

米MITメディアラボ開発 ビジュアル・プログラミング環境 「Scratch」Web版version3 高度なプログラミング事例

令和7年8月24日
茂出木敏雄

ホームページ <http://www.bekkoame.ne.jp/~modegi/>

掲載プログラムの全データは上記サイトよりダウンロードできます。
Scratch_example.zip内の¥Scratch¥adv¥フォルダ

6. 配列の使用例(1):シーザー暗号(1)

入力された文字列(本例では英大文字に限定)の各文字コード(ASCIIコード、本例では1~26)に暗号キー(正負の整数値)を加算し、コード表の順にずらした文字に変換する基本的な暗号化手法。暗号キーの正負を逆にして暗号化すれば復号化できる。

暗号化と復号化で使用する暗号キーが同一(正負反転するが)なので、共通鍵暗号方式とも呼ばれる。暗号キーをどのように相手に知らせるかが問題である(鍵配送問題)。

"Scratch3_encrypt.sb3"

The image shows the Scratch3_encrypt.sb3 project interface. The code is organized into several sections:

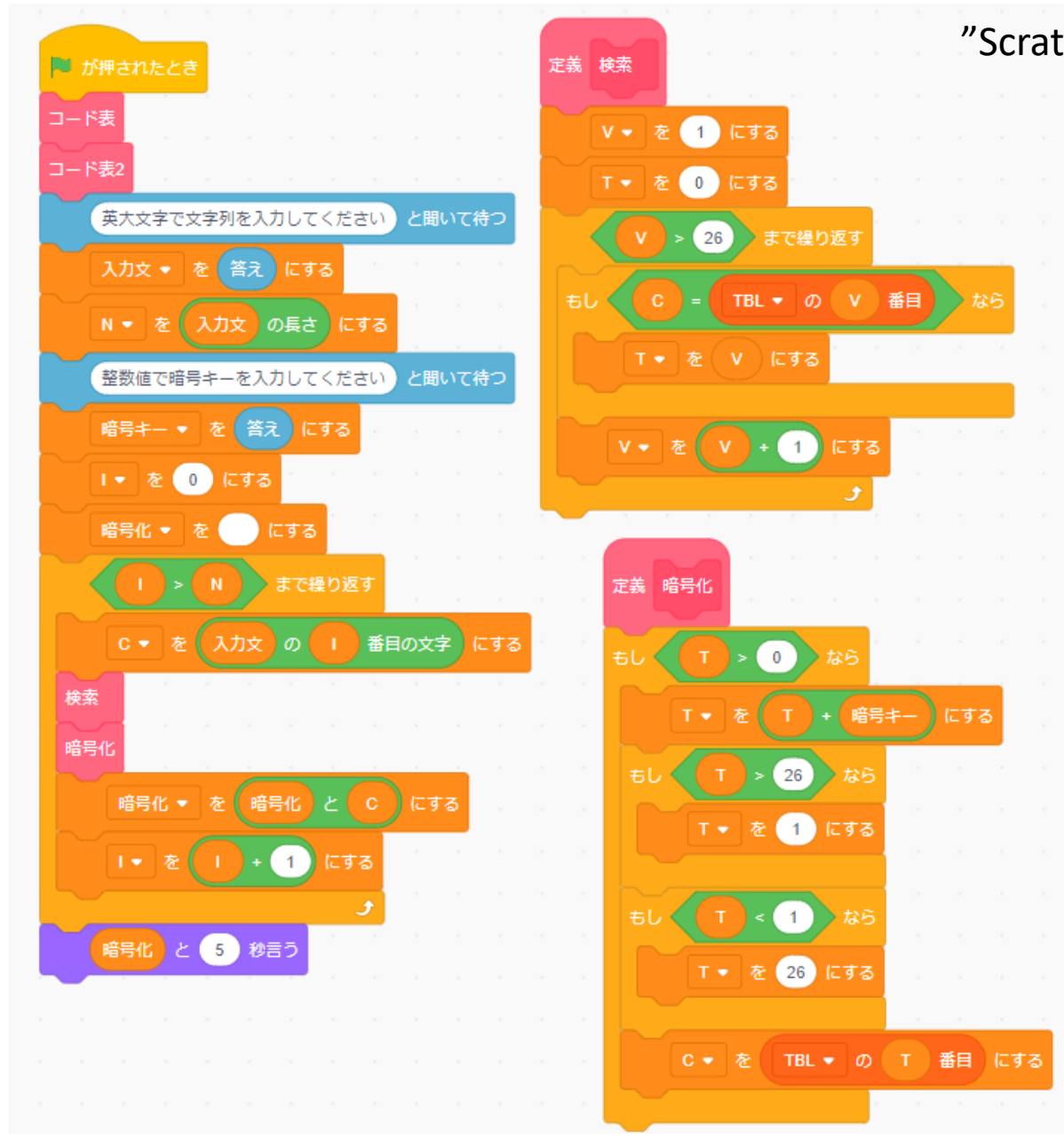
- が押されたとき (When clicked):** Initializes variables for code table (コード表), code table 2 (コード表2), input text (入力文), number of characters (N), encryption key (暗号キー), encryption/decryption mode (暗号化), and a loop counter (I). It prompts the user to enter the input text and the encryption key.
- 定義 検索 (Define Search):** A function to find the index of a character in the code table.
- 定義 暗号化 (Define Encrypt):** A function to encrypt a character by adding the encryption key to its code.
- 定義 コード表 (Define Code Table):** A function to initialize the code table with letters A-Z.
- 定義 コード表2 (Define Code Table 2):** A function to initialize the code table 2 with letters A-Z.

The right side of the interface shows the project settings:

- 入力文 (Input Text):** IBM
- 暗号キー (Encryption Key):** -1
- 暗号化 (Encryption/Decryption):** HAL
- TBL (Code Table):** A list of 26 slots, currently showing A through F.
- 長さ (Length):** 26
- スプライト (Sprite):** スプライト1 (Sprite1)
- 表示する (Show):** On
- 向き (Direction):** 90

The Scratch cat character is visible on the right side of the stage.

6. 配列の使用例(1):シーザー暗号(2):プログラム部拡大



6. 配列の使用例(1):シーザー暗号(3):コード定義部拡大

"Scratch3_encrypt.sb3"

The image shows two Scratch code blocks for defining tables. The first block, titled 'コード表', contains a sequence of blocks to initialize a table: 'TBL のすべてを削除する' followed by 'A' through 'M' being added to 'TBL'. The second block, titled 'コード表2', contains a sequence of blocks to initialize a second table: 'N' through 'Z' being added to 'TBL'.

コード表

- TBL のすべてを削除する
- A を TBL に追加する
- B を TBL に追加する
- C を TBL に追加する
- D を TBL に追加する
- E を TBL に追加する
- F を TBL に追加する
- G を TBL に追加する
- H を TBL に追加する
- I を TBL に追加する
- J を TBL に追加する
- K を TBL に追加する
- L を TBL に追加する
- M を TBL に追加する

コード表2

- N を TBL に追加する
- O を TBL に追加する
- P を TBL に追加する
- Q を TBL に追加する
- R を TBL に追加する
- S を TBL に追加する
- T を TBL に追加する
- U を TBL に追加する
- V を TBL に追加する
- W を TBL に追加する
- X を TBL に追加する
- Y を TBL に追加する
- Z を TBL に追加する

6. 配列の使用例(1):シーザー暗号(4):実行部拡大

入力する文字列を"IBM COMPUTER"、暗号キーを「-1」とすると、暗号化結果は、"HAL BNLOTSDQ"になる。復号化する場合、入力する文字列を" HAL BNLOTSDQ"、暗号キーを「+1」とすると、暗号化結果は、"IBM COMPUTER"になる。

"Scratch3_encrypt.sb3"

[文字列の入力]

[暗号化結果]

N 入力文

暗号キー 暗号化

TBL

1	A
2	B
3	C
4	D
5	E
6	F
7	G
+	長さ 26 =

C

T

I

V

英大文字で文字列を入力してください

IBM COMPUTER

N 入力文

暗号キー 暗号化

TBL

1	A
2	B
3	C
4	D
5	E
6	F
7	G
+	長さ 26 =

C

T

I

V

HAL BNLOTSDQ

Scratchでは「リスト」と呼ばれる1次元の配列を扱うことができる。26要素の配列(「リスト」)を定義している。

6. 配列の使用例(2): 公開鍵暗号(1)

入力された文字列(本例では英大文字に限定)の各文字コード(ASCIIコード、本例では1~32)に暗号キー(正の整数値)で累乗し、33で割った余りのコードの文字に変換する高度な暗号化手法。暗号化と復号化で異なるキー(3または7)を使用するのが特徴で、共通鍵暗号方式の鍵配送問題を解決できる。2種の鍵を生成し、一方のキーを公開鍵として相手に知らせ(第3者に漏洩しても問題ない)、もう一方のキーは秘密鍵として自分が保管する。 "Scratch3 rsaenc.sb3"

The screenshot shows the Scratch 3.0 interface with a custom block script for a Caesar cipher. The script is as follows:

```

when green flag clicked
  wait for user
  loop
    for each letter in table
      shift letter by 3
  end loop

```

The right panel shows the 'Custom Blocks' section with a table of 26 letters and a 'Shift' value of 3.

Letter	Shift
A	3
B	3
C	3
D	3
E	3
F	3
G	3
H	3
I	3
J	3
K	3
L	3
M	3
N	3
O	3
P	3
Q	3
R	3
S	3
T	3
U	3
V	3
W	3
X	3
Y	3
Z	3

6. 配列の使用例(2): 公開鍵暗号(2): プログラム部拡大

Scratch3_rsaenc.sb3

```
graph TD
    subgraph Main_Script [Main Script]
        Start([が押されたとき]) --> Prompt1[英大文字で文字列を入力して下さい]
        Prompt1 --> Input1[入力文]
        Input1 --> SetN[N = 入力文の長さ]
        Prompt1 --> Prompt2[暗号キー(3または7)を入力して下さい]
        Prompt2 --> Input2[暗号キー]
        Input2 --> SetI[I = 0]
        SetI --> SetEnc[暗号化 = ]
        SetEnc --> Loop1{I > N}
        Loop1 -- まで繰り返す --> SetC[C = 入力文の I 番目の文字]
        SetC --> Lookup[検索]
        Lookup --> Encrypt[暗号化]
        Encrypt --> SetEnc[暗号化 = 暗号化 と C]
        SetEnc --> SetI[I = I + 1]
        SetI --> Loop1
        SetEnc --> Speak[暗号化 と 5 秒言う]
    end

    subgraph Lookup_Function [定義 検索]
        SetV[V = 1]
        SetT[T = 0]
        Loop2{V > 32}
        Loop2 -- まで繰り返す --> SetC2[C = TBLのV番目]
        SetC2 --> SetT[T = V]
        SetT --> SetV[V = V + 1]
        SetV --> Loop2
    end

    subgraph Encrypt_Function [定義 暗号化]
        SetT3[T = 0]
        SetS[S = T]
        SetV3[V = 1]
        Loop3{V = 暗号キー}
        Loop3 -- まで繰り返す --> SetS3[S = S * Tを33で割った余り]
        SetS3 --> SetV3[V = V + 1]
        SetV3 --> Loop3
        SetC3[C = TBLのS番目]
    end
```

The script is divided into three main sections: the main execution flow, a 'lookup' function, and an 'encrypt' function.

Main Execution Flow:

- When the green flag is clicked, prompt the user for a message (English uppercase letters).
- Store the message in a variable 'Input Text'.
- Calculate the length of the message and store it in 'N'.
- Prompt the user for an encryption key (3 or 7) and store it in 'Key'.
- Initialize 'I' to 0 and 'Encrypted' to an empty string.
- Loop while 'I' is less than 'N':
 - Get the character at index 'I' from the input text.
 - Call the 'lookup' function to find its index in the table.
 - Call the 'encrypt' function to calculate the encrypted value.
 - Append the encrypted value and the original character to the 'Encrypted' string.
 - Increment 'I' by 1.
- Speak the final encrypted string for 5 seconds.

lookup Function (定義 検索):

- Initialize 'V' to 1 and 'T' to 0.
- Loop while 'V' is less than or equal to 32:
 - Set 'C' to the value at index 'V' in the 'TBL' table.
 - Set 'T' to 'V'.
 - Increment 'V' by 1.

encrypt Function (定義 暗号化):

- Initialize 'T' to 0 and 'S' to 'T'.
- Initialize 'V' to 1.
- Loop while 'V' is less than or equal to the 'Key':
 - Set 'S' to the remainder of 'S * T' divided by 33.
 - Increment 'V' by 1.
- Set 'C' to the value at index 'S' in the 'TBL' table.

6. 配列の使用例(2):公開鍵暗号(3):コード定義部拡大

The image shows three columns of Scratch code blocks, each with a '定義' (Definition) tab and a 'コード表' (Code Table) tab. The blocks are orange and contain text in Japanese.

Column 1: コード表

- TBL ▼ のすべてを削除する
- A を TBL ▼ に追加する
- B を TBL ▼ に追加する
- C を TBL ▼ に追加する
- D を TBL ▼ に追加する
- E を TBL ▼ に追加する
- F を TBL ▼ に追加する
- G を TBL ▼ に追加する
- H を TBL ▼ に追加する
- I を TBL ▼ に追加する
- J を TBL ▼ に追加する
- K を TBL ▼ に追加する
- L を TBL ▼ に追加する
- M を TBL ▼ に追加する

Column 2: コード表2

- N を TBL ▼ に追加する
- O を TBL ▼ に追加する
- P を TBL ▼ に追加する
- Q を TBL ▼ に追加する
- R を TBL ▼ に追加する
- S を TBL ▼ に追加する
- T を TBL ▼ に追加する
- U を TBL ▼ に追加する
- V を TBL ▼ に追加する
- W を TBL ▼ に追加する
- X を TBL ▼ に追加する
- Y を TBL ▼ に追加する
- Z を TBL ▼ に追加する

Column 3: コード表3

- + を TBL ▼ に追加する
- を TBL ▼ に追加する
- % を TBL ▼ に追加する
- & を TBL ▼ に追加する
- # を TBL ▼ に追加する
- = を TBL ▼ に追加する

"Scratch3_rsaenc.sb3"

6. 配列の使用例(2): 公開鍵暗号(4): 実行部拡大

入力する文字列を"IBM COMPUTER"、暗号キーを「3」とすると、暗号化結果は、"CHS +ISDUNZX"になる。復号化する場合、入力する文字列を"CHS +ISDUNZX"、暗号キーを「7」とすると、暗号化結果は、"IBM COMPUTER"になる。逆に暗号キー「7」で暗号化した場合は、暗号キー「3」でないと復号化できない。 "Scratch3_encrypt.sb3"

[文字列の入力]

The interface shows a Scratch-style form for inputting a string and an encryption key. At the top, there are two rows of input fields: "N" with the value "3" and "入力文" (Input Text) with the value "IBM". Below these, there are two more rows: "暗号キー" (Encryption Key) with the value "3" and "暗号化" (Encryption) with the value "CHS". On the left, there is a table labeled "TBL" with 7 rows and 2 columns. The first column contains letters A through G, and the second column contains numbers 13, 4, 3, 19, and an empty cell. A speech bubble from the Scratch cat character says "英大文字で文字列を入力して下さい" (Please input the string in uppercase letters). At the bottom, there is a text input field containing "IBM COMPUTER" and a blue checkmark button.

