23-kansu



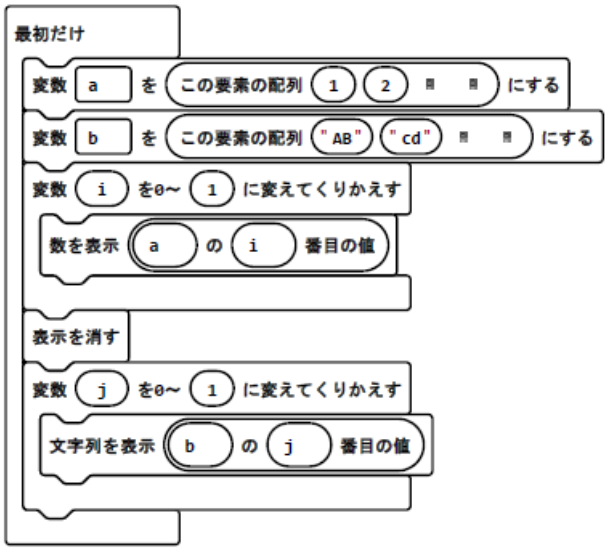
ren2-3-1



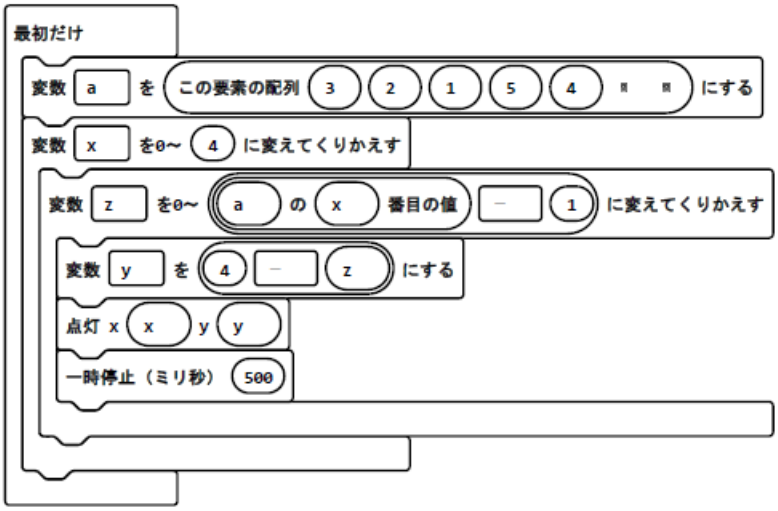
ren2-3-2



ren2-4

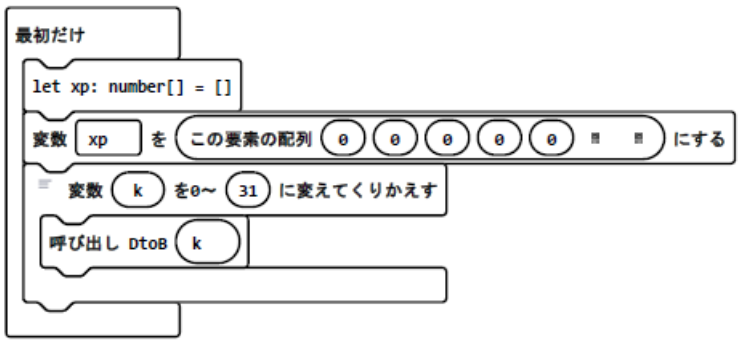

最初だけ
変数 a を この要素の配列 1 2 にする
変数 b を この要素の配列 "AB" "cd" にする
変数 i を0～ 1 に変えてくりかえす
数を表示 a の i 番目の値
表示を消す
変数 j を0～ 1 に変えてくりかえす
文字列を表示 b の j 番目の値

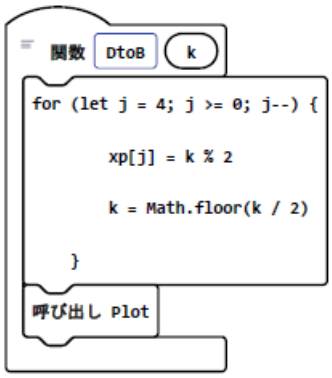
rei2-6

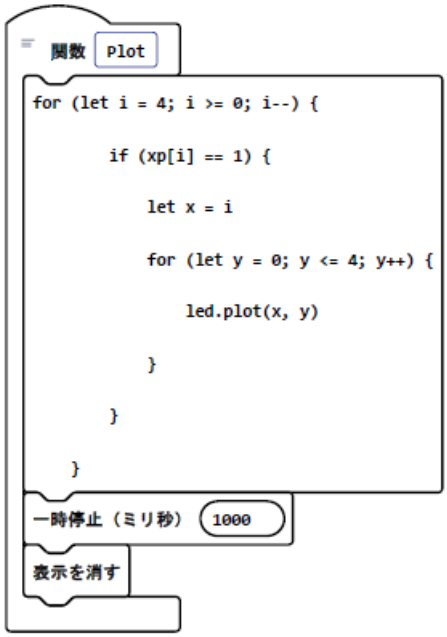

最初だけ
変数 a を この要素の配列 3 2 1 5 4 にする
変数 x を0～ 4 に変えてくりかえす
変数 z を0～ a の x 番目の値 − 1 に変えてくりかえす
変数 y を 4 − z にする
点灯 x x y y
一時停止（ミリ秒） 500