TBL	
1	A
2	B
3	C
4	D
5	E
6	F
7	G

長さ 32 =

[暗号化結果]

The interface shows the result of encryption. At the top, there are two rows of input fields: "N" with the value "12" and "入力文" (Input Text) with the value "IBM COMPUTER". Below these, there are two more rows: "暗号キー" (Encryption Key) with the value "3" and "暗号化" (Encryption) with the value "CHS +ISDUNZX". On the left, there is a table labeled "TBL" with 7 rows and 2 columns. The first column contains letters A through G, and the second column contains numbers 18, 13, 3, 24, and an empty cell. A speech bubble from the Scratch cat character says "CHS +ISDUNZX". At the bottom, there is a text input field containing "CHS +ISDUNZX" and a blue checkmark button.

TBL	
1	A
2	B
3	C
4	D
5	E
6	F
7	G

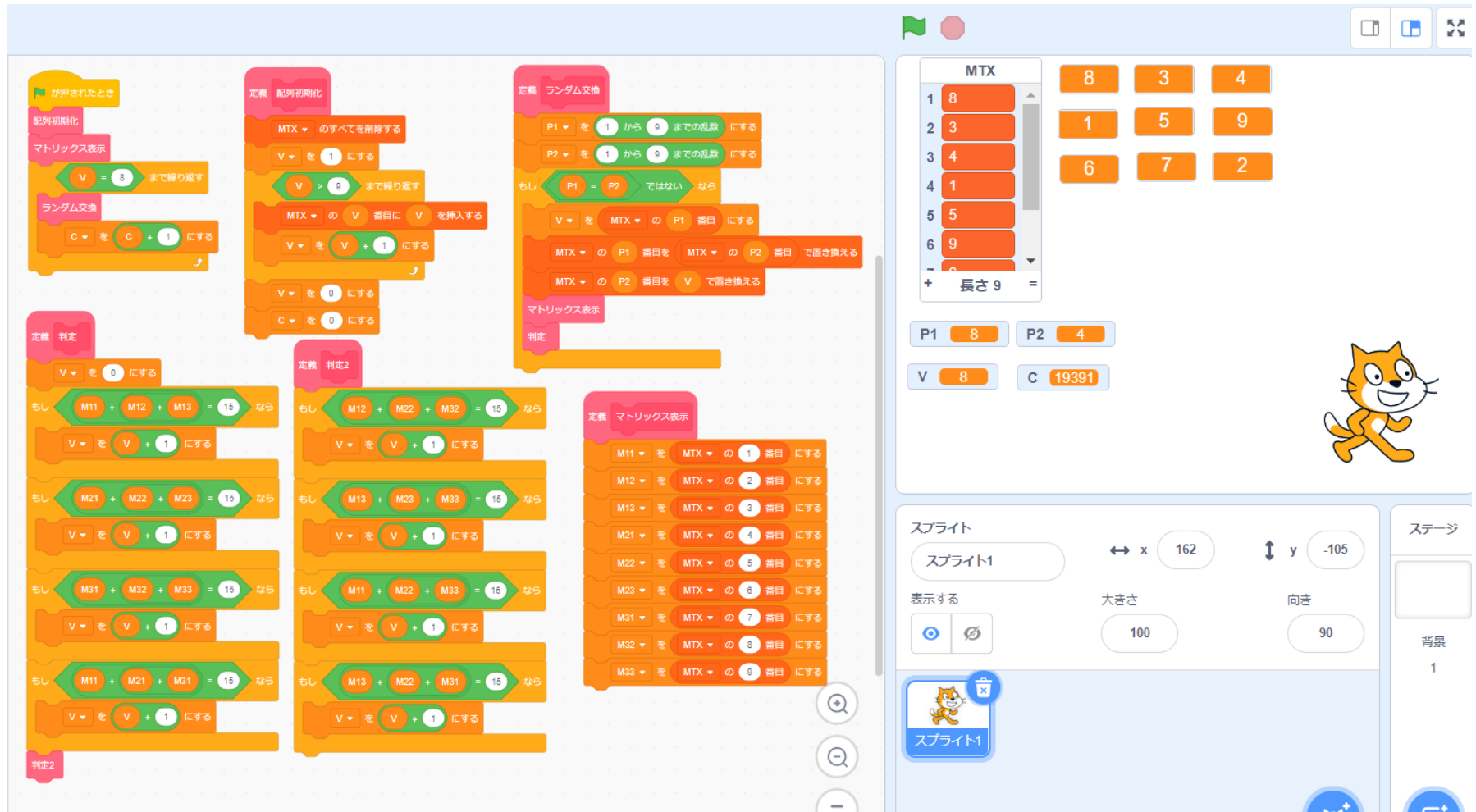
長さ 32 =

Scratchでは「リスト」と呼ばれる1次元の配列を扱うことができる。32要素の配列(「リスト」)を定義している。

6. 配列の使用例(3): 3×3魔方陣(スクランブル法)(1)

魔方陣とは、 $N \times N$ ($N > 2$) のマスに $1 \sim N \times N$ までの数字を、各行の N 個の総和、各列の N 個の総和、対角2方向の各々 N 個の総和がいずれも同一値 ($N=3$ の場合、15) になるように配置したもの。本例題は、初期状態として、 $1 \sim N \times N$ までの数字を順に配置し、ランダムに2つのマスを交換しながら、魔方陣を満足するパターンを探索します。

"Scratch3_magicr.sb3"



The screenshot displays a Scratch3 project titled "Scratch3_magicr.sb3" with a 3x3 magic square implementation. The project includes several custom blocks and a main script.

Custom Blocks:

 - 初期状態 (Initial State):** A block that initializes the magic square matrix (MTX) and sets the current value (V) to 1. It includes a loop to fill the matrix with numbers 1 to 9.
 - ランダム交換 (Random Swap):** A block that randomly selects two positions (P1, P2) and swaps the values in the MTX matrix.
 - 判定 (Check):** A block that checks if the current configuration is a valid magic square by verifying the sum of each row, column, and diagonal is 15. If valid, it increments the counter (C) and sets V to 0.
 - 判定2 (Check 2):** A block that checks if the current configuration is a valid magic square by verifying the sum of each row, column, and diagonal is 15. If valid, it increments the counter (C) and sets V to 0.
 - マトリックス表示 (Matrix Display):** A block that displays the current state of the MTX matrix.

Main Script:

- When the green flag is clicked, the "初期状態" block is executed.
- A loop is entered where the "ランダム交換" block is executed repeatedly.
- The "判定" block is used to check for a valid magic square.
- If a valid magic square is found, the "判定2" block is executed, and the counter (C) is incremented.
- The "マトリックス表示" block is used to display the current state of the MTX matrix.

MTX Matrix:

	1	2	3
1	8	3	4
2	1	5	9
3	6	7	2

Variables:

- P1: 8
- P2: 4
- V: 8
- C: 19391

Sprite: A cat sprite named "Sprite1" is positioned at x=162, y=-105, with a size of 100 and a rotation of 90 degrees.

6. 配列の使用例(3): 3×3 魔方陣(スクランブル法)(2): プログラム部拡大

The image displays a collection of Scratch code blocks for a 3x3 magic square program. The blocks are organized into several functional groups:

- が押されたとき (When clicked):** Contains a sequence of blocks: 配列初期化 (Initialize array), マトリックス表示 (Display matrix), a loop from V=8 to 9, and ランダム交換 (Random swap).
- 定義 配列初期化 (Function: Initialize array):** Includes blocks for deleting all MTX, setting V to 1, a loop from V=1 to 9, inserting V into MTX at index V, incrementing V, and finally setting V to 0 and C to 0.
- 定義 ランダム交換 (Function: Random swap):** Includes blocks for generating random numbers P1 and P2, a conditional check if P1 is not equal to P2, and logic to swap elements in MTX at indices P1 and P2.
- 定義 判定 (Function: Judgment):** Includes a block to set V to 0, followed by four conditional checks for row sums (M11+M12+M13, M21+M22+M23, M31+M32+M33, M11+M21+M31) and column sums (M12+M22+M32, M13+M23+M33, M11+M22+M33, M13+M22+M31). Each check increments V by 1 if the sum is 15.
- 定義 判定2 (Function: Judgment2):** This block is currently empty.
- 定義 マトリックス表示 (Function: Matrix display):** A list of blocks that map matrix elements M11 through M33 to specific indices in the MTX array (e.g., M11 to MTX index 1, M33 to MTX index 9).

At the bottom right, the text "Scratch3_magicr.sb3" is visible, along with standard Scratch interface icons (search, zoom, and a menu button).

6. 配列の使用例(3) : 3 × 3 魔方陣 (スクランブル法) (3) : 実行部拡大

Scratchでは「リスト」と呼ばれる1次元の配列を扱うことができる。9要素の配列(「リスト」)を定義している。

The image shows a Scratch interface with a 3x3 magic square and a list named MTX. The magic square contains the numbers 8, 3, 4 in the first row; 1, 5, 9 in the second row; and 6, 7, 2 in the third row. The list MTX contains the numbers 8, 3, 4, 1, 5, 9, 6, 7, 2. A red arrow points from the text above to the MTX list. A blue arrow points from the MTX list to the '大きな表示' (Large Display) option in a menu. The menu also includes '普通に表示' (Normal Display) and 'スライダー' (Slider). Below the magic square, there are four variable monitors: P1 (8), P2 (4), V (8), and C (19391). The Scratch cat is visible in the bottom right corner.

変数は以下のように3通りの表示モードを使用できる。魔方陣の部分は「大きな表示」を使用

普通に表示
大きな表示
スライダー

6. 配列の使用例(4) : 3×3 魔方陣 (インドの奇数魔方陣の高速化解法) (1)

$N \times N$ (N は奇数) のマトリックスの中央列のトップを1として、右上方向に2~ $N \times N$ までの数字を順に配置する。枠からはみ出したら、はみ出した方向と反対側に配置し、配置しようとするマスが配置済の場合、右上でなく1行下にずらした位置に配置する操作を繰り返す。

”Scratch3_magich.sb3”

The screenshot shows a Scratch3 script for generating a 3x3 magic square. The script is organized into several defined functions:

- マトリックス表示 (Matrix Display):** Iterates through the 9 elements of the MTX array, assigning values to M11 through M33.
- 配列初期化 (Array Initialization):** Clears the MTX array, sets P to 1, and enters a loop that fills the array with values 1 through 9.
- 配置確認 (Configuration Check):** Checks if the current value P is greater than 0. If so, it updates X and Y coordinates and increments P.
- マトリックス表示 (Matrix Display):** Increments V by 1.

On the right, a 3x3 grid displays the resulting magic square:

8	1	6
3	5	7
4	9	2

Below the grid, variables are shown: X=1, XP=0, V=10, YP=0.

6. 配列の使用例(4) : 3×3 魔方陣(インドの奇数魔方陣の高速化解法)(2)

The image displays a Scratch script titled "Scratch3_magich.sb3" designed to generate a 3×3 magic square using the Siyuan method. The script is organized into several defined functions:

- が押されたとき (When clicked):** Initiates the process by calling the "配列初期化" (Initialize array) function.
- 配列初期化 (Initialize array):** Resets the magic square array (MTX) and sets the starting position (P) to 1. It then enters a loop where it inserts the number 0 into the array at position P and increments P until it reaches 9.
- マトリックス表示 (Matrix display):** A function that iterates through the 9 elements of the MTX array, assigning them to variables M11 through M33 for display.
- 定義 マトリックス表示 (Define Matrix display):** The implementation of the display function, showing the values of M11 through M33.
- 定義 配列初期化 (Define Initialize array):** The implementation of the initialization function, including the loop to fill the array with 0s.
- 定義 配列確認 (Define Array check):** A function that checks if the current position P is greater than 0. If so, it updates the position (X, Y) and the value (V) according to the Siyuan method's rules.

The script uses a loop to repeatedly call the "配列確認" function until the magic square is complete. The final output is a 3×3 magic square where each row, column, and diagonal sums to 15.

6. 配列の使用例(4): 3×3 魔方陣(インドの奇数魔方陣の高速化解法)(3)

Scratchでは「リスト」と呼ばれる1次元の配列を扱うことができる。9要素の配列（「リスト」）を定義している。

変数は以下のように3通りの表示モードを使用できる。
魔方陣の部分は「大きな表示」を使用

普通
大きな表示
スライダー

6. 配列の使用例(5): 3×3魔方陣(しらみつぶし法)(1)

(1)配列IIIに初期値を000 000 000として、最下位の桁を1ずつ増加させながら、9桁の数字を最大012 345 678まで順に生成する。各桁には可変の上限を012 345 678と設定し、上限を超えたら、上位桁に繰り上げる。

例)配列II: 000 345 679 →配列II: 001 000 000

(2)配列IIの値をもとに、配列MTXの値を最下位の桁から順に、配列TBLで決定済の数字を管理しながら、下位桁の数字と重複しないように決定する。数字が重複する場合は、その値より大きい重複しない数字を探索し、8を超えたら、0から探索し直す。配列MTXでは、各桁の数字に1を加えて、1～9の値をもたせる。

例)配列II: 001 000 000 → MTX: 978 654 321

配列II: 001 035 451 → MTX: 834 159 672 (魔法陣の条件適合)

"Scratch3_magic3.sb3"

6. 配列の使用例(5): 3×3魔方陣(しらみつぶし法)(2)

Scratch3 magic3.sb3

定義 ブロック「判定」「判定2」「マトリックス表示」は、3×3魔方陣(スクランブル法)と同じ

6. 配列の使用例(5): 3×3魔方陣(しらみつぶし法)(3) プログラム部拡大1

The image displays three sections of Scratch code blocks for a 3x3 magic square program. The first section, titled 'が押されたとき' (When clicked), includes '配列初期化' (Initialize arrays), 'マトリックス表示' (Matrix display), 'MTX設定' (MTX setting), and a loop 'V = 8'まで繰り返す (Repeat until V=8). The second section, titled '定義 配列初期化' (Definition: Initialize arrays), shows the initialization of arrays II, MTX, and TBL, and a loop 'V > 9'まで繰り返す (Repeat until V>9). The third section, titled '定義 MTX更新' (Definition: Update MTX), shows the logic for updating the MTX array based on the current value of V. The fourth section, titled '定義 マトリックス表示' (Definition: Matrix display), shows the logic for displaying the matrix elements M11 through M33.