2.4　10 進数から 2 進数への変換

rei2-7

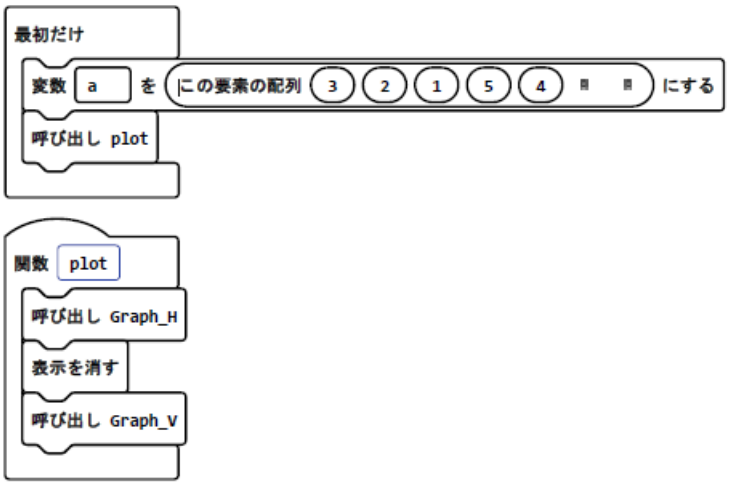




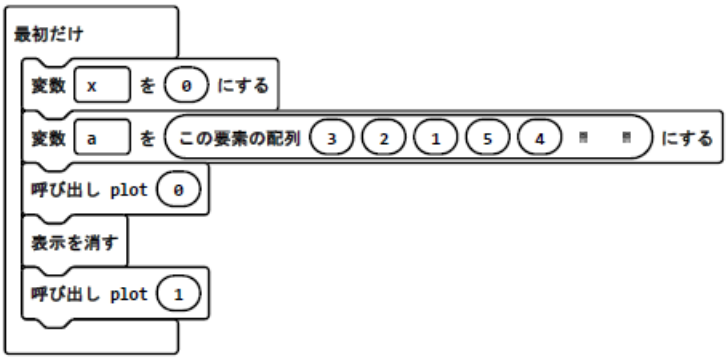


演習問題

ens2-1



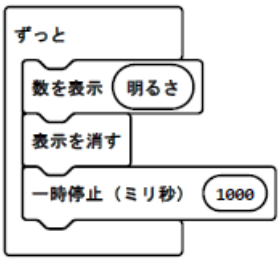
ens2-2



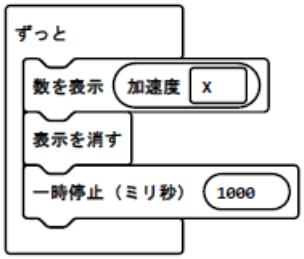
３．センサによる計測・制御プログラム

3.1　micro:bit の各種センサと制御

rei3-1

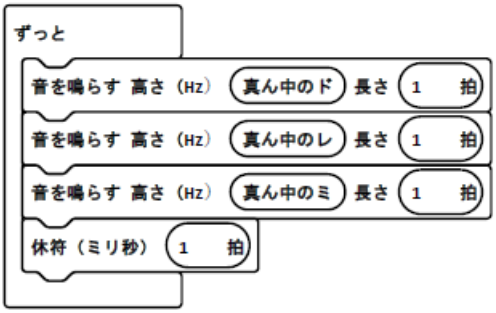


ren3-1

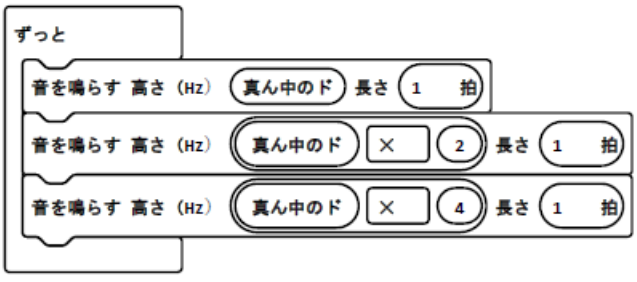


3.2　音センサによる音の制御

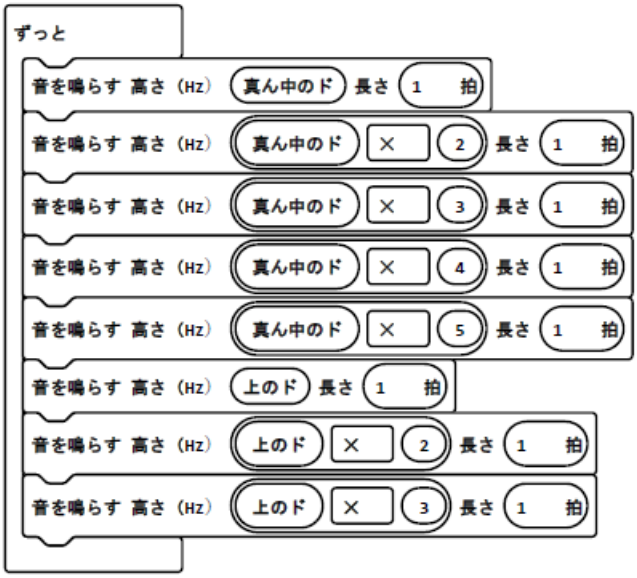
rei3-2



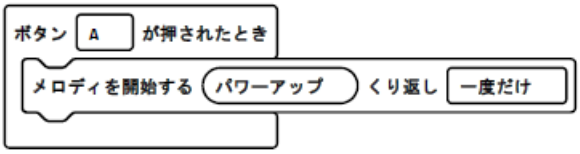
ren3-2



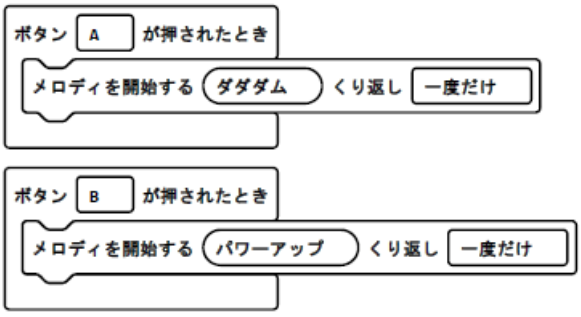
ren3-3



rei3-3

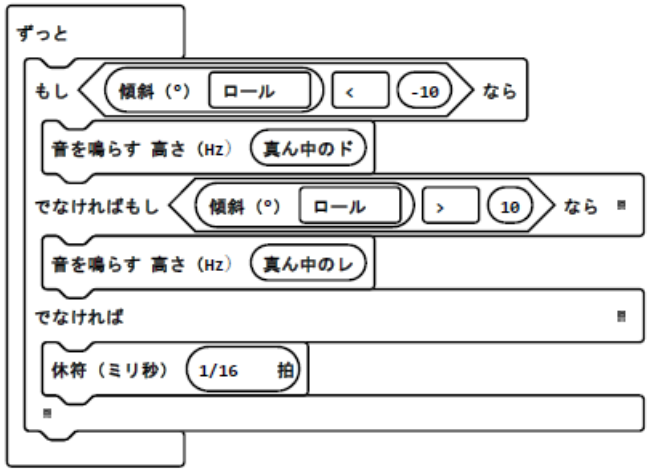


ren3-4



3.3　傾きセンサを使った計測・制御

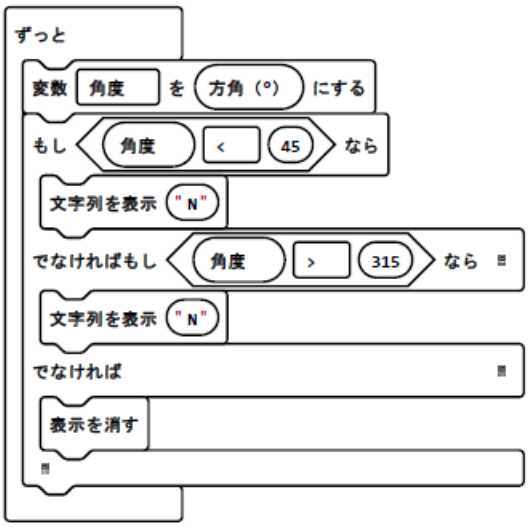
rei3-4



ren3-5

3.4　地磁気センサを使った計測・制御

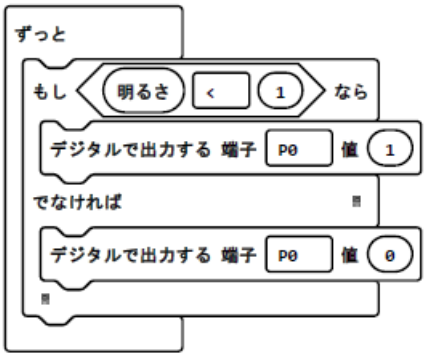
rei3-5



rei3-6

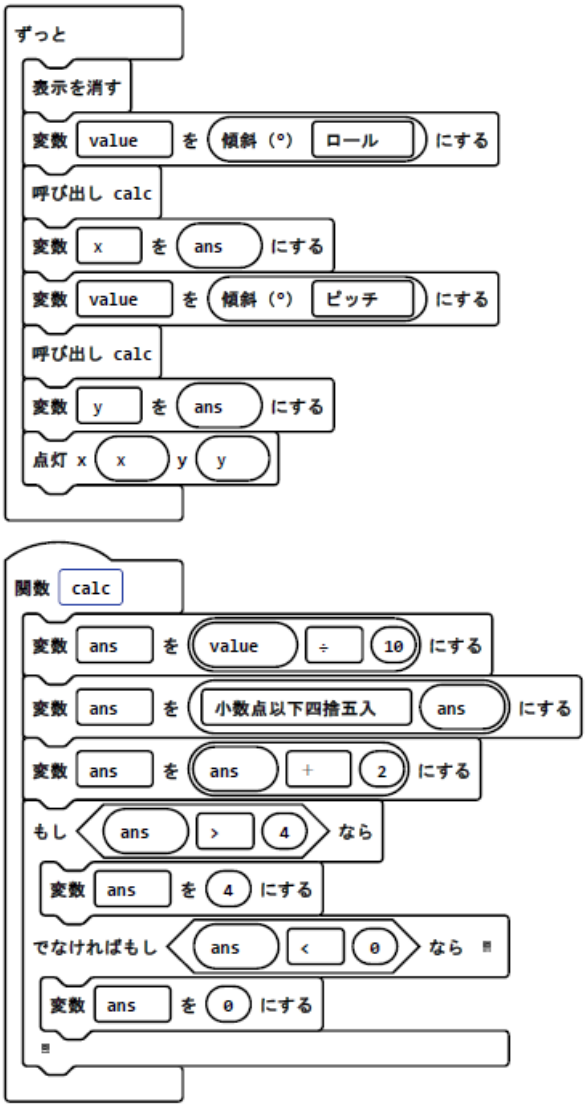
3.5　光センサを使った計測・制御

rei3-7

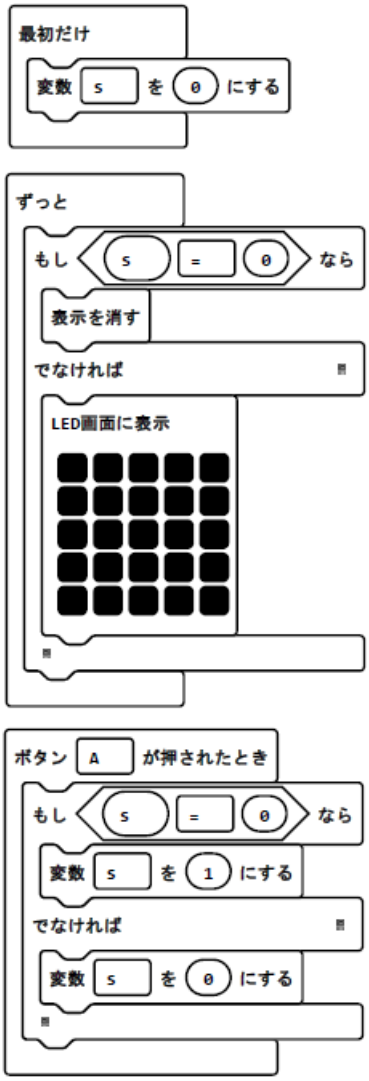


演習問題

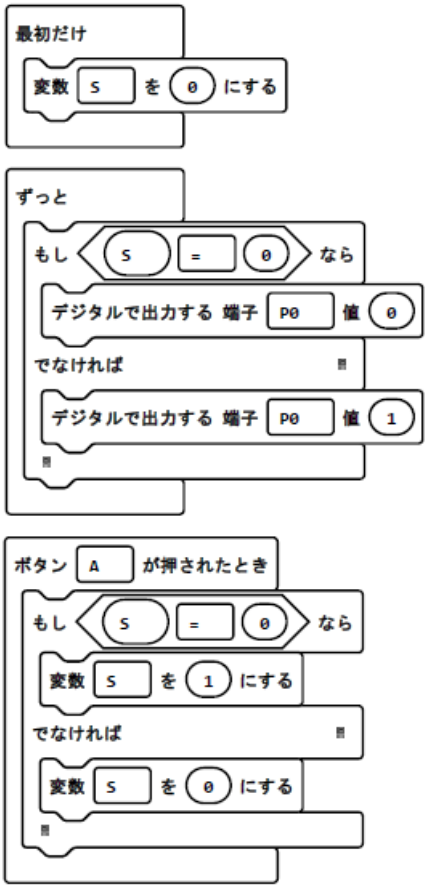
ens3-1


ずっと
表示を消す
変数 value を 傾斜 (°) ロール にする
呼び出し calc
変数 x を ans にする
変数 value を 傾斜 (°) ピッチ にする
呼び出し calc
変数 y を ans にする
点灯 x x y y
関数 calc
変数 ans を value ÷ 10 にする
変数 ans を 小数点以下四捨五入 ans にする
変数 ans を ans + 2 にする
もし ans > 4 なら
変数 ans を 4 にする
でなければもし ans < 0 なら
変数 ans を 0 にする

ens3-2



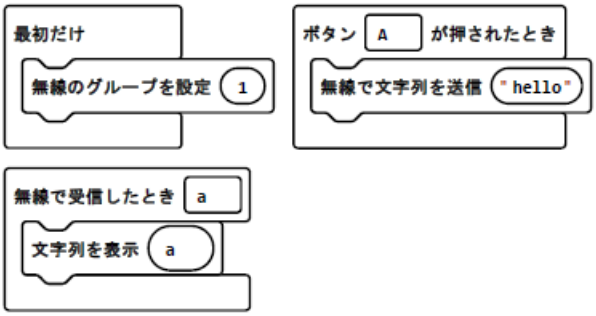
c-ens3-2



４．無線通信を利用したプログラム

4.1　無線通信の利用

rei4-1

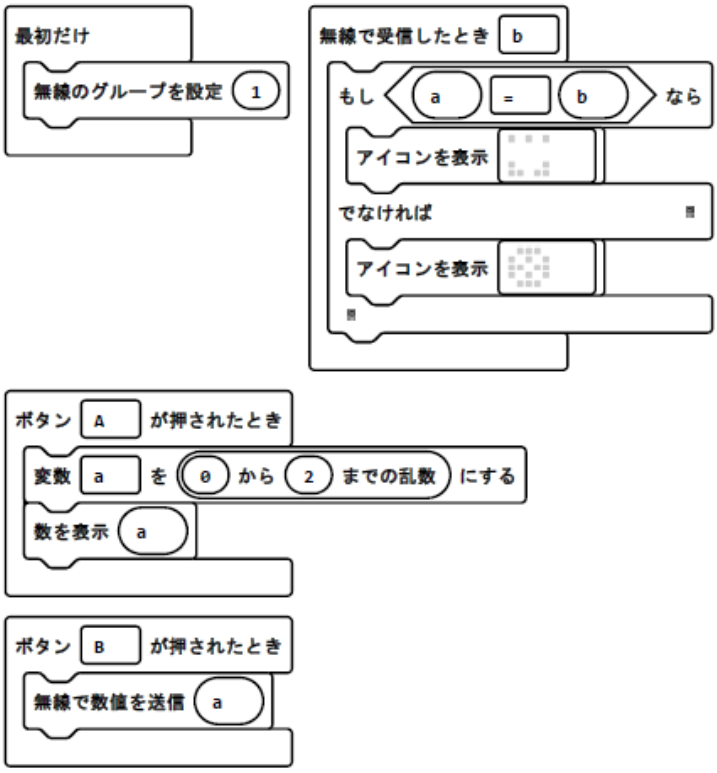


【仕様変更に伴う注意】例題 4-1～例題 4-3

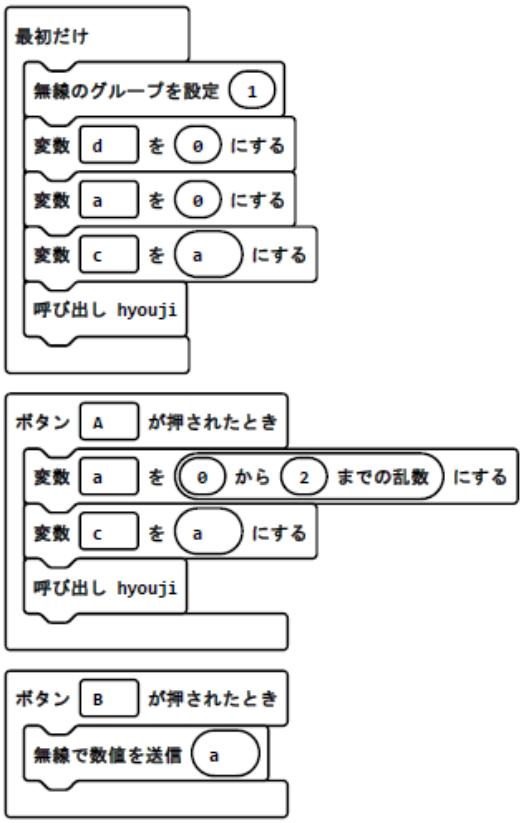
プログラムを Web サイトからダウンロードした場合は、そのまま実行できるが、新しくプログラムを作成する場合は、「無線を受信したとき」の変数が変更できなくなっているので、「a」や「b」に変更しないで、「receivedString」のまま使用する。