が押されたとき

配列初期化

マトリックス表示

V = 8 まで繰り返す

MTX設定

CNT を CNT + 1 にする

マトリックス表示

判定

もし V < 8 なら

MTX更新

定義 配列初期化

II のすべてを削除する

MTX のすべてを削除する

TBL のすべてを削除する

V を 1 にする

V > 9 まで繰り返す

II の V 番目に 0 を挿入する

MTX の V 番目に V を挿入する

TBL の V 番目に 0 を挿入する

V を V + 1 にする

CNT を 0 にする

定義 MTX更新

II の 1 番目を II の 1 番目 + 1 で置き換える

I を 1 にする

I > 8 まで繰り返す

もし II の I 番目 > 9 - I なら

II の I 番目を 0 で置き換える

II の I + 1 番目を II の I + 1 番目 + 1 で置き換える

I を I + 1 にする

定義 マトリックス表示

M11 を MTX の 1 番目 にする

M12 を MTX の 2 番目 にする

M13 を MTX の 3 番目 にする

M21 を MTX の 4 番目 にする

M22 を MTX の 5 番目 にする

M23 を MTX の 6 番目 にする

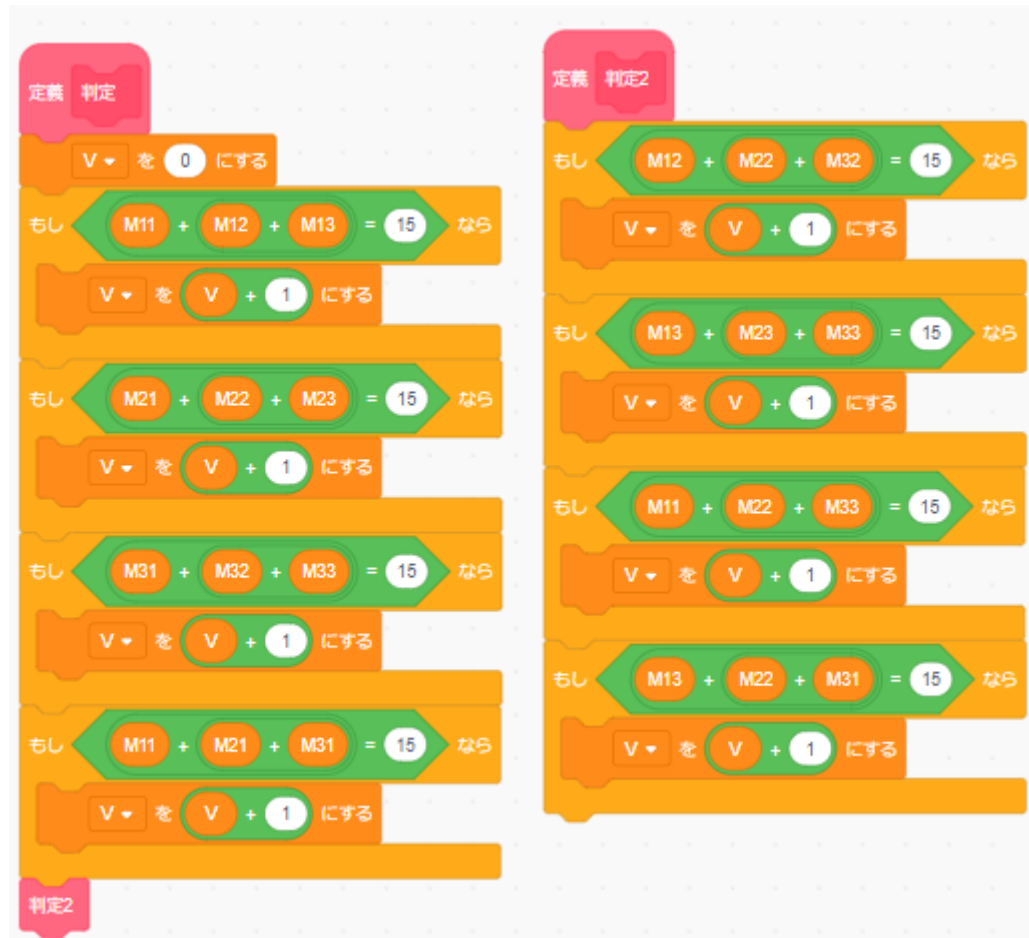
M31 を MTX の 7 番目 にする

M32 を MTX の 8 番目 にする

M33 を MTX の 9 番目 にする

"Scratch3_magic3.sb3"

6. 配列の使用例(5): 3×3魔方陣(しらみつぶし法)(4) プログラム部拡大2



"Scratch3_magic3.sb3"

6. 配列の使用例(5): 3×3魔方陣(しらみつぶし法)(5) 実行部拡大

The image displays the Scratch3 magic3.sb3 interface, which is used for solving a 3x3 magic square using a brute-force method. The interface includes several components:

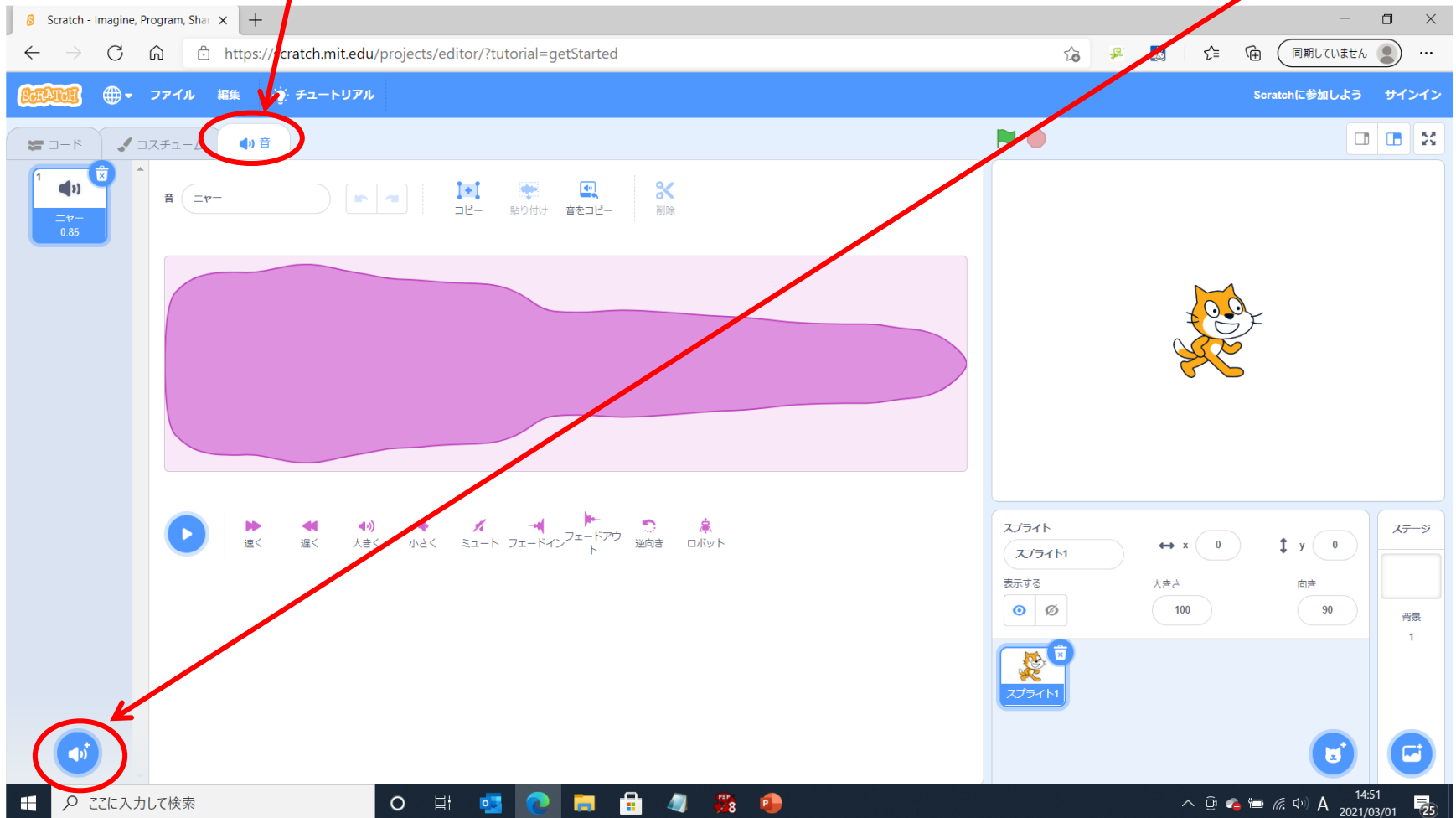
- MTX (Magic Square Matrix):** A 3x3 grid of orange boxes containing the numbers 2, 7, 6 in the first row; 9, 5, 1 in the second row; and 4, 3, 8 in the third row.
- II (Intermediate Array):** A vertical array of orange boxes containing the numbers 1, 5, 4, 5, 3, 0, 1.
- TBL (Table Array):** A vertical array of orange boxes containing the number 1 repeated seven times.
- Variables:** A set of variables at the bottom, including V (8), C (1), K (9), I (0), J (10), and CNT (72407).
- Buttons:** A set of buttons labeled V, C, K, I, J, and CNT, each with a corresponding value.
- Scratch Cat:** A small Scratch Cat character in the bottom right corner.

The interface also features a scroll bar for the MTX array and a scroll bar for the II and TBL arrays. The text "長さ 9" (Length 9) is visible below the MTX and II arrays, indicating the size of the arrays.

"Scratch3_magic3.sb3"

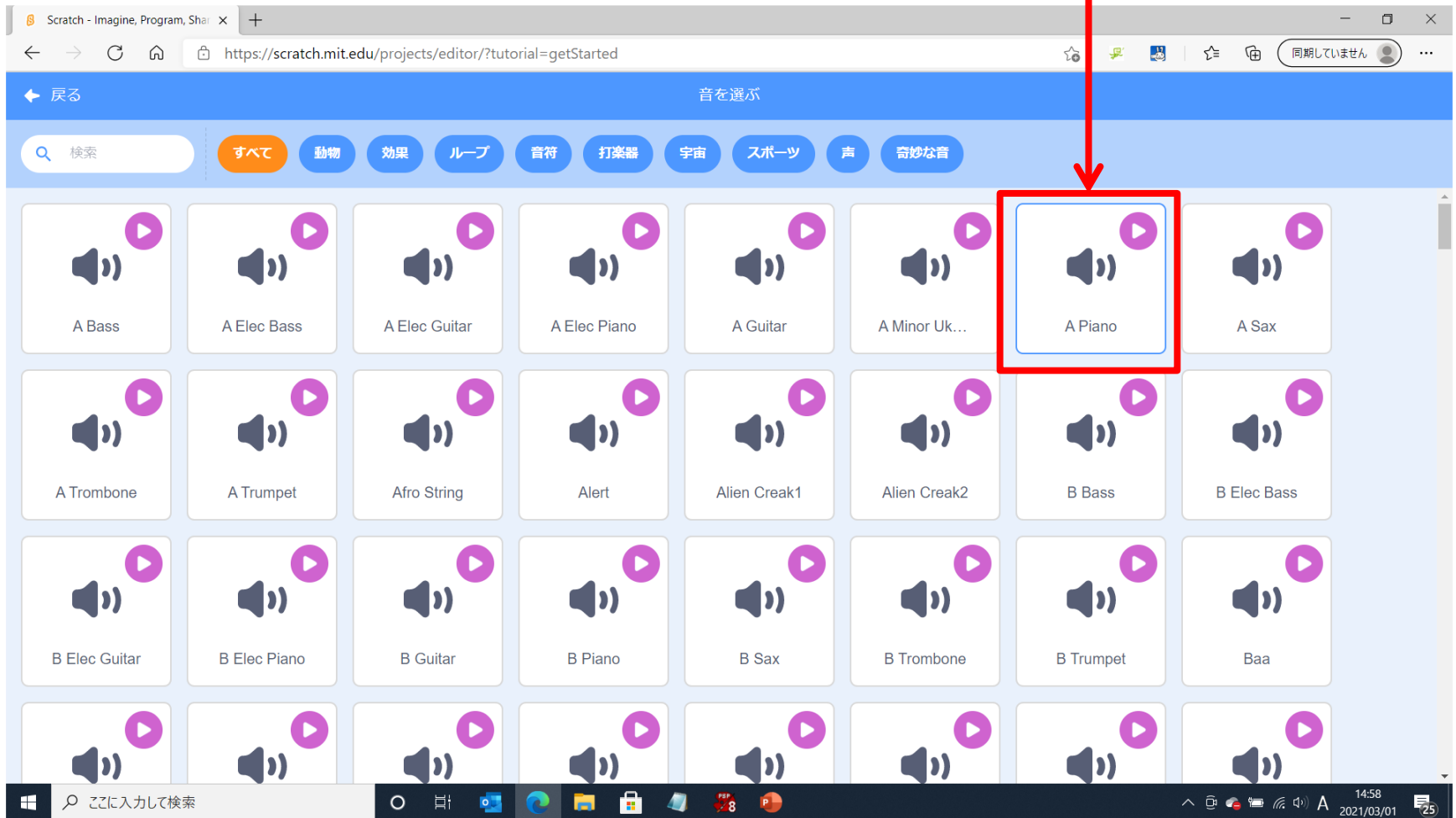
7. 音の使用例: 音の合成・うなり音(1): サウンド素材の追加方法(1)

Scratchにはデフォルトの「ニャー」音だけでなく、種々のサウンド素材を使用できます。以下のように「音」タブを選択して波形が表示された後に、左下の音素材追加のアイコンをクリックします。