4.2　無線通信を利用したじゃんけんゲーム

rei4-2

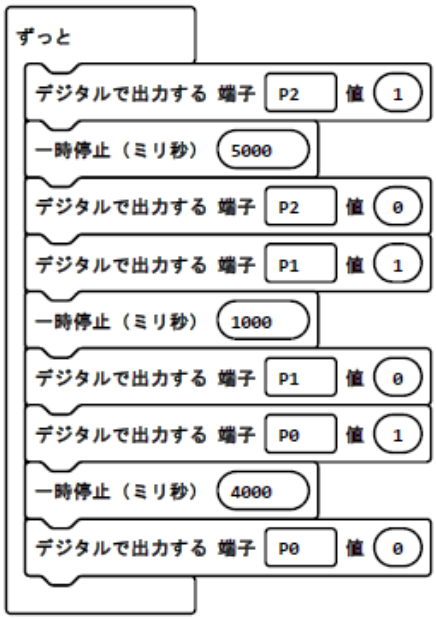


rei4-3

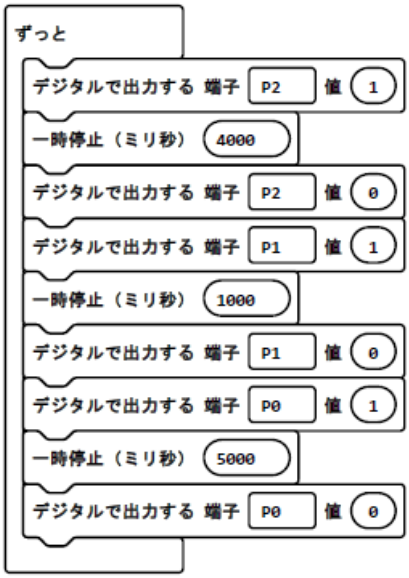


4.3　信号機の制御

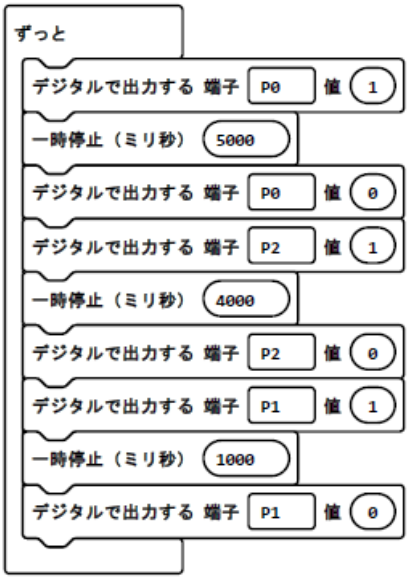
rei4-4



ren4-2-1

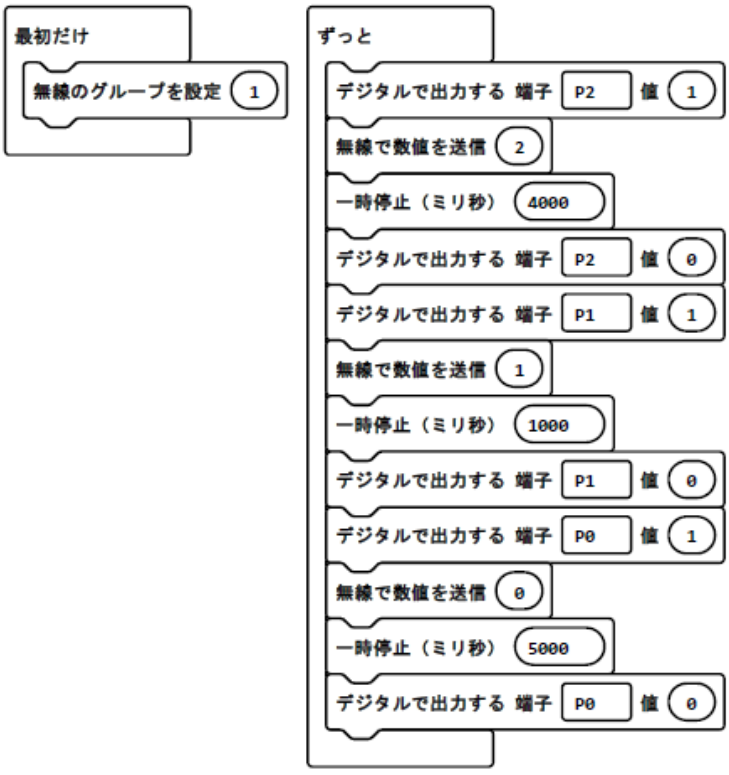


ren4-2-2

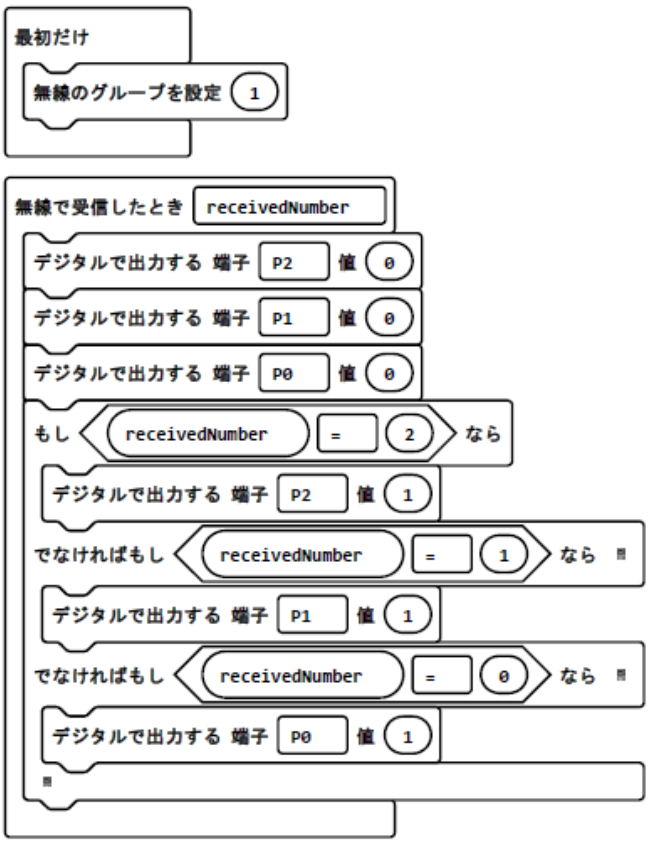


4.4　通信による信号機の制御

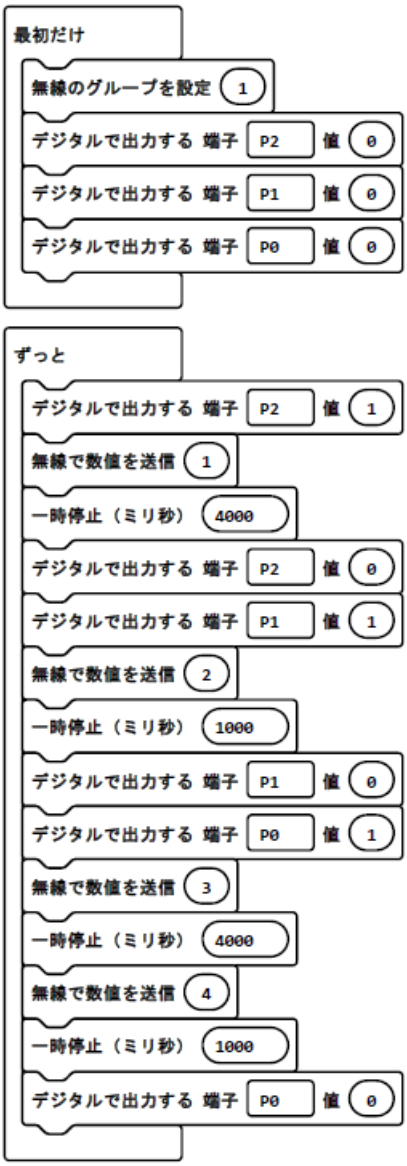
rei4-5



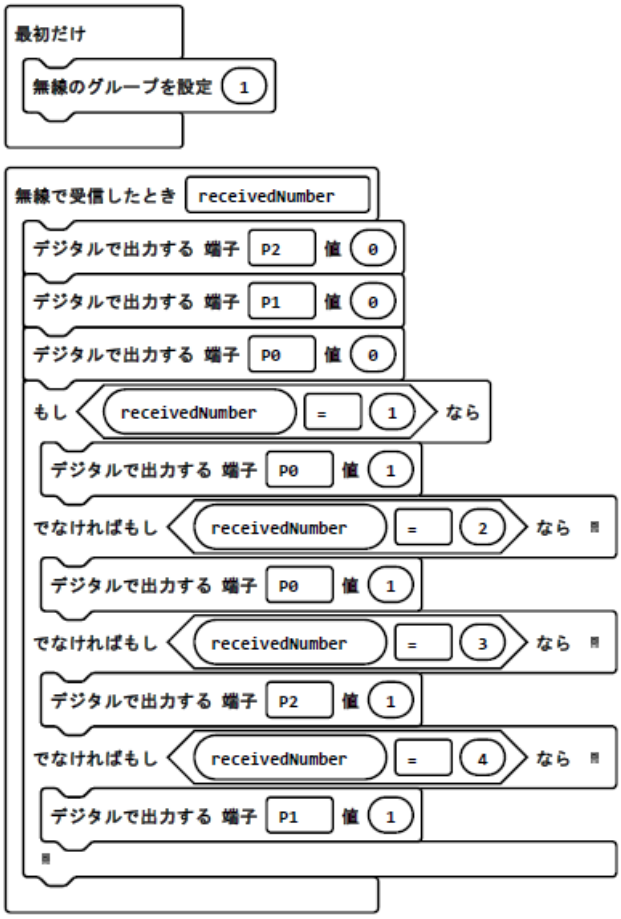
rei4-6


最初だけ
無線のグループを設定 1
無線で受信したとき receivedNumber
デジタルで出力する 端子 P2 値 0
デジタルで出力する 端子 P1 値 0
デジタルで出力する 端子 P0 値 0
もし receivedNumber = 2 なら
デジタルで出力する 端子 P2 値 1
でなければもし receivedNumber = 1 なら
デジタルで出力する 端子 P1 値 1
でなければもし receivedNumber = 0 なら
デジタルで出力する 端子 P0 値 1

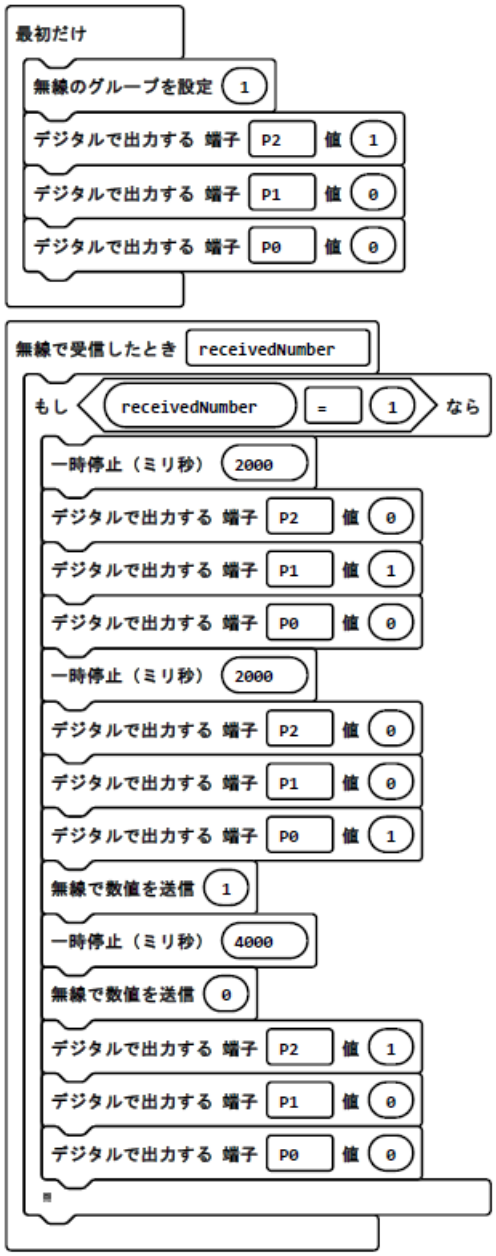
ren4-4-1



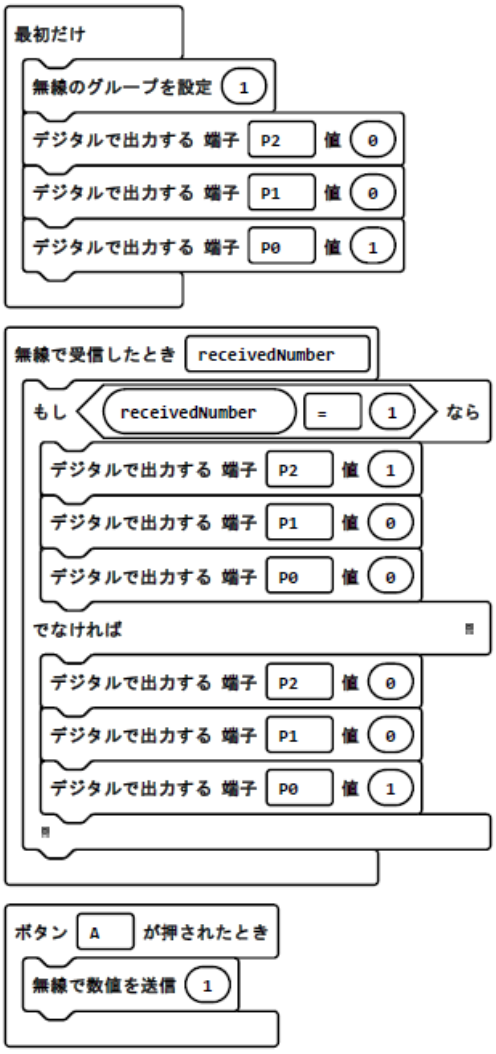
ren4-4-2



rei4-7-1

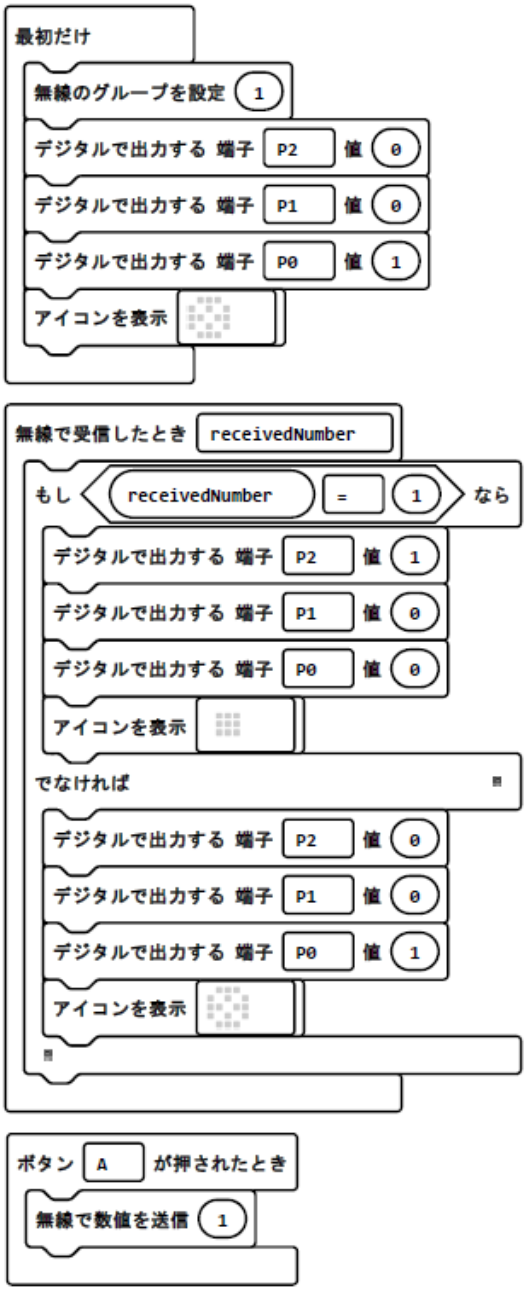


rei4-7-2



演習問題

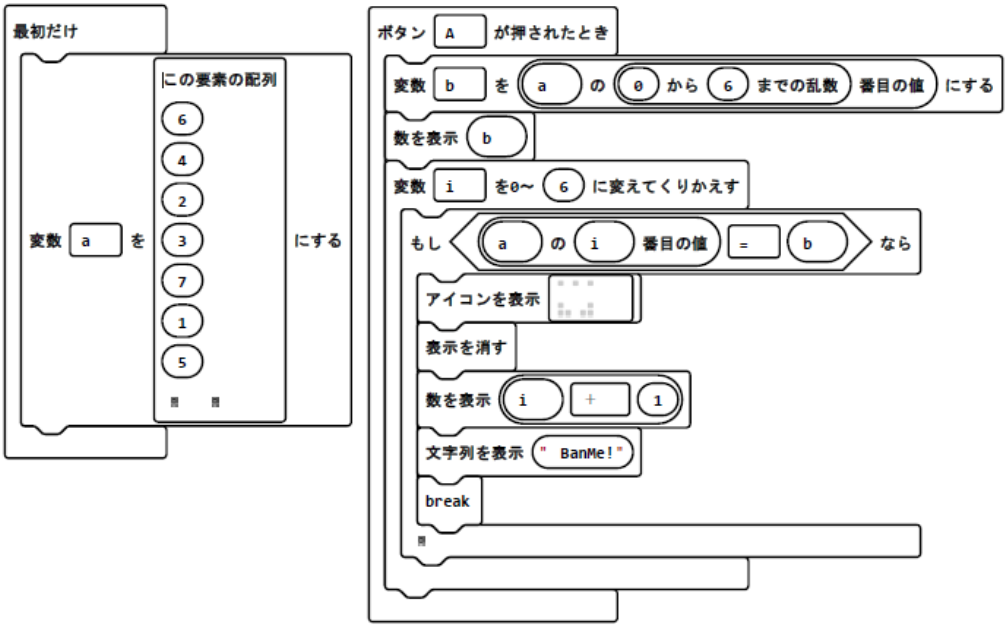
ens4-1


最初だけ
無線のグループを設定 1
デジタルで出力する 端子 P2 値 0
デジタルで出力する 端子 P1 値 0
デジタルで出力する 端子 P0 値 1
アイコンを表示
無線で受信したとき receivedNumber
もし receivedNumber = 1 なら
デジタルで出力する 端子 P2 値 1
デジタルで出力する 端子 P1 値 0
デジタルで出力する 端子 P0 値 0
アイコンを表示
でなければ
デジタルで出力する 端子 P2 値 0
デジタルで出力する 端子 P1 値 0
デジタルで出力する 端子 P0 値 1
アイコンを表示
ボタン A が押されたとき
無線で数値を送信 1