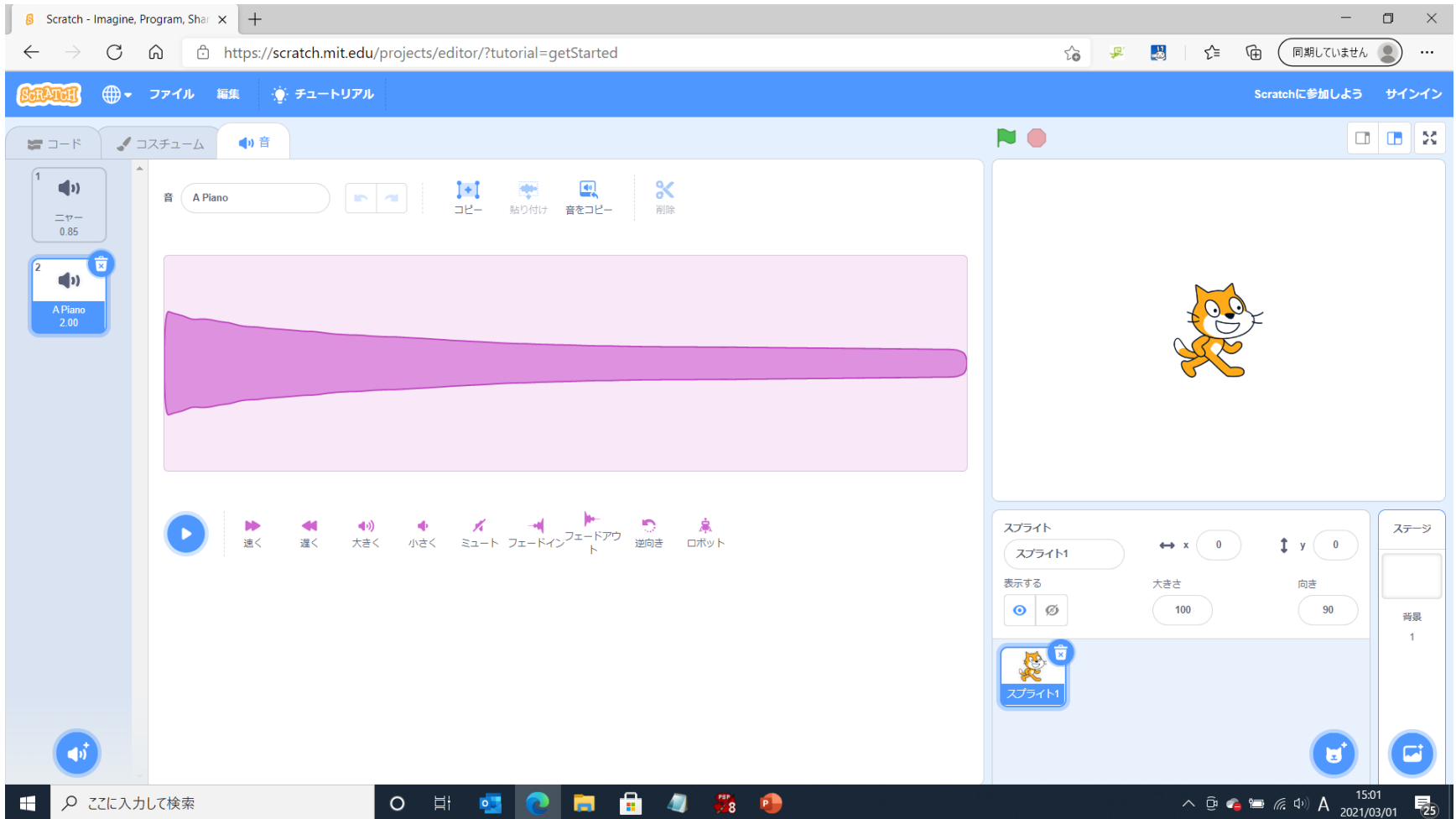
7. 音の使用例: 音の合成・うなり音(2): サウンド素材の追加方法(2)

Scratchにはデフォルトの「ニャー」音だけでなく、種々のサウンド素材を使用できます。以下のように「音を選ぶ」が表示されたら、例えば、「A Piano」のアイコンをクリックします。



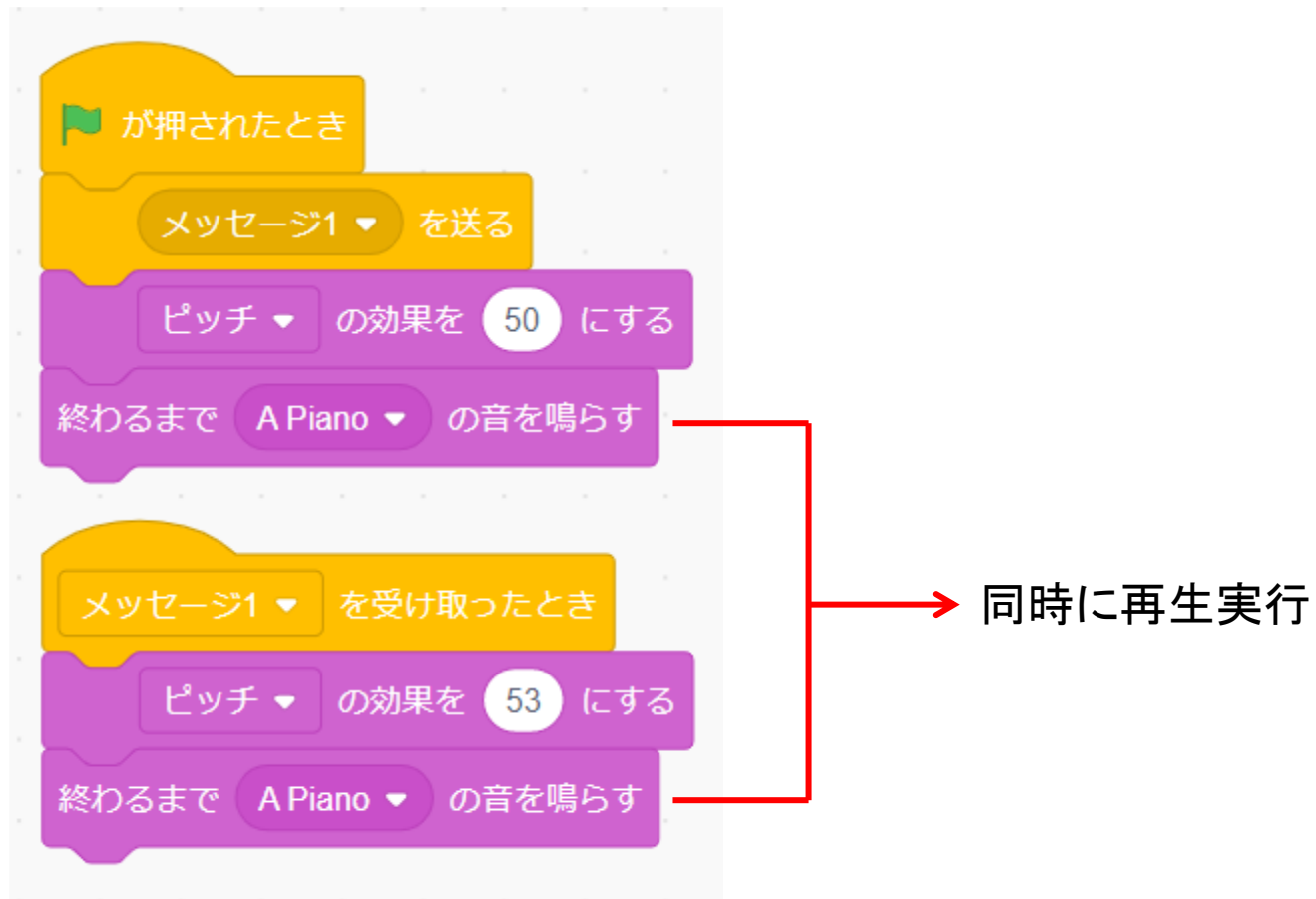
7. 音の使用例: 音の合成・うなり音(3): サウンド素材の追加方法(3)

これにより、デフォルトの「ニャー」音に加え、「A Piano」(ピアノのう音)のサウンド素材をScratchプログラムで使用可能になります。



7. 音の使用例: 音の合成・うなり音(4): プログラム例

追加した音素材“A Piano”(ピアノのう音)を選択し、ピッチを50%と53%の2種(220Hzと232Hz)に変化させて合奏させる。合奏させる場合は、マルチスレッド機能を用いて2本の再生プログラムを作成し、メイン側のプログラムで再生する直前にメッセージを送り、もう一方のプログラムに同期再生させる。参考32ページ “Scratch3_beet.sb3”



7. 音の使用例(2):ネコとピアノの音階合奏(1):プログラム例

2つの音素材("ニャー"と"A Piano")を選択し、音階に合わせてピッチを変化させ音階を合奏する。ドレミファソラシドに対応するノートナンバーをC=0,2,4,5,7,9,11,12とすると、ピッチ[%]は、 $100 \cdot 2^{(C-6)/12} = 100 \cdot (10^{0.301})^{C/12}$ で算出でき、 $P = 100 \cdot 10^{0.301 \cdot (C-6)/12} - 100$ をサウンド再生時のピッチ効果として与える。

"Scratch3_audio.sb3"

The screenshot displays the Scratch3 interface with the following components:

- Code Editor:**
 - When Green Flag Clicked:**
 - Set C to 0
 - Loop: While C > 12, repeat:
 - Set P to $C - 6 / 12 \cdot 0.3010$
 - Set P to $P \cdot 10^{\Delta} \cdot 100 - 100$
 - Set Pitch effect to P
 - Set volume to 100%
 - Play sound "A Piano"
 - Set volume to 40%
 - Play sound "ニャー"
 - Set C to C + 1, wait 0.8 seconds
 - Set C to C + 2
 - Define - Patch:**
 - While C = 6, set C to 5
 - While C = 13, set C to 12
 - Define - Scale Setting:**
 - Remove all scale names
 - Add: ド(Do), ド#(Do#), レ(Re), レ#(Re#), ミ(Mi), ファ(Fa), ファ#(Fa#), ソ(Sol), ソ#(Sol#), ラ(La), ラ#(La), シ(Si), ド(Do)
- Properties Panel:**
 - Sound:** P: -10.909103, C: 4
 - Scale:** List of 13 notes: 1 ド(Do), 2 ド#(...), 3 レ(Re), 4 レ#(...), 5 ミ(Mi), 6 ファ(Fa), ..., 13
 - Sprite:** Sprite1, x: 0, y: 0, size: 100, direction: 90
 - Stage:** Background 1

7. 音の使用例(2):ネコとピアノの音階合奏(2):プログラム部拡大

"Scratch3_audio.sb3"

Scratch 3.0 スクリプトブロックのスクリーンショット。左側のメインスクリプトは、フラグ「が押されたとき」で開始し、以下の処理を実行する。

- 階名設定
- C を 0 にする
- ループ: C > 12 まで繰り返す
 - P を $C - 6 / 12 * 0.3010$ にする
 - P を $P \text{ の } 10^{\wedge} * 100 - 100$ にする
 - ピッチ の効果を P にする
 - 音量を 100 %にする (ピアノ伴奏の音量調整)
 - A Piano の音を鳴らす
 - 音量を 40 %にする (ネコの鳴き声の音量調整)
 - ニャー の音を鳴らす
 - 階名 の C + 1 番目 と 0.8 秒言う
 - C を C + 2 にする
- 補正

Cの値でそのままピッチを計算すると、ピッチが高すぎるため、-6半音下げる。

定義 補正

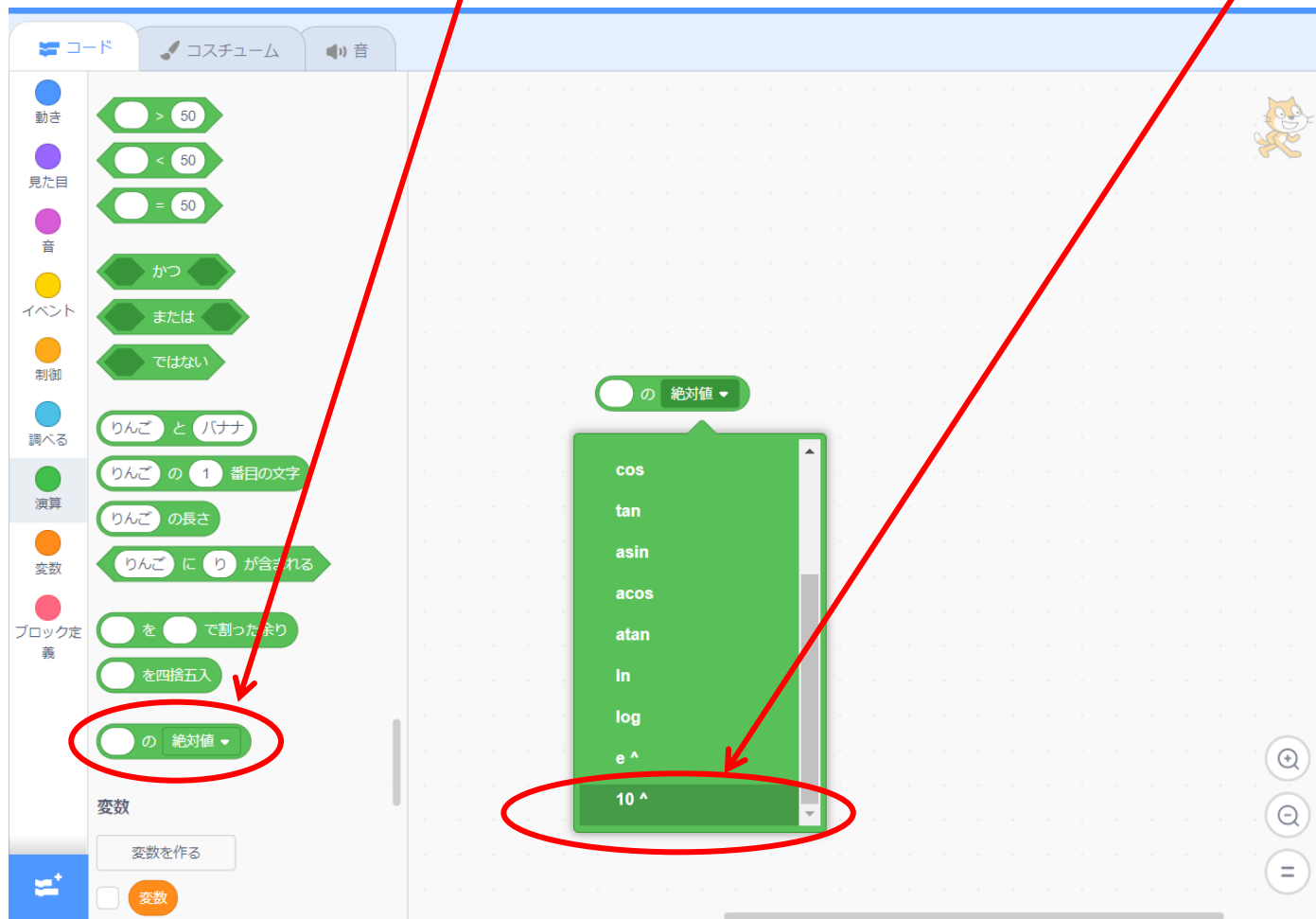
- もし C = 6 なら
 - C を 5 にする
- もし C = 13 なら
 - C を 12 にする

定義 階名設定

- 階名 のすべてを削除する
- ド(Do) を 階名 に追加する
- ド#(Do#) を 階名 に追加する
- レ(Re) を 階名 に追加する
- レ#(Re#) を 階名 に追加する
- ミ(Mi) を 階名 に追加する
- ファ(Fa) を 階名 に追加する
- ファ#(Fa#) を 階名 に追加する
- ソ(Sol) を 階名 に追加する
- ソ#(Sol#) を 階名 に追加する
- ラ(La) を 階名 に追加する
- ラ#(La#) を 階名 に追加する
- シ(Si) を 階名 に追加する
- ド(Do) を 階名 に追加する