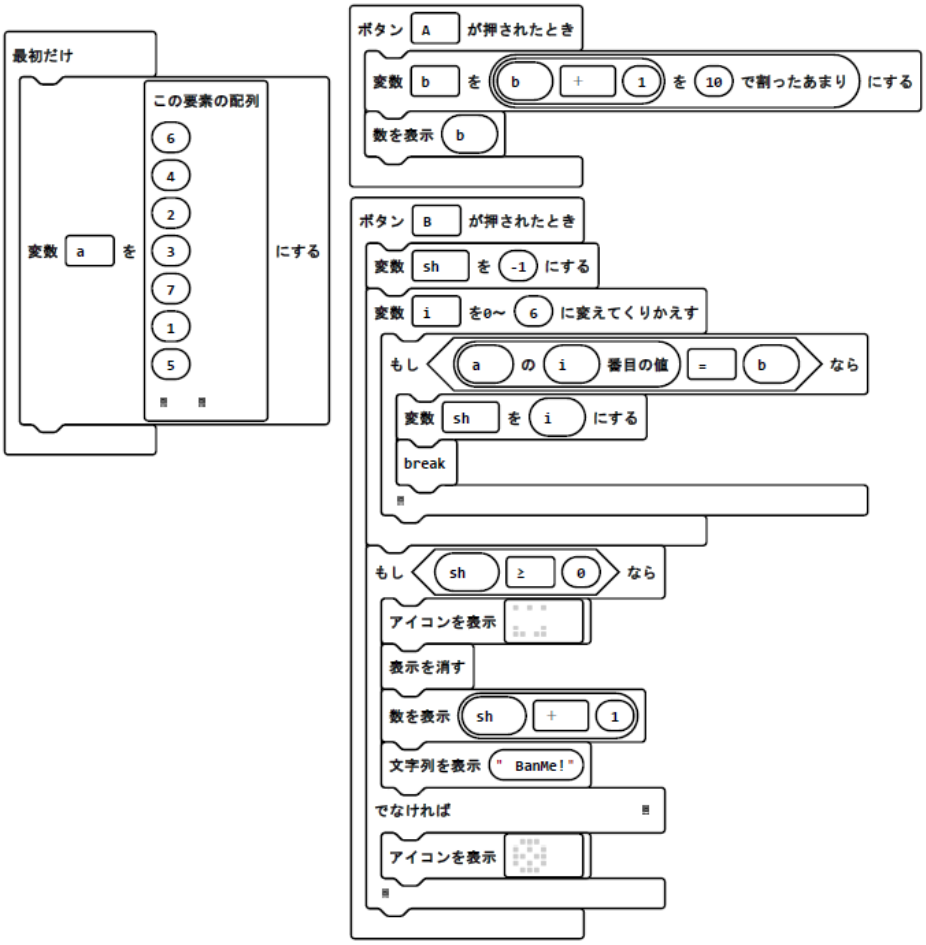
５．アルゴリズムとプログラム

5.1　探索

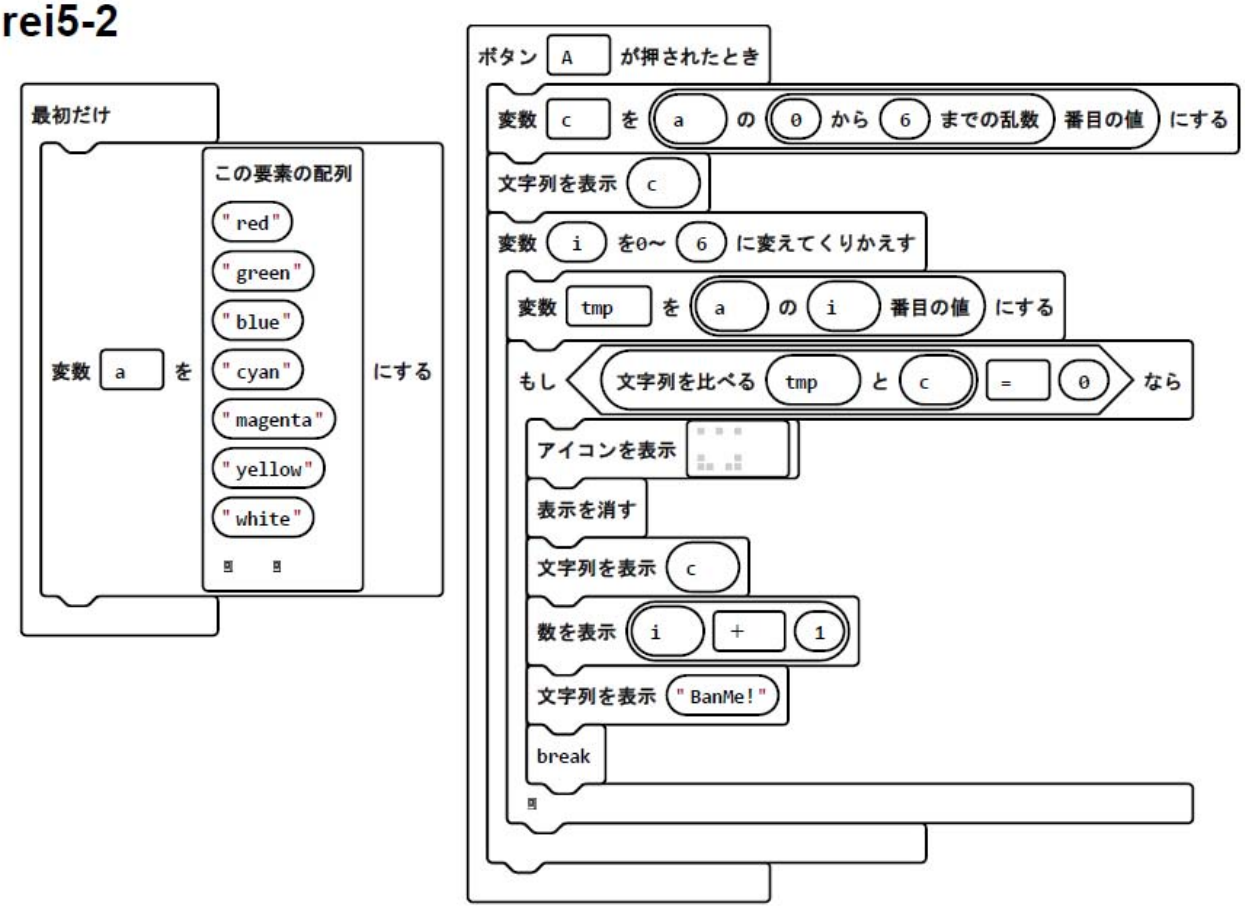
rei5-1



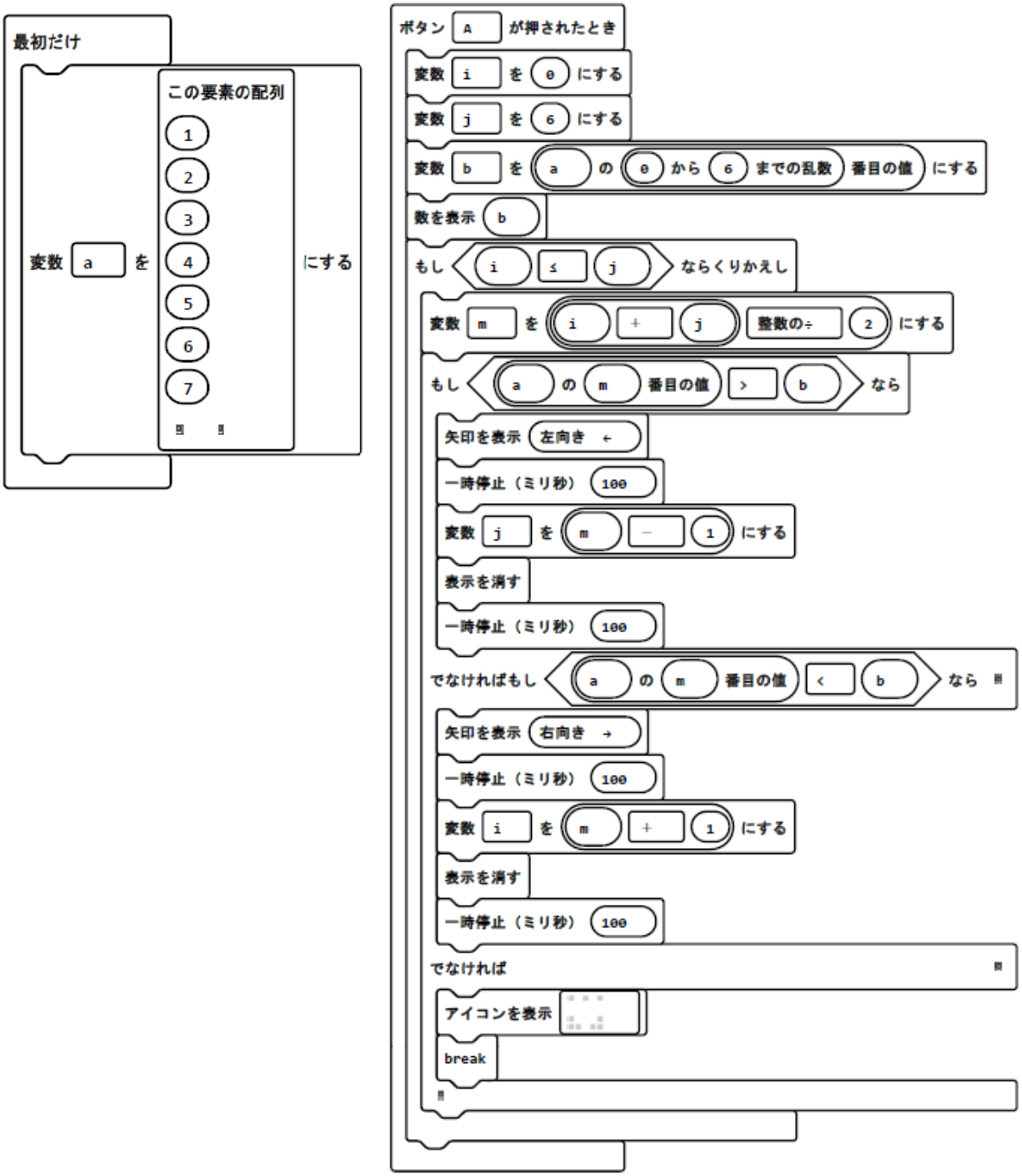
ren5-1



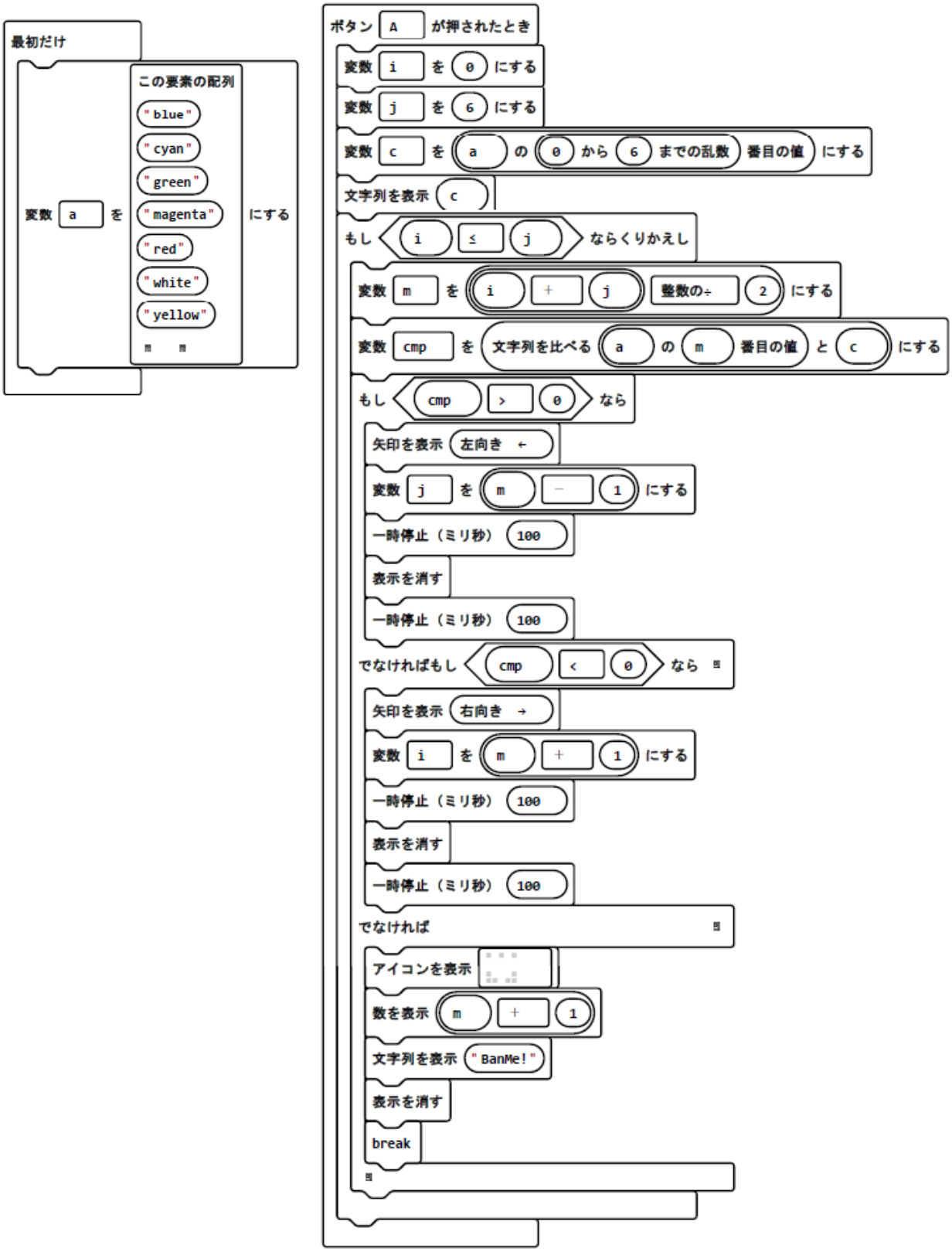
rei5-2


最初だけ
この要素の配列
"red"
"green"
"blue"
"cyan"
"magenta"
"yellow"
"white"
変数 a を にする
ボタン A が押されたとき
変数 c を a の 0 から 6 までの乱数 番目の値 にする
文字列を表示 c
変数 i を0〜 6 に変えてくりかえす
変数 tmp を a の i 番目の値 にする
もし 文字列を比べる tmp と c = 0 なら
アイコンを表示
表示を消す
文字列を表示 c
数を表示 i + 1
文字列を表示 "BanMe!"
break