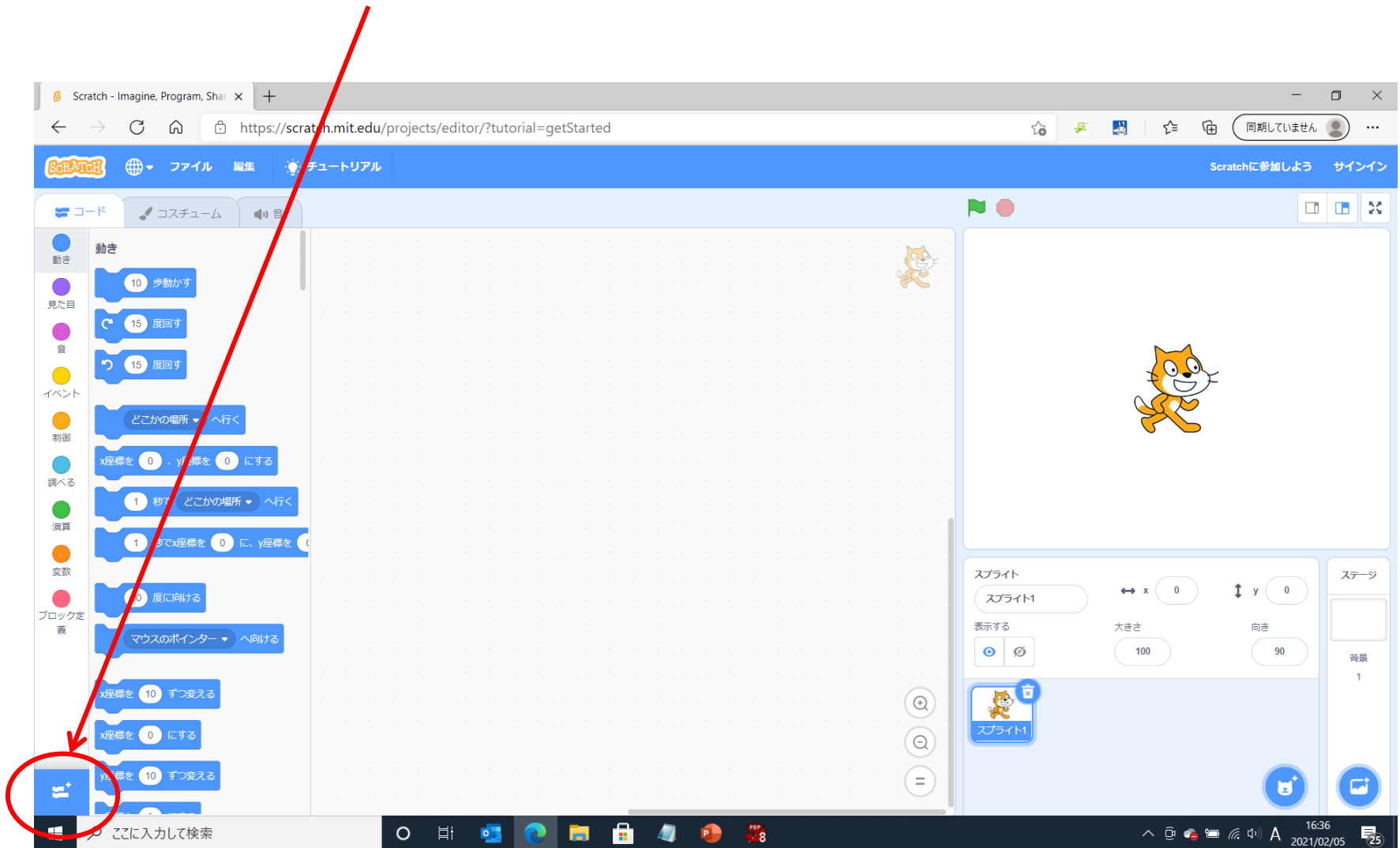
7. 音の使用例(2):ネコとピアノの音階合奏(3):指数関数の使用方法

音階のピッチを算出するにあたり、2の累乗という指数演算が必要です。Scratchでは指数の底として、自然対数(exponential)と常用対数(10)のいずれかしか使用できません。そこで、 2^x は、 $10^{0.301x}$ と常用対数の底10を用いて計算します。指数関数は左欄の「演算」メニューより「の絶対値」を使用し、絶対値をクリックして所望の関数(本例では「 10^A 」)を選択します。



8. 拡張機能「音楽(MIDI機能)」の使用方法(1)

左下の拡張機能アイコンをクリックします。



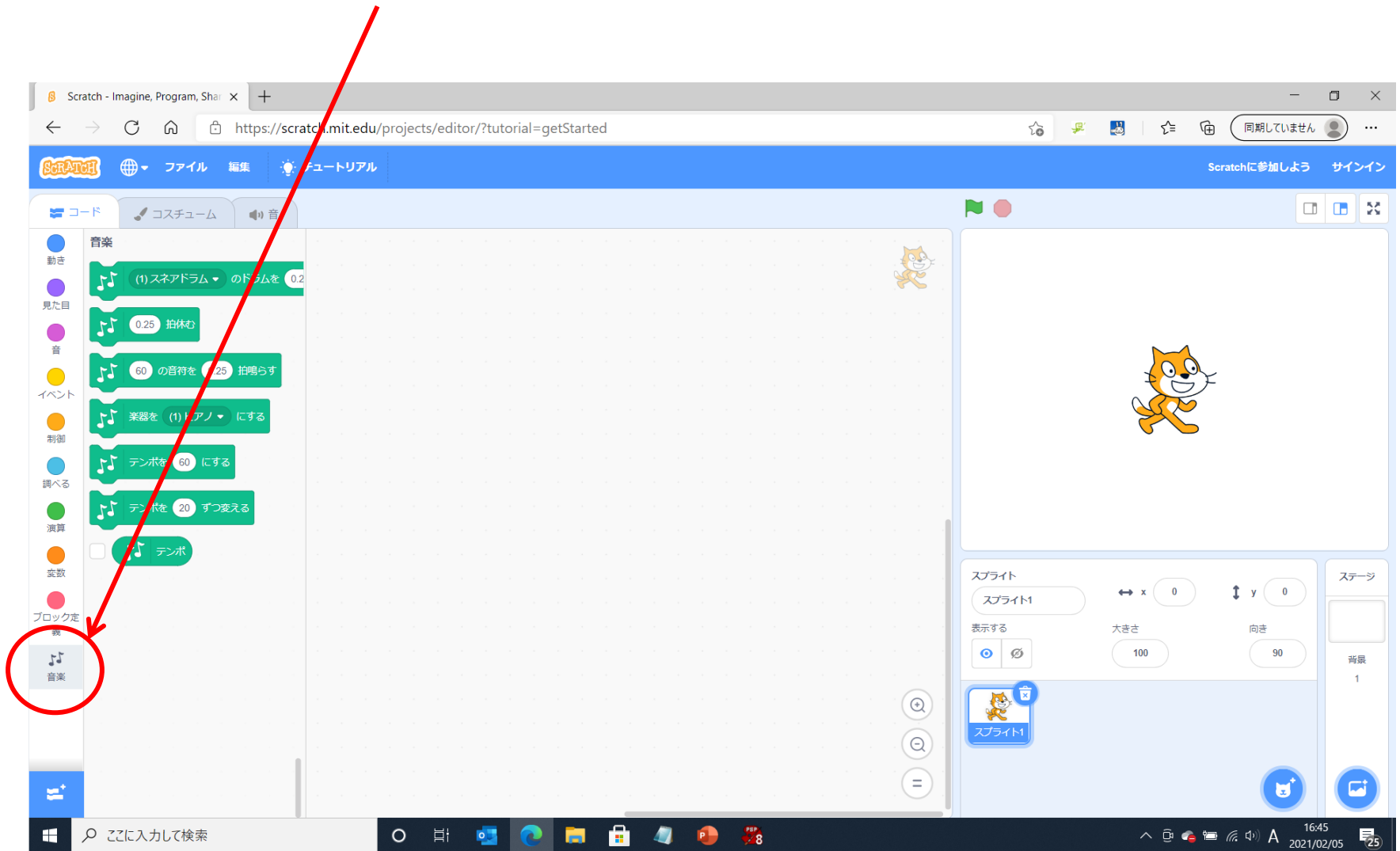
8. 拡張機能「音楽(MIDI機能)」の使用方法(2)

下記の拡張機能メニューで「音楽」を選択します。



8. 拡張機能「音楽(MIDI機能)」の使用方法(3)

左端のメニューの最下部に拡張機能「音楽」が追加されています。



8. 拡張機能「音楽(MIDI機能)」の使用例(1) : ハ長調の音階

"Scratch3_MIDI.sb3"

MIDI機能を用いて、ドレミファソラシド(C3, D3, E3, F3, G3, A3, C4)の音階を再生します。

2・3番目のブロックで楽器やテンポを変更できます。

MIDIノートナンバー

楕円枠をクリックすると音楽鍵盤が表示されるので、鳴らしたい音符をクリックします。



例題の音階では面白みがないので、次ページの和音の入力方法も参照の上、皆さんがご存じのメロディー(単旋律および和声)を打ち込んでみてください。

8. 拡張機能「音楽(MIDI機能)」の使用例(2) : ハ長調の音階＋和音付け

和音を付加したい音符の直前に、「イベント」の「メッセージ(例. 和音C)を送る」を追加します。

和音C

和音Cを受け取ったとき

48 の音符を 0.25 拍鳴らす C2(ド)

和音Cを受け取ったとき

52 の音符を 0.25 拍鳴らす E2(ミ)

和音Cを受け取ったとき

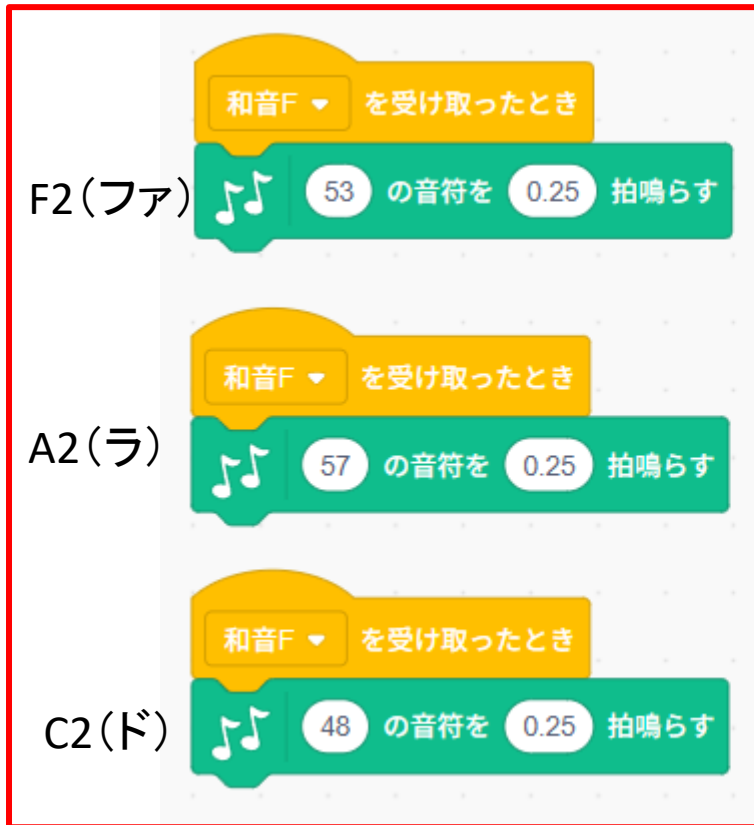
55 の音符を 0.25 拍鳴らす G2(ソ)

和音の構成音の各音符ごとに、「イベント」の「メッセージ(例. 和音C)を受け取ったとき」と「()の音符を()拍鳴らす」を組み合わせたブロックを定義する。

8. 拡張機能「音楽(MIDI機能)」の使用例(2) : ハ長調の音階＋和音付け(2)

"Scratch3_MIDIh.sb3"

和音F



Scratch3 MIDI script for F major triad. The script consists of three blocks, each triggered by the 'When I receive F chord' event. Each block plays a specific note for 0.25 seconds.

和音F ▼ を受け取ったとき	音符	拍鳴らす
F2(ファ)	53	0.25
A2(ラ)	57	0.25
C2(ド)	48	0.25

和音G



Scratch3 MIDI script for G major triad. The script consists of three blocks, each triggered by the 'When I receive G chord' event. Each block plays a specific note for 0.25 seconds.

和音G ▼ を受け取ったとき	音符	拍鳴らす
G2(ソ)	55	0.25
B2(シ)	59	0.25
D2(レ)	50	0.25

8. 拡張機能「音楽(MIDI機能)」の使用例(3) : MIDIシーケンサ

前ページで紹介した単音と和音が混在するメロディーを簡便に作成できる、汎用的なMIDIシーケンサを作成します。

前ページでは、標準搭載の音符を鳴らすブロック「()の音符を()拍鳴らす」にて、音符のノートナンバーを直接指示していましたが、以下のように、英語の音名「C(ド)D(レ)E(ミ)F(ファ)G(ソ)A(ラ)B(シ)」とオクターブ番号で指示可能にします。

			ド		レ		ミ	ファ		ソ		ラ		シ	ド		
音名	C-2	--	C3	C#3	D3	D#3	E3	F3	F#3	G3	G#3	A3	A#3	B3	C4	--	G8
				Db3		Eb3			Gb3		Ab3		Bb3				
ノート ナンバー	0	--	60	62	63	64	65	66	67	68	69	70	71	72		--	127

標準搭載の音符を鳴らすブロック「()の音符を()拍鳴らす」に対して、独自ブロック「ノート単音(音名)(長さ、拍数)」と「ノート和音(音名)(長さ、拍数)」を定義した。「ノート単音」ブロックは、標準搭載の音符を鳴らすブロックと同等な機能で、実行すると、シーケンサの時刻が拍数だけずれる。一方、「ノート和音」ブロックは、実行してもシーケンサの時刻は変化しないため、後続の「ノート単音」ブロックまたは「ノート和音」ブロックで指示された音符は同時刻に発音され和音となる。シーケンサの時刻をずらしたい場合は、標準搭載のブロック「()拍休む」を使用する。

8. 拡張機能「音楽(MIDI機能)」の使用例(3) : MIDIシーケンサ(2)

Scratch 3.0 での音楽制作プロジェクトのスクリーンショット。スコアは以下の通りです。

スコア:

- テンポを 80 にする
- 楽器を (1) ピアノ にする
- 音量を 100 %にする
- 三和音 C2 E2 G2
- ノート単音 C3 0.25
- ノート単音 D3 0.25
- ノート単音 E3 0.25
- 三和音 F2 A2 C2
- ノート単音 F3 0.25
- ノート単音 G3 0.25
- ノート単音 A3 0.25
- 三和音 G2 B2 D2
- ノート単音 B3 0.25
- 三和音 C2 E2 G2
- ノート単音 C4 0.25