rei5-3

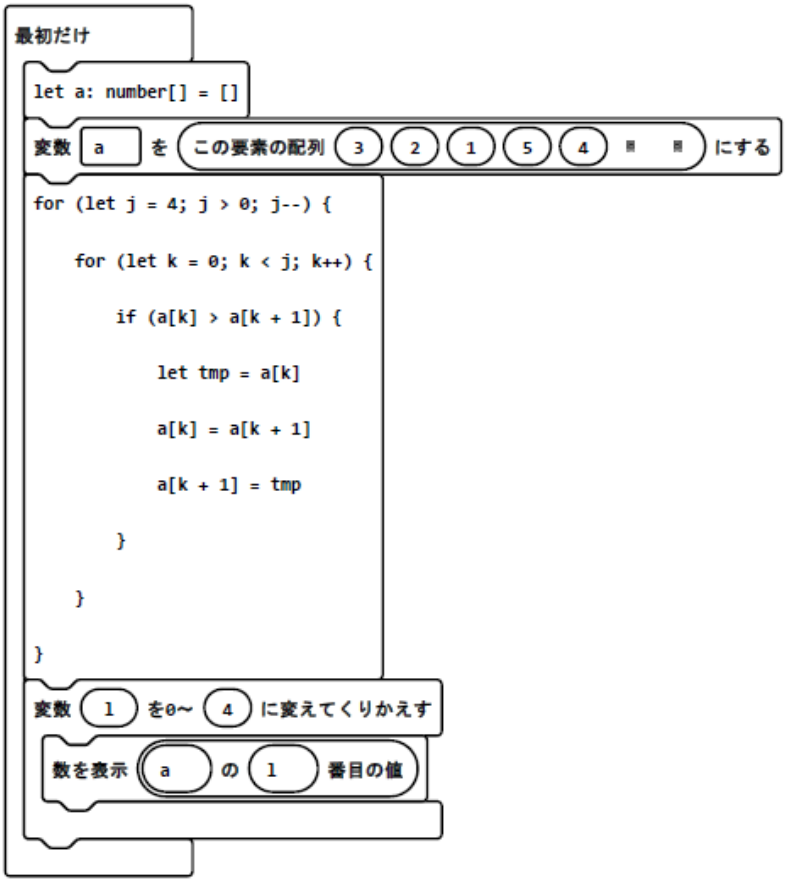

最初だけ
この要素の配列
1
2
3
4
5
6
7
変数 a を にする
ボタン A が押されたとき
変数 i を 0 にする
変数 j を 6 にする
変数 b を a の 0 から 6 までの乱数 番目の値 にする
数を表示 b
もし i ≤ j ならくりかえし
変数 m を i + j 整数の÷ 2 にする
もし a の m 番目の値 > b なら
矢印を表示 左向き ←
一時停止（ミリ秒） 100
変数 j を m − 1 にする
表示を消す
一時停止（ミリ秒） 100
でなければもし a の m 番目の値 < b なら
矢印を表示 右向き →
一時停止（ミリ秒） 100
変数 i を m + 1 にする
表示を消す
一時停止（ミリ秒） 100
でなければ
アイコンを表示
break

rei5-4

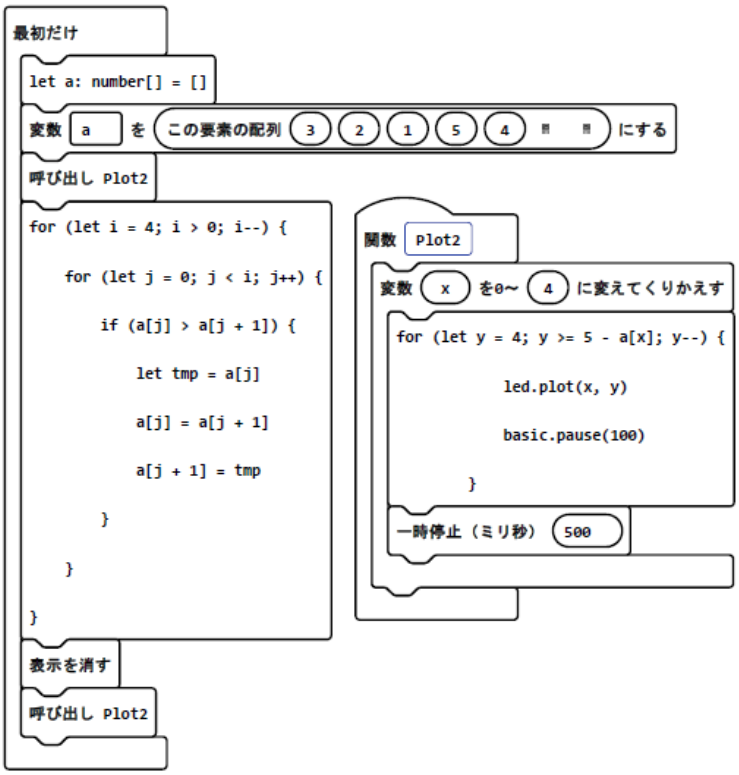

最初だけ
この要素の配列
"blue"
"cyan"
"green"
"magenta"
"red"
"white"
"yellow"
変数 a を にする
ボタン A が押されたとき
変数 i を 0 にする
変数 j を 6 にする
変数 c を a の 0 から 6 までの乱数 番目の値 にする
文字列を表示 c
もし i ≤ j ならくりかえし
変数 m を i + j 整数の÷ 2 にする
変数 cmp を 文字列を比べる a の m 番目の値 と c にする
もし cmp > 0 なら
矢印を表示 左向き ←
変数 j を m − 1 にする
一時停止（ミリ秒） 100
表示を消す
一時停止（ミリ秒） 100
でなければもし cmp < 0 なら
矢印を表示 右向き →
変数 i を m + 1 にする
一時停止（ミリ秒） 100
表示を消す
一時停止（ミリ秒） 100
でなければ
アイコンを表示
数を表示 m + 1
文字列を表示 "BanMe!"
表示を消す
break

5.2　整列

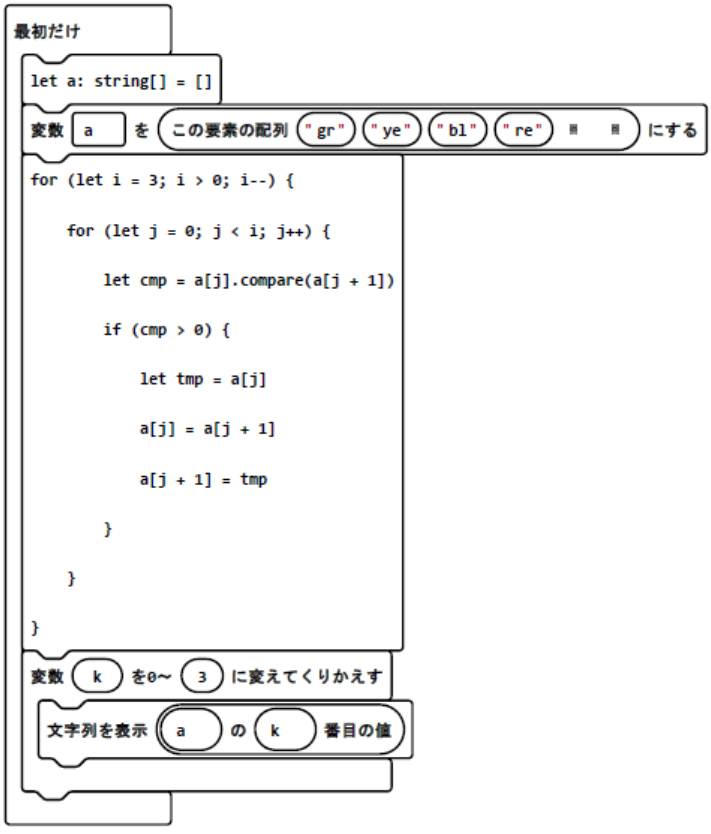
rei5-5


最初だけ
let a: number[] = []
変数 a を この要素の配列 3 2 1 5 4 にする
for (let j = 4; j > 0; j--) {
for (let k = 0; k < j; k++) {
if (a[k] > a[k + 1]) {
let tmp = a[k]
a[k] = a[k + 1]
a[k + 1] = tmp
}
}
}
変数 i を0〜 4 に変えてくりかえす
数を表示 a の i 番目の値

ren5-3

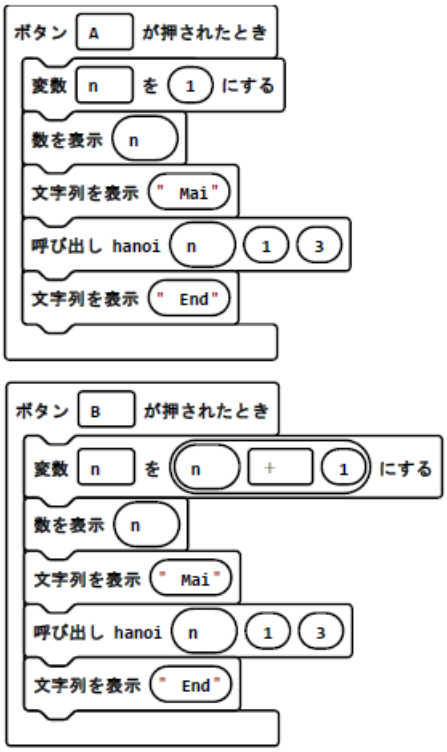

最初だけ
let a: number[] = []
変数 a を この要素の配列 3 2 1 5 4 にする
呼び出し Plot2
for (let i = 4; i > 0; i--) {
for (let j = 0; j < i; j++) {
if (a[j] > a[j + 1]) {
let tmp = a[j]
a[j] = a[j + 1]
a[j + 1] = tmp
}
}
}
表示を消す
呼び出し Plot2
関数 Plot2
変数 x を0～ 4 に変えてくりかえす
for (let y = 4; y >= 5 - a[x]; y--) {
led.plot(x, y)
basic.pause(100)
}
一時停止（ミリ秒） 500

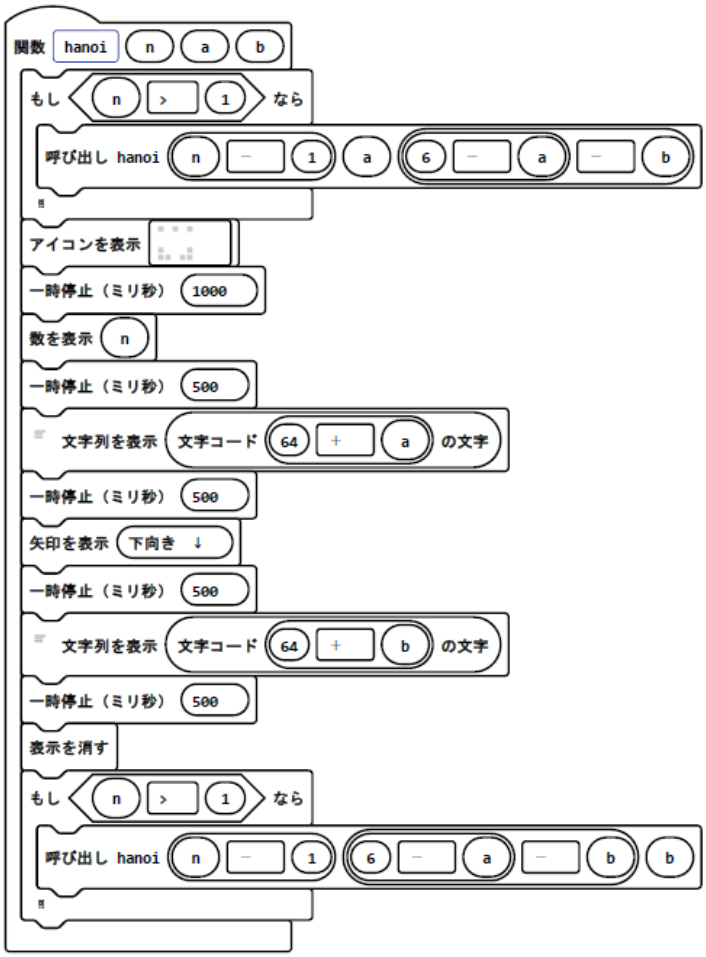
ren5-4


最初だけ
let a: string[] = []
変数 a を この要素の配列 "gr" "ye" "bl" "re" にする
for (let i = 3; i > 0; i--) {
for (let j = 0; j < i; j++) {
let cmp = a[j].compare(a[j + 1])
if (cmp > 0) {
let tmp = a[j]
a[j] = a[j + 1]
a[j + 1] = tmp
}
}
}
変数 k を0～ 3 に変えてくりかえす
文字列を表示 a の k 番目の値