コードブロック:

- 定義 ノート単音 note 長さ
- ノートナンバー変換 note
- 定義 ノートナンバー変換2 note
- 定義 ノートナンバー変換3 note

スプライト1: NN 72

スプライト1: C 8

スプライト1: oct 4

スプライト1: P 2

スプライト1: NN2 -1

スプライト1: 長さ 0.25

音名		ナンバー	
1	C	1	0
2	D	2	2
3	E	3	4
4	F	4	5
5	G	5	7
6	A	6	9
7	B	7	11

+ 長さ 7 =

+ 長さ 7 =

「スプライト1:NN」など4種の変数は、各スプライト内で参照可能なローカル変数にする必要があります。詳細は、「描画の高速化処理“Scratch_advanceh_v3.pdf”」の第12章を参照ください。

「音名」と「ナンバー」は本資料の第6章で説明した配列(リスト)で、7個の音名をノートナンバーに変換するための変換テーブルを定義しています。独自ブロック「ノートナンバー変換」で参照しています。

8. 拡張機能「音楽(MIDI機能)」の使用例(3) : MIDIシーケンサ(3) MIDIデータ部

前ページの使用例(2)で紹介した単音と和音が混在するメロディーの事例は右のように作成できます。

「和音C」、「和音F」、「和音G」の3種類の和音は、独自定義ブロック「三和音」でまとめることができ、3つの和音構成音を引数で指定します。

「ノート単音」ブロックを実行すると、シーケンサの時刻が拍数だけずれる。

一方、「ノートと音」ブロックは、実行してもシーケンサの時刻は変化しないため、後続の「ノート単音」ブロックまたは「ノートと音」ブロックで指示された音符は同時刻に発音され和音となる。

シーケンサの時刻をずらしたい場合は、標準搭載のブロック「() 拍休む」を使用する。

The image displays a Scratch3 MIDIseq.sb3 score editor. The main sequence of blocks is as follows:

- が押されたとき (When clicked)
- テンポを 60 にする (Set tempo to 60)
- 楽器を (1) ピアノ にする (Set instrument to (1) Piano)
- 音量を 100 %にする (Set volume to 100 %)
- NN2 を -1 にする (Set NN2 to -1)
- 三和音 C2 E2 G2 (Triad C2 E2 G2)
- ノート単音 C3 0.25 (Note C3 0.25)
- ノート単音 D3 0.25 (Note D3 0.25)
- ノート単音 E3 0.25 (Note E3 0.25)
- 三和音 F2 A2 C2 (Triad F2 A2 C2)
- ノート単音 F3 0.25 (Note F3 0.25)
- ノート単音 G3 0.25 (Note G3 0.25)
- ノート単音 A3 0.25 (Note A3 0.25)
- 三和音 G2 B2 D2 (Triad G2 B2 D2)
- ノート単音 B3 0.25 (Note B3 0.25)
- 三和音 C2 E2 G2 (Triad C2 E2 G2)
- ノート単音 C4 0.25 (Note C4 0.25)

On the right, a custom block named "定義 三和音" (Define Triad) is shown with three arguments: N1, N2, and N3. Below it, three "ノートと音" (Note and Chord) blocks are listed, each with a duration of 0.25:

- ノートと音 N1 0.25
- ノートと音 N2 0.25
- ノートと音 N3 0.25

"Scratch3_MIDIseq.sb3"

8. 拡張機能「音楽(MIDI機能)」の使用例(3) : MIDIシーケンサ(4) シーケンサ部

The image displays a collection of Scratch 3.0 code blocks for a MIDI sequencer, organized into five functional sections:

- 定義 ノート単音 note 長さ**
 - ノートナンバー変換 note
 - 🎵 NN の音符を 長さ 拍鳴らす
- 定義 ノート単音 note 長さ**
 - 長さ を 長さ にする
 - ノートナンバー変換 note
 - NN2 を NN にする
 - メッセージH を送る
- 定義 ノート単音 note 長さ**
 - メッセージH を受け取ったとき
 - もし NN2 > -1 なら
 - 🎵 NN2 の音符を 長さ 拍鳴らす
 - NN2 を -1 にする
- 定義 ノートナンバー変換 note**
 - C を 1 にする
 - NN を 0 にする
 - C > 7 まで繰り返す
 - もし note の 1 番目の文字 = 音名 の C 番目 なら
 - NN を ナンバー の C 番目 にする
 - C を 7 にする
 - C を C + 1 にする
 - ノートナンバー変換2 note
- 定義 ノートナンバー変換2 note**
 - P を 2 にする
 - もし note の P 番目の文字 = # なら
 - NN を NN + 1 にする
 - P を P + 1 にする
 - もし note の P 番目の文字 = b なら
 - NN を NN - 1 にする
 - P を P + 1 にする
 - ノートナンバー変換3 note
- 定義 ノートナンバー変換3 note**
 - もし note の P 番目の文字 = - なら
 - oct を note の P + 1 番目の文字 にする
 - oct を 0 - oct にする
 - でなければ
 - oct を note の P 番目の文字 にする
 - NN を 12 * oct + 2 + NN にする

"Scratch3_MIDIseq.sb3"

8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(1)

前ページで紹介したMIDIシーケンサを用いて、実際の楽曲データを入力する際、Scratchの独自ブロック機能を用いて小節単位にブロック定義をすると効率が良いです。

具体例として、近年流行の音楽生成AIの元祖と言える、[モーツァルト作曲「音楽のサイコロ遊び」作品\(K.516f\)](#)、"[musical dice game](#)"を一部抜粋して作成してみます。

この音楽作品は16小節で構成され、下表のように、各小節ごとに11パターンのフレーズ(和音付きで、全 $16 \times 11 = 176$ パターン)が用意されています。各小節を演奏するごとに、2個のサイコロを振り、2個のサイコロの目の和(2~12)に基づいて演奏するフレーズを決定します。

Dice	小節1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	96	22	141	41	105	122	11	30	70	121	26	9	112	49	109	14
3	32	6	128	63	146	46	134	81	117	39	126	56	174	18	116	83
4	69	95	158	13	153	55	110	24	66	139	15	132	73	58	145	79
5	40	17	113	85	161	2	159	100	90	176	7	34	67	160	52	170
6	148	74	163	45	80	97	36	107	25	143	64	125	76	136	1	93
7	104	157	27	167	154	68	118	91	138	71	150	29	101	162	23	151
8	152	60	171	53	99	133	21	127	16	155	57	175	43	168	89	172
9	119	84	114	50	140	86	169	94	120	88	48	166	51	115	72	111
10	98	142	42	156	75	129	62	123	65	77	19	82	137	38	149	8
11	3	87	165	61	135	47	147	33	102	4	31	164	144	59	173	78
12	54	130	10	103	28	37	106	5	35	20	108	92	12	124	44	131

8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(2)

ここでは、前ページ表の左上4×4マスに対応する、冒頭の4小節と各小節に対してサイコロの目の和(2～5)の4パターン(全16パターン)を用いて、4小節の楽曲を4回繰り返し再生するプログラムを作成します。各回、各小節ごとにサイコロを振り直し、4通りのBGMを生成します。

具体的には、原曲に付与された下記小節番号のフレーズをランダムに組み合わせます。

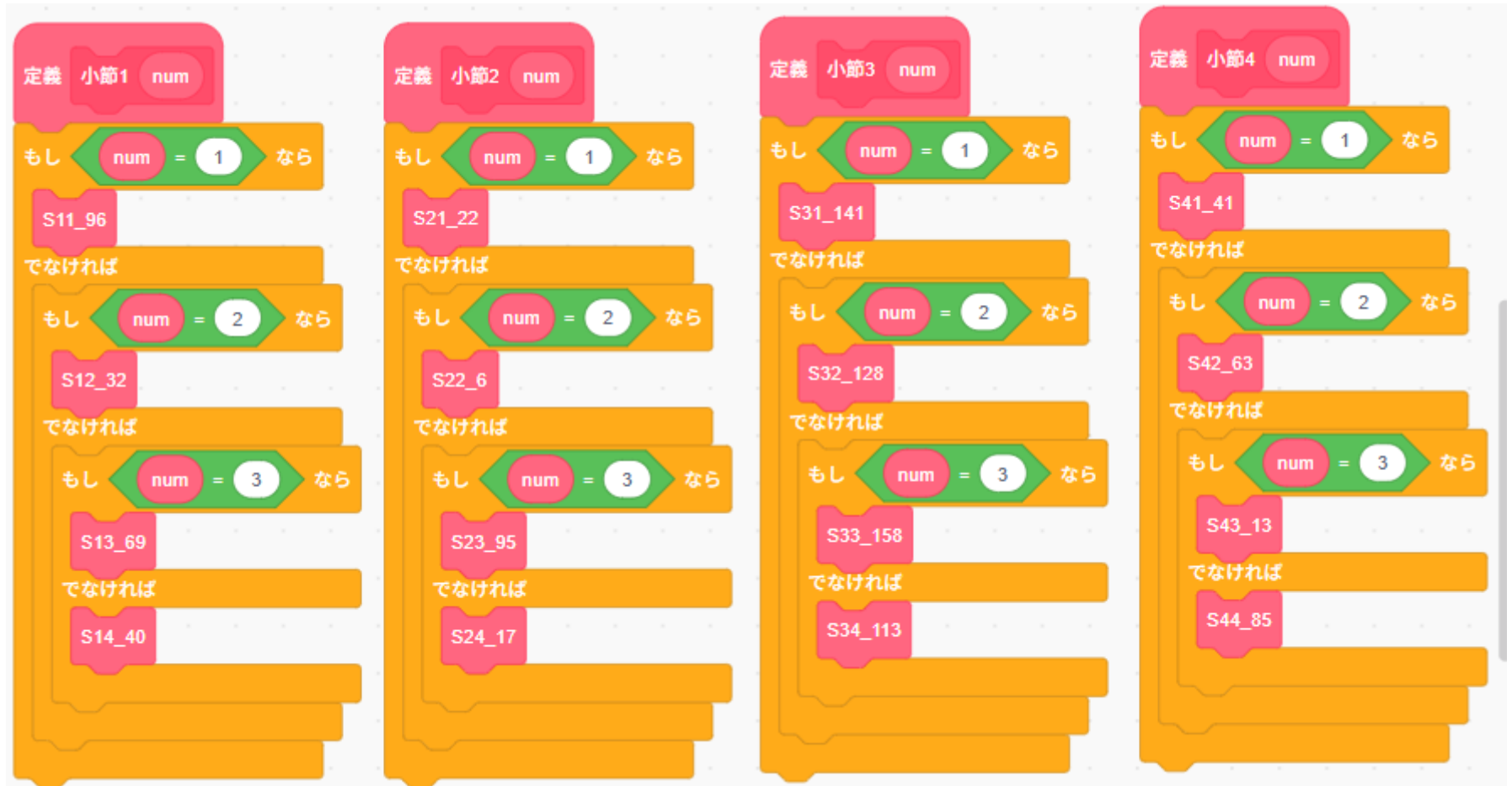
	第1小節	第2小節	第3小節	第4小節
(2)	96	22	141	41
(3)	32	6	128	63
(4)	69	95	158	13
(5)	40	17	113	85

興味のある方は、[原曲の176小節の譜面](#)をダウンロードして、前ページの表に基づいて全曲を完成させてみてください。



"Scratch3_MIDIseq_mozart.sb3"

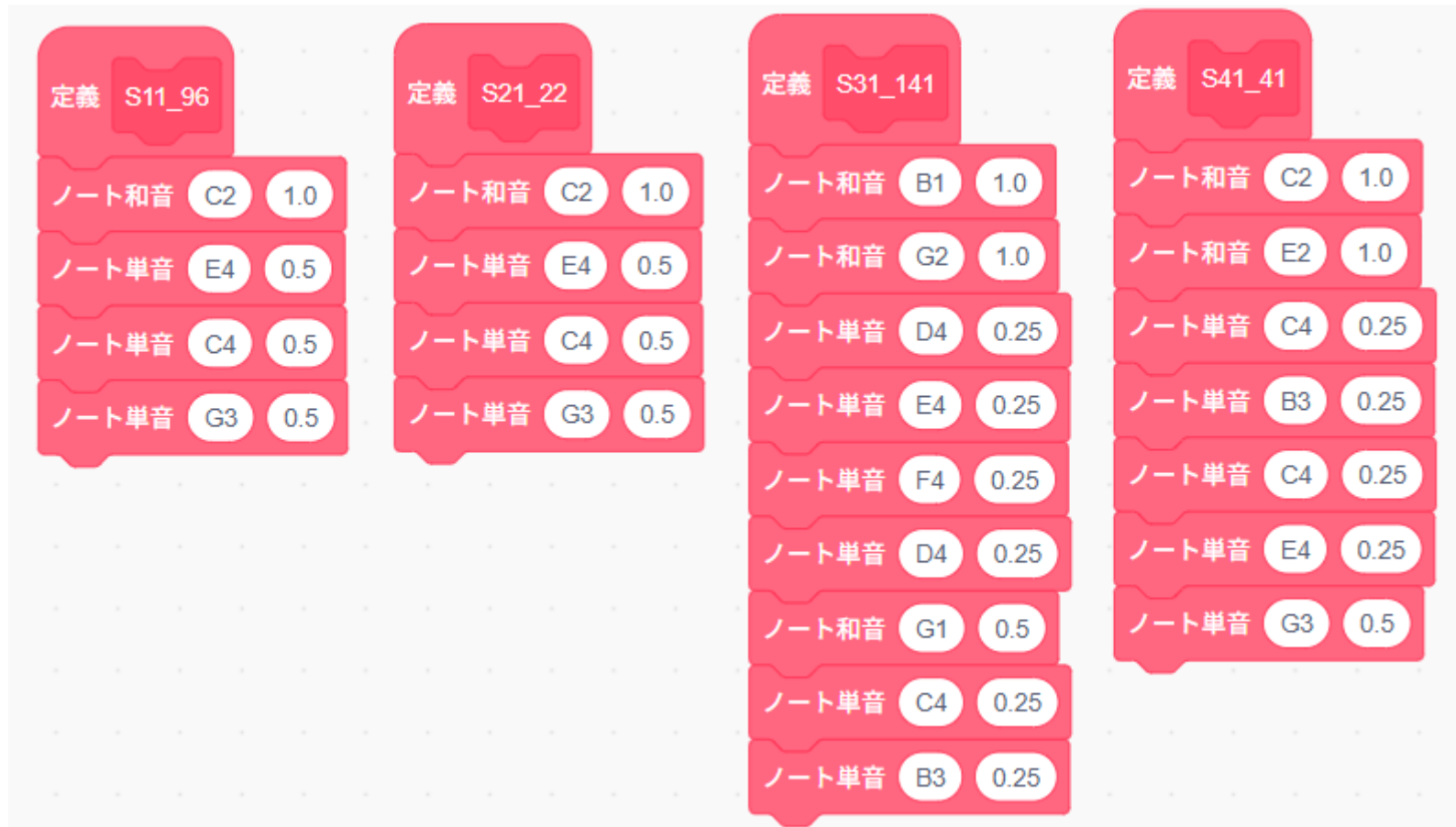
8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(3)