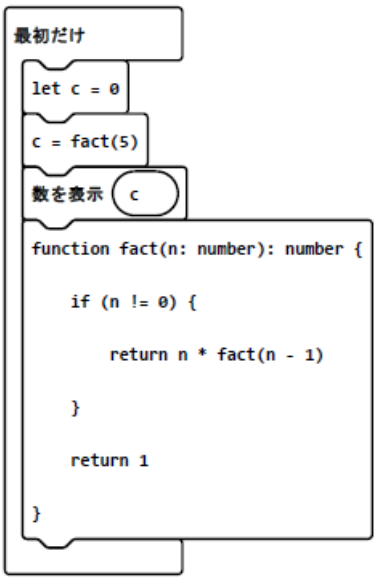
5.3　ハノイの塔

rei5-6

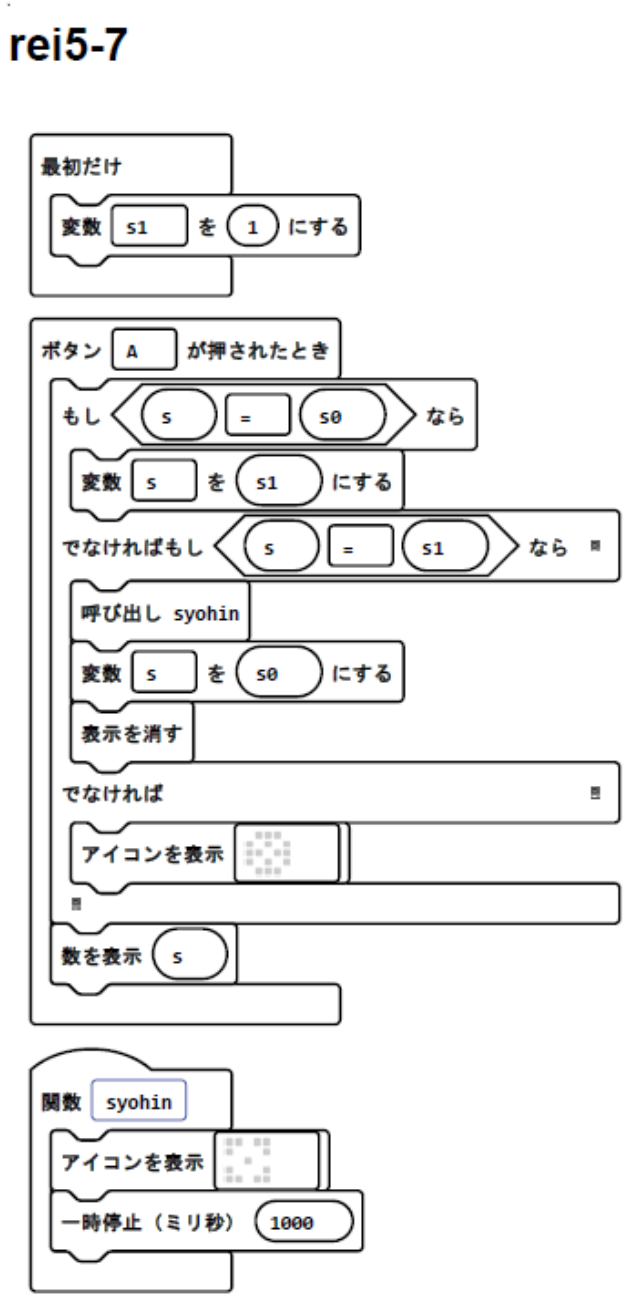




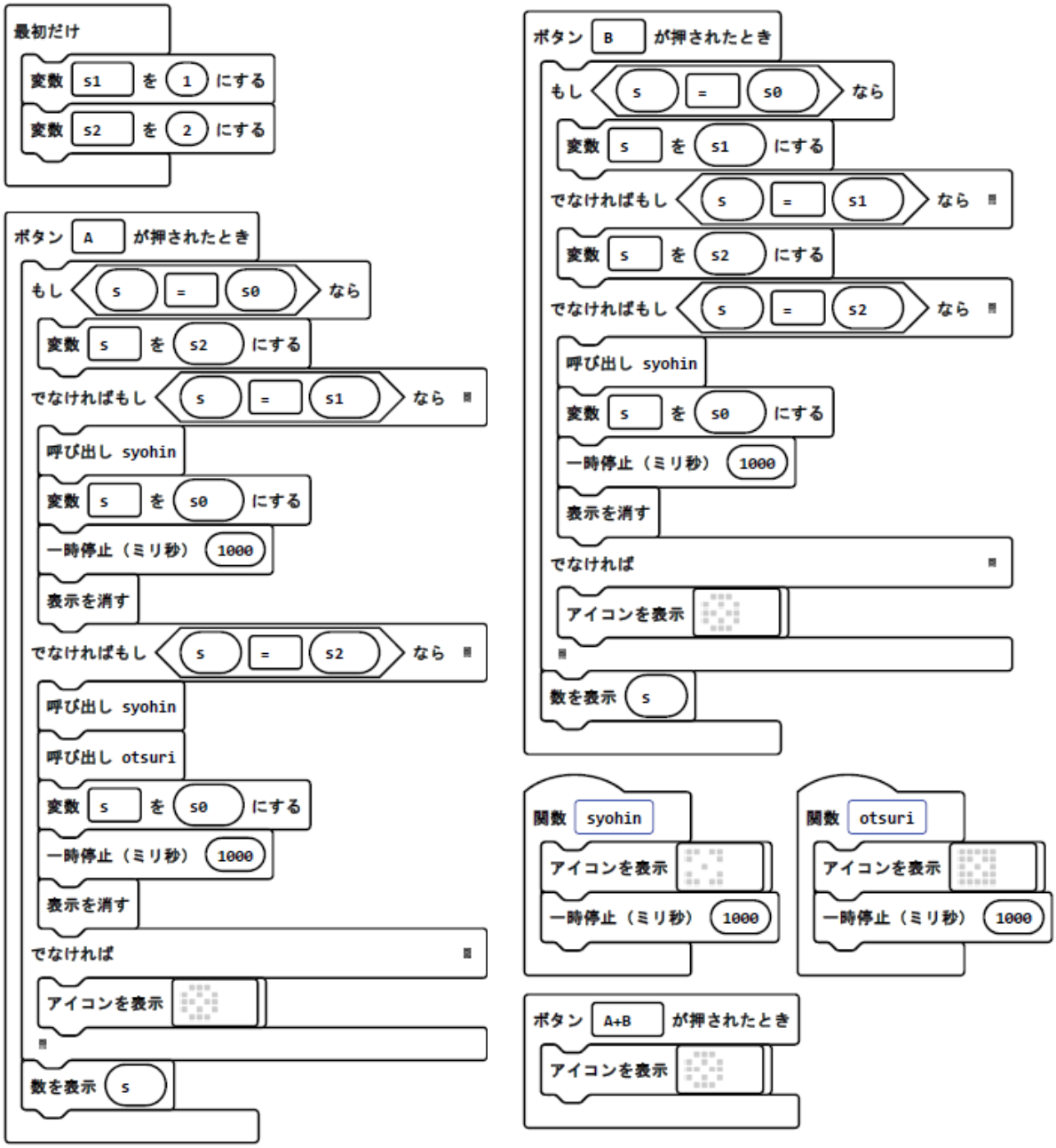
c53-fact



5.4　自動販売機の状態遷移図

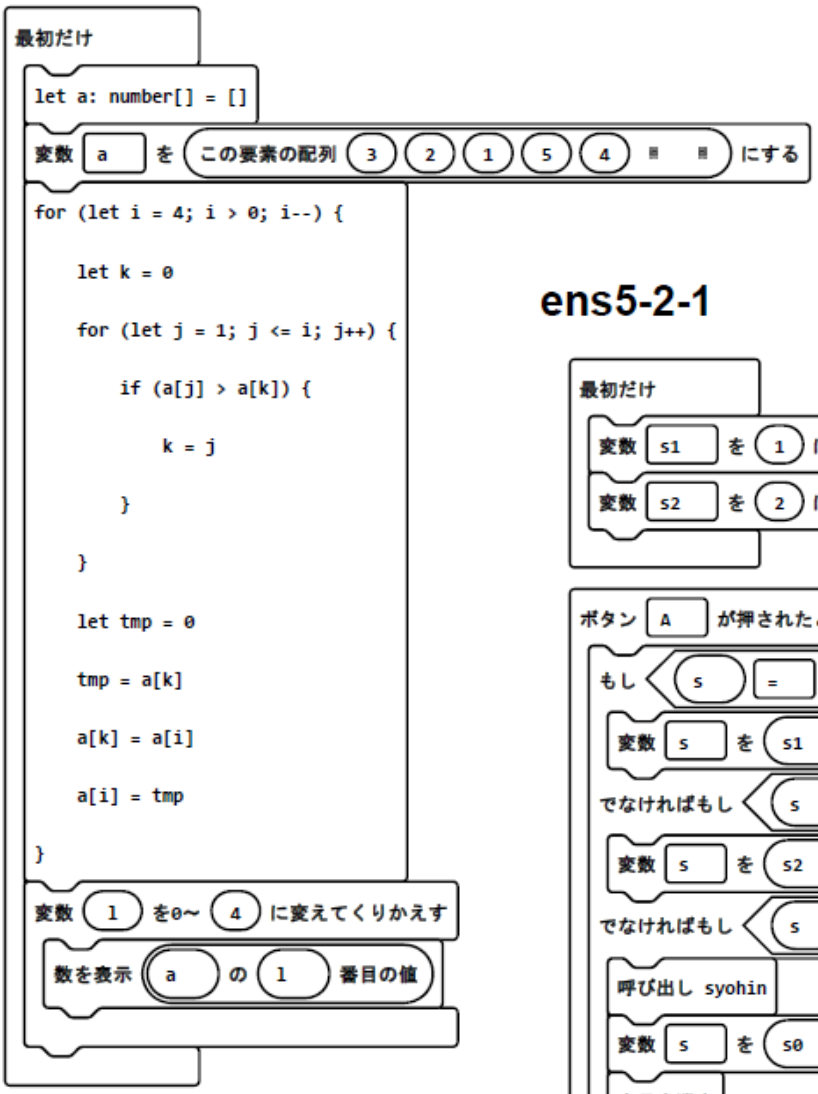


rei5-8

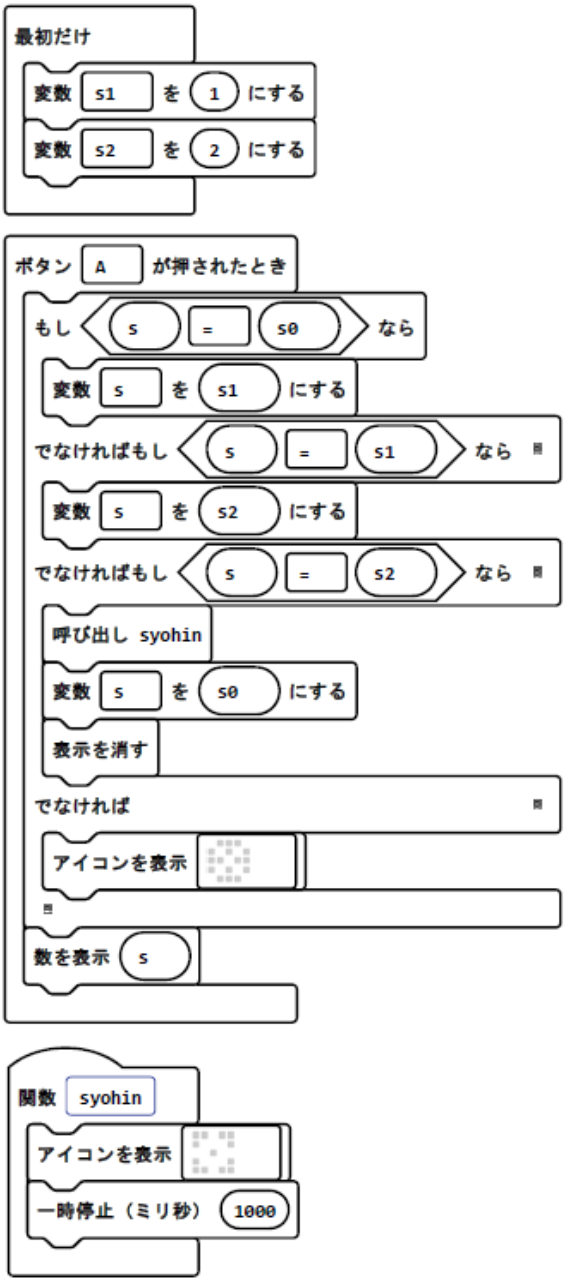

最初だけ
変数 s1 を 1 にする
変数 s2 を 2 にする
ボタン A が押されたとき
もし s = s0 なら
変数 s を s2 にする
でなければもし s = s1 なら
呼び出し syohin
変数 s を s0 にする
一時停止（ミリ秒） 1000
表示を消す
でなければもし s = s2 なら
呼び出し syohin
呼び出し otsuri
変数 s を s0 にする
一時停止（ミリ秒） 1000
表示を消す
でなければ
アイコンを表示
数を表示 s
ボタン B が押されたとき
もし s = s0 なら
変数 s を s1 にする
でなければもし s = s1 なら
変数 s を s2 にする
でなければもし s = s2 なら
呼び出し syohin
変数 s を s0 にする
一時停止（ミリ秒） 1000
表示を消す
でなければ
アイコンを表示
数を表示 s
関数 syohin
アイコンを表示
一時停止（ミリ秒） 1000
関数 otsuri
アイコンを表示
一時停止（ミリ秒） 1000
ボタン A+B が押されたとき
アイコンを表示

演習問題

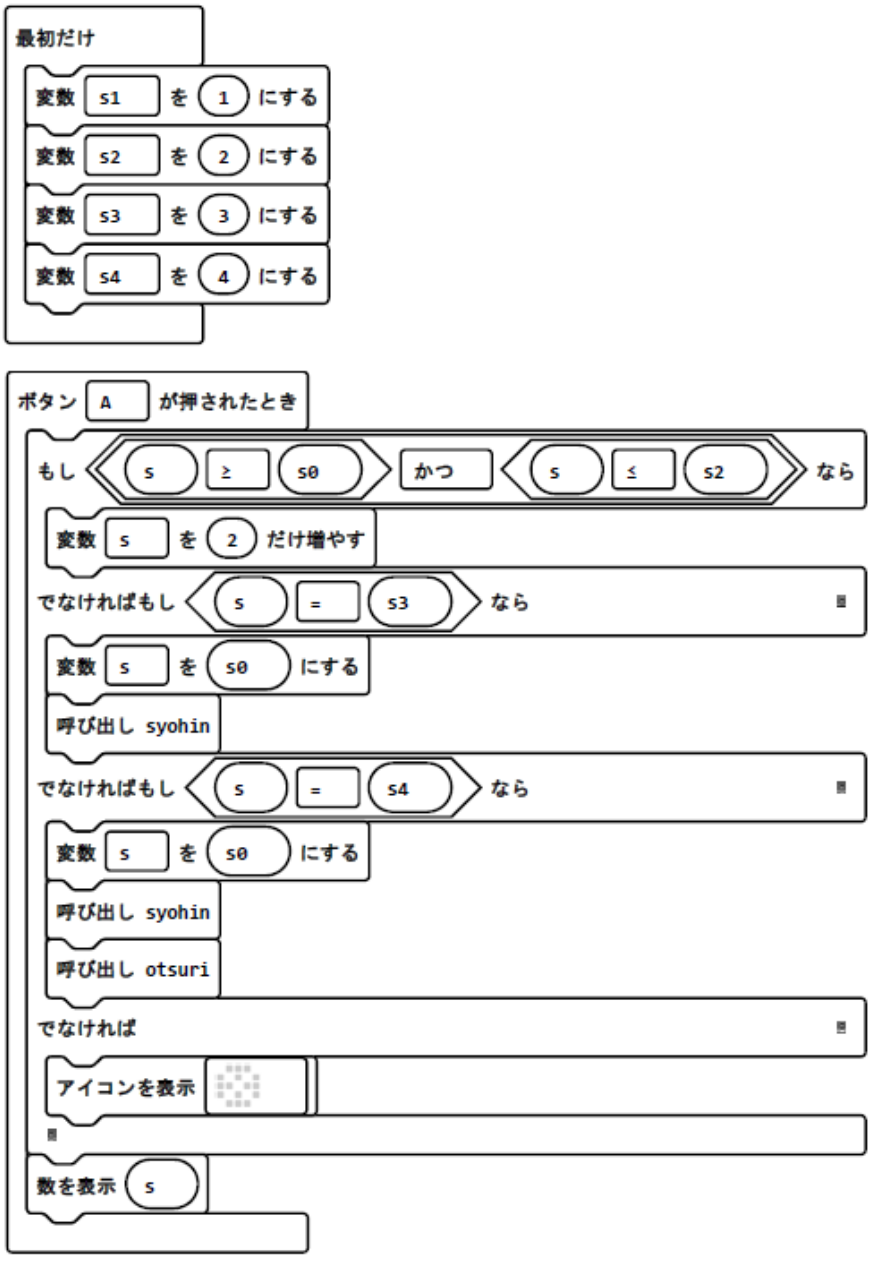
ens5-1

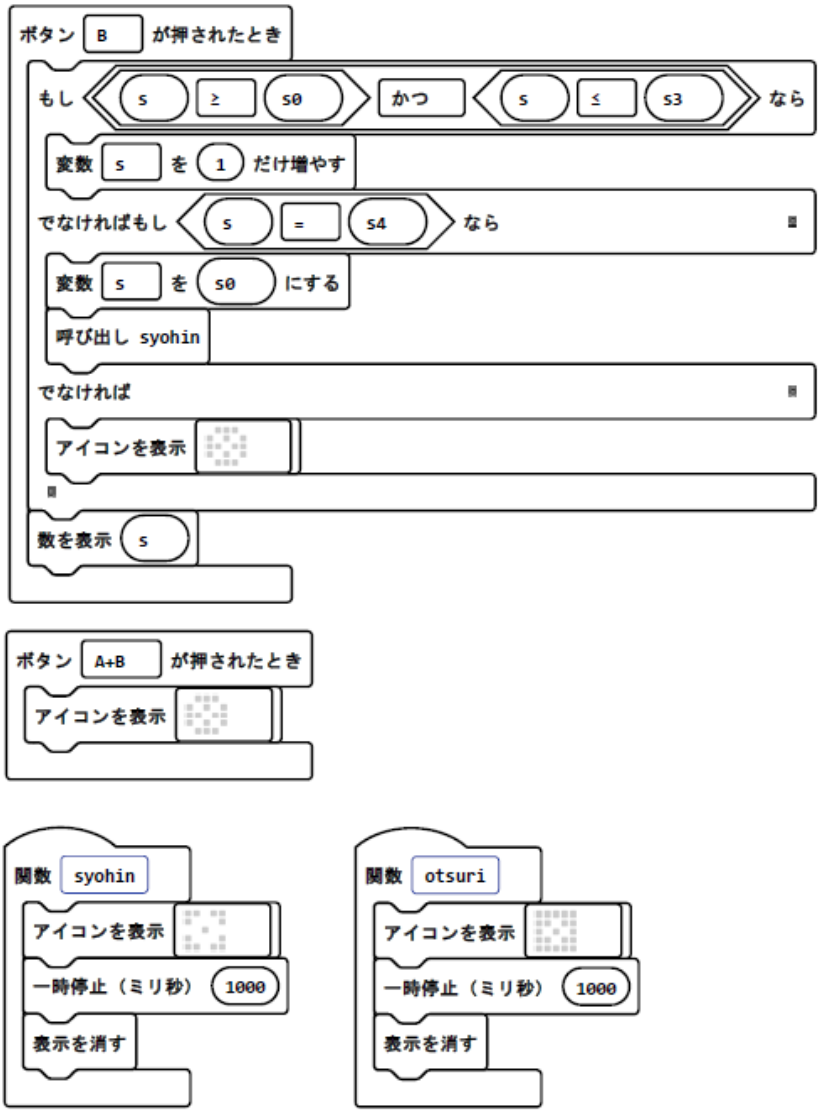


ens5-2-1



ens5-2-2


最初だけ
変数 s1 を 1 にする
変数 s2 を 2 にする
変数 s3 を 3 にする
変数 s4 を 4 にする
ボタン A が押されたとき
もし s ≥ s0 かつ s ≤ s2 なら
変数 s を 2 だけ増やす
でなければもし s = s3 なら
変数 s を s0 にする
呼び出し syohin
でなければもし s = s4 なら
変数 s を s0 にする
呼び出し syohin
呼び出し otsuri
でなければ
アイコンを表示
数を表示 s


ボタン B が押されたとき
もし s ≥ s0 かつ s ≤ s3 なら
変数 s を 1 だけ増やす
でなければもし s = s4 なら
変数 s を s0 にする
呼び出し syohin
でなければ
アイコンを表示
数を表示 s

ボタン A+B が押されたとき
アイコンを表示

関数 syohin
アイコンを表示
一時停止（ミリ秒） 1000
表示を消す

関数 otsuri
アイコンを表示
一時停止（ミリ秒） 1000
表示を消す