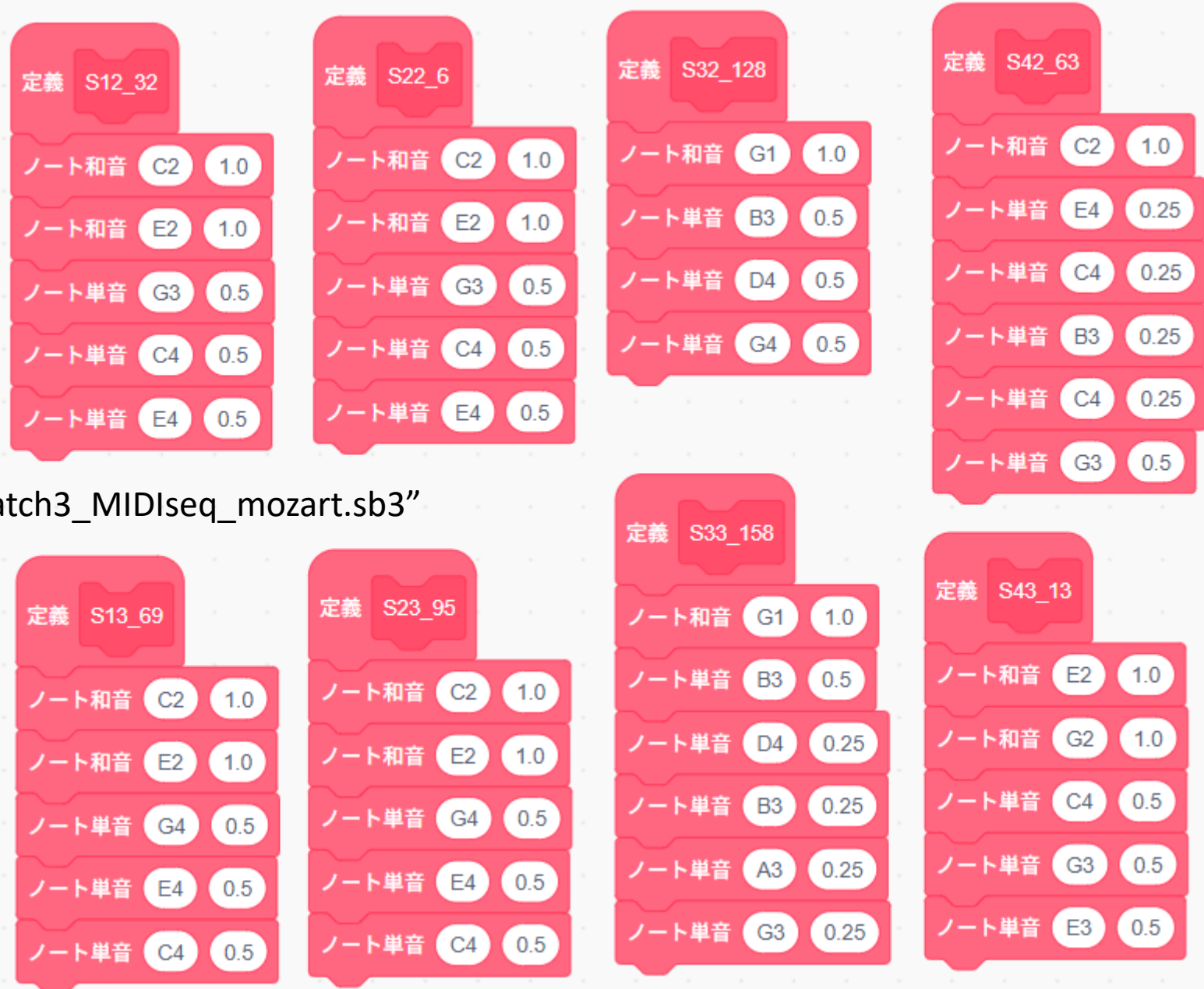
"Scratch3_MIDIseq_mozart.sb3"

8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(4)

「ノート単音」ブロックおよび「ノート和音」ブロックは、前ページで紹介したMIDIシーケンサと同様です。



8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(5)



"Scratch3_MIDIseq_mozart.sb3"

8. 拡張機能「音楽(MIDI機能)」の使用例(4) : モーツァルトのサイコロ音楽(6)

The image displays a Scratch3 MIDI sequence editor with four tracks, each containing a series of musical notes. The tracks are labeled as follows:

- 定義 S14_40**
 - ノート和音 C2 1.0
 - ノート和音 E2 1.0
 - ノート単音 C4 0.25
 - ノート単音 B3 0.25
 - ノート単音 C4 0.25
 - ノート単音 E4 0.25
 - ノート単音 G3 0.25
 - ノート単音 C4 0.25
- 定義 S24_17**
 - ノート和音 E2 1.0
 - ノート和音 G2 1.0
 - ノート単音 C4 0.25
 - ノート単音 G3 0.25
 - ノート単音 C4 0.25
 - ノート単音 E4 0.25
 - ノート単音 G3 0.25
 - ノート単音 C4 0.25
- 定義 S34_113**
 - ノート和音 G2 1.0
 - ノート和音 B2 1.0
 - ノート単音 F4 0.5
 - ノート単音 D4 0.5
 - ノート単音 B3 0.5
- 定義 S44_85**
 - ノート和音 E2 1.0
 - ノート和音 G2 1.0
 - ノート単音 C4 0.5
 - ノート単音 E4 0.5
 - ノート単音 G3 0.5

"Scratch3_MIDIseq_mozart.sb3"

8. 拡張機能「音楽(MIDI機能)」の使用例(5) : マルチトラックのサポート

前ページまでに紹介した単音と和音が混在するメロディーはシングルトラック(単一MIDIチャンネル)で構成されていました。

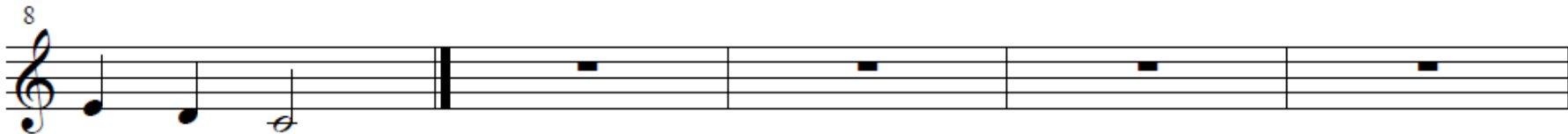
右記の通り、先の音階メロディーから、和音演奏部(Cのコードのシーケンス)を別のトラックに分離して定義することもできます。

「ノート単音」ブロックおよび「ノートと音」ブロックは、前ページと同様です。

The image displays two Scratch3 MIDI sequence tracks. The left track, triggered by a 'when clicked' event, sets tempo to 60, piano instrument, and volume to 100%. It then sends a message to the right track and plays a sequence of single notes: C3 (0.25), D3 (0.25), E3 (0.25), F3 (0.25), G3 (0.25), A3 (0.25), B3 (0.25), and C4 (0.25). The right track, triggered by 'when message 1 received', plays a sequence of chords: C2-E2-G2 (0.75), F2-A2-C2 (0.75), G2-B2-D2 (0.75), and C2-E2-G2 (0.25). Below this sequence is a definition block for chords N1, N2, and N3, each with a duration of 0.25.

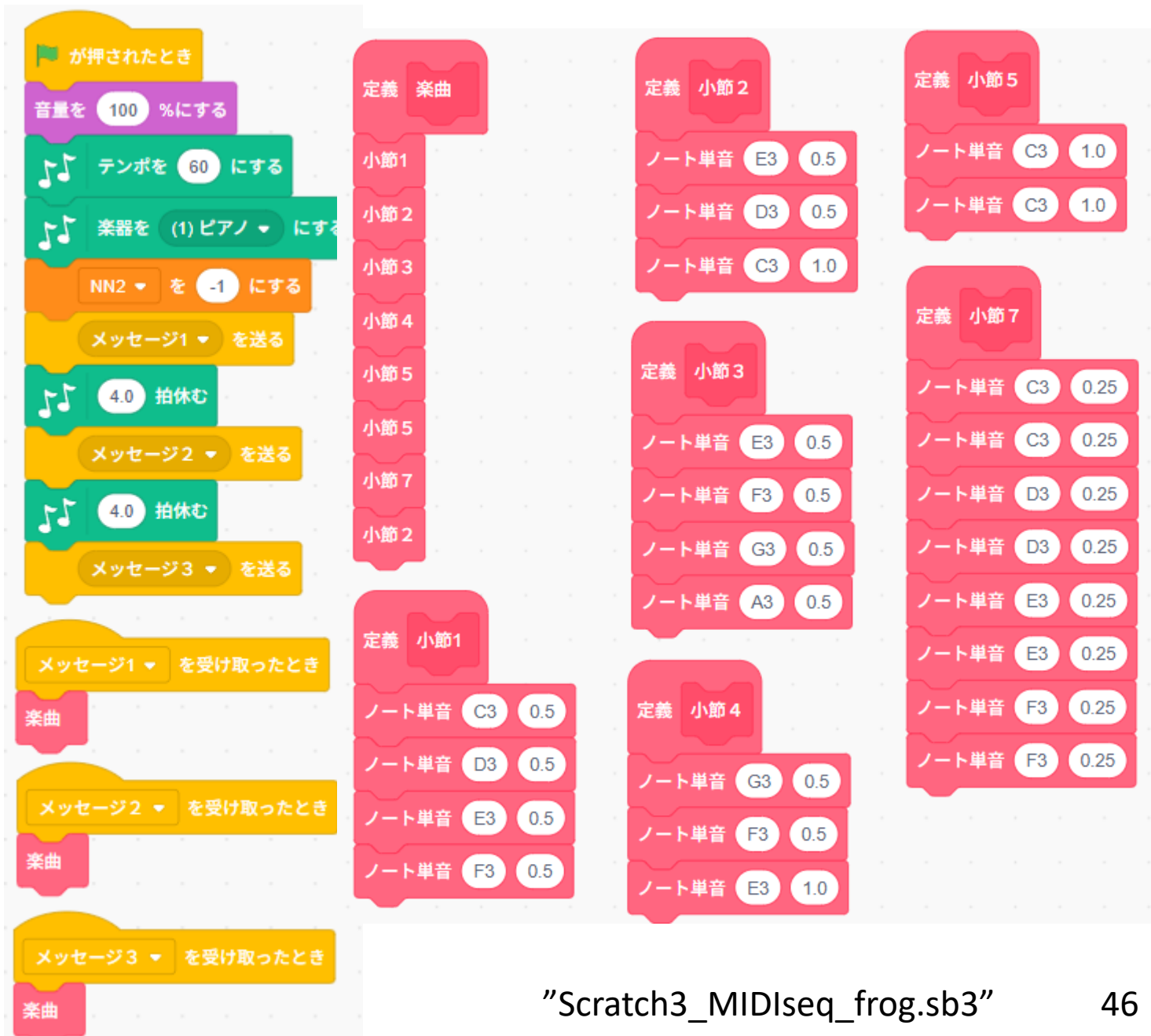
8. 拡張機能「音楽(MIDI機能)」の使用例(6) : 「かえるの歌」輪唱への応用

前ページのマルチトラックを応用し、下記「[かえるの歌](#)」を3声のカノンで3トラックで作成してみます。



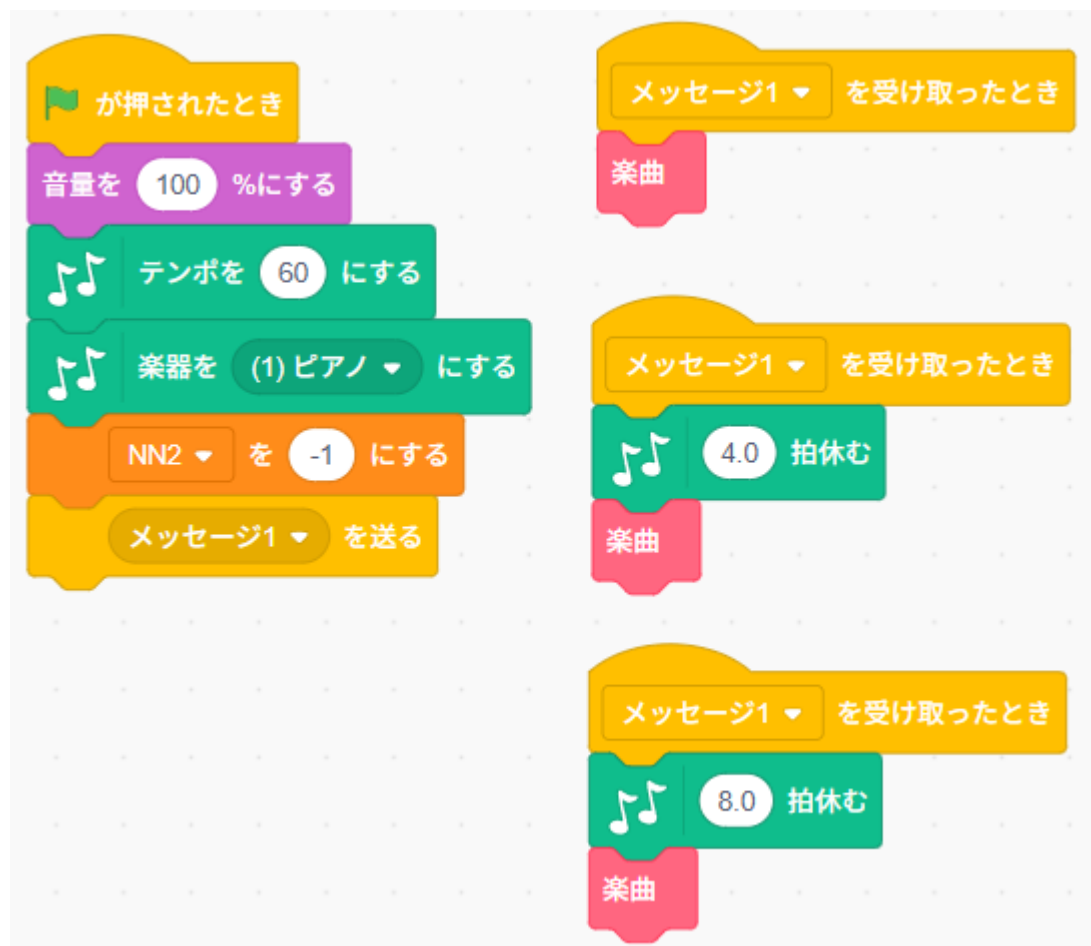
8. 拡張機能「音楽(MIDI機能)」の使用例(6) : 「かえるの歌」輪唱への応用(2)

3種のメッセージを送って3スレッドで実行させる必要があります。「ノート単音」ブロックは、前ページと同様です。



8. 拡張機能「音楽(MIDI機能)」の使用例(6) : 「かえるの歌」輪唱への応用(3)

前ページでは3種のメッセージを送る方法でしたが、下記のように単一のメッセージを送る構成をとることもできます。



8. 拡張機能「音楽(MIDI機能)」の使用例(6) : 「かえるの歌」輪唱への応用(4)

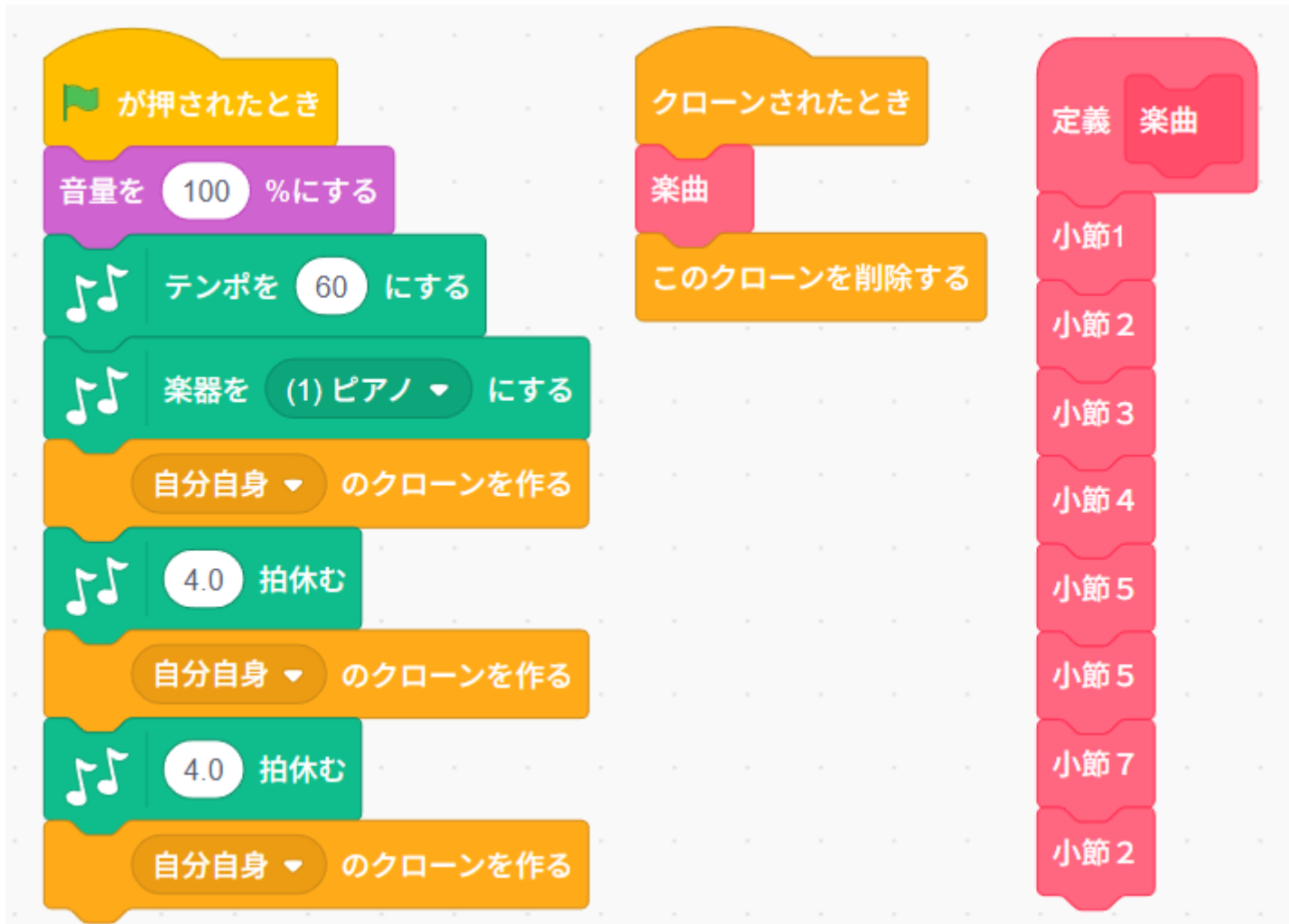
前ページの「かえるの歌」の事例では「ノート和音」を使用していないため、下記のように「MIDIシーケンサにおける独自ブロック「ノートと音」は削除することができ、次ページのようにクローンを生成する方法で簡単に3スレッドを実行させることもできます。

The image displays two parallel Scratch3 MIDIseq_frog2.sb3 code blocks. The left block defines a function 'ノート単音' (Note Single) which takes 'note' and 'length' as inputs, converts the note to a MIDI number (NN), and plays it for the specified length. The right block defines a function 'ノートナンバー変換2' (Note Number Conversion 2) which takes 'note' as input and converts it to a MIDI number (NN). Both blocks are part of a larger sequence that generates a melody. The left block includes a loop that iterates through 8 notes, while the right block includes a loop that iterates through 8 notes, with a final calculation for the next note's MIDI number.

Scratch3_MIDIseq_frog2.sb3

8. 拡張機能「音楽(MIDI機能)」の使用例(6) : 「かえるの歌」輪唱への応用(5)

独自ブロック「楽曲」を構成する「小節1」「小節2」・・・は、前ページと同様です。



8. 拡張機能「音楽(MIDI機能)」の使用例(4) : メトロノーム(リズムマシン)

テンポ(BPM、例:120)と拍子(例:4)を入力すると、ドラム音でリズムを刻むとともに、リズムに合わせてキャラクターが回転する。

”Scratch3_rhythm.sb3”

The image displays the Scratch 3.0 development environment. On the left, the 'Scripts' area shows a 'when green flag clicked' event block followed by a 'main' script block. The 'main' script block contains the following code:

- テンポ を 120 にする
- 拍子 を 4 にする
- 大きさを 200 %にする
- テンポを テンポ にする
- C を 1 にする

On the right, the 'main' script block contains the following code:

- スペース キーが押された まで繰り返す
- $C - 1 * 360 / 拍子 + 90$ 度に向ける
- もし $C = 1$ なら
- 大きさを 200 %にする
- (4) クラッシュシンバル のドラムを 1 拍鳴らす
- でなければ
- 大きさを 150 %にする
- (1) スネアドラム のドラムを 1 拍鳴らす
- C を $C + 1$ にする
- もし $C > 拍子$ なら
- C を 1 にする

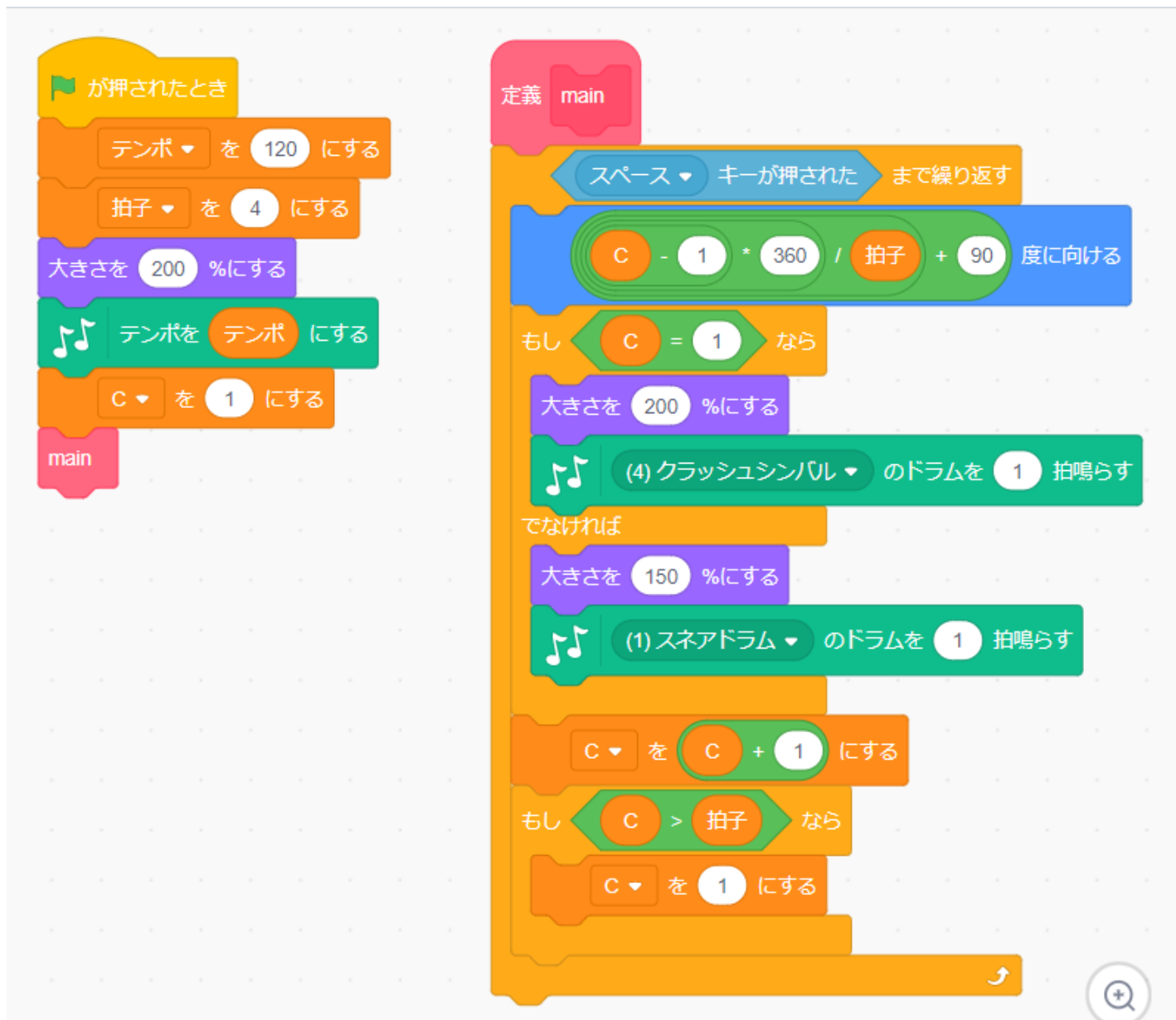
On the right side of the interface, the 'Stage' area shows a character (Scratch cat) with the following properties:

- テンポ: 120
- 拍子: 4
- C: 4
- スプライト: スプライト1
- 表示する: 表示する
- 向き: 0
- 大きさ: 150

The character is shown in a dynamic pose, indicating it is rotating.

8. 拡張機能「音楽(MIDI機能)」の使用例(4) : メトロノーム(2):プログラム部

"Scratch3_rhythm.sb3"



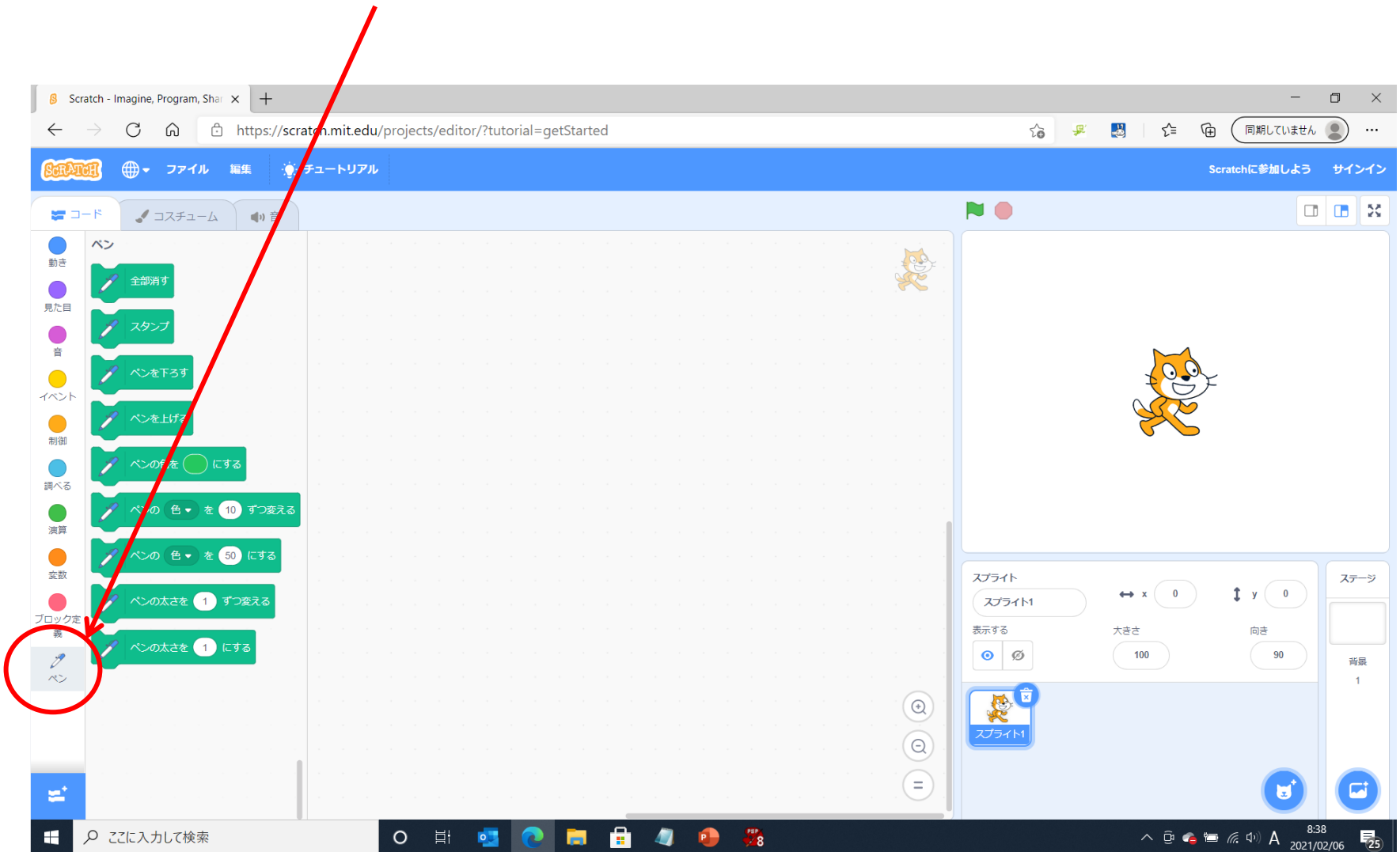
9. 拡張機能「ペン(2次元グラフィックス機能)」の使用方法(1)

下記の拡張機能メニューで「ペン」を選択します。



9. 拡張機能「ペン(2次元グラフィックス機能)」の使用方法(2)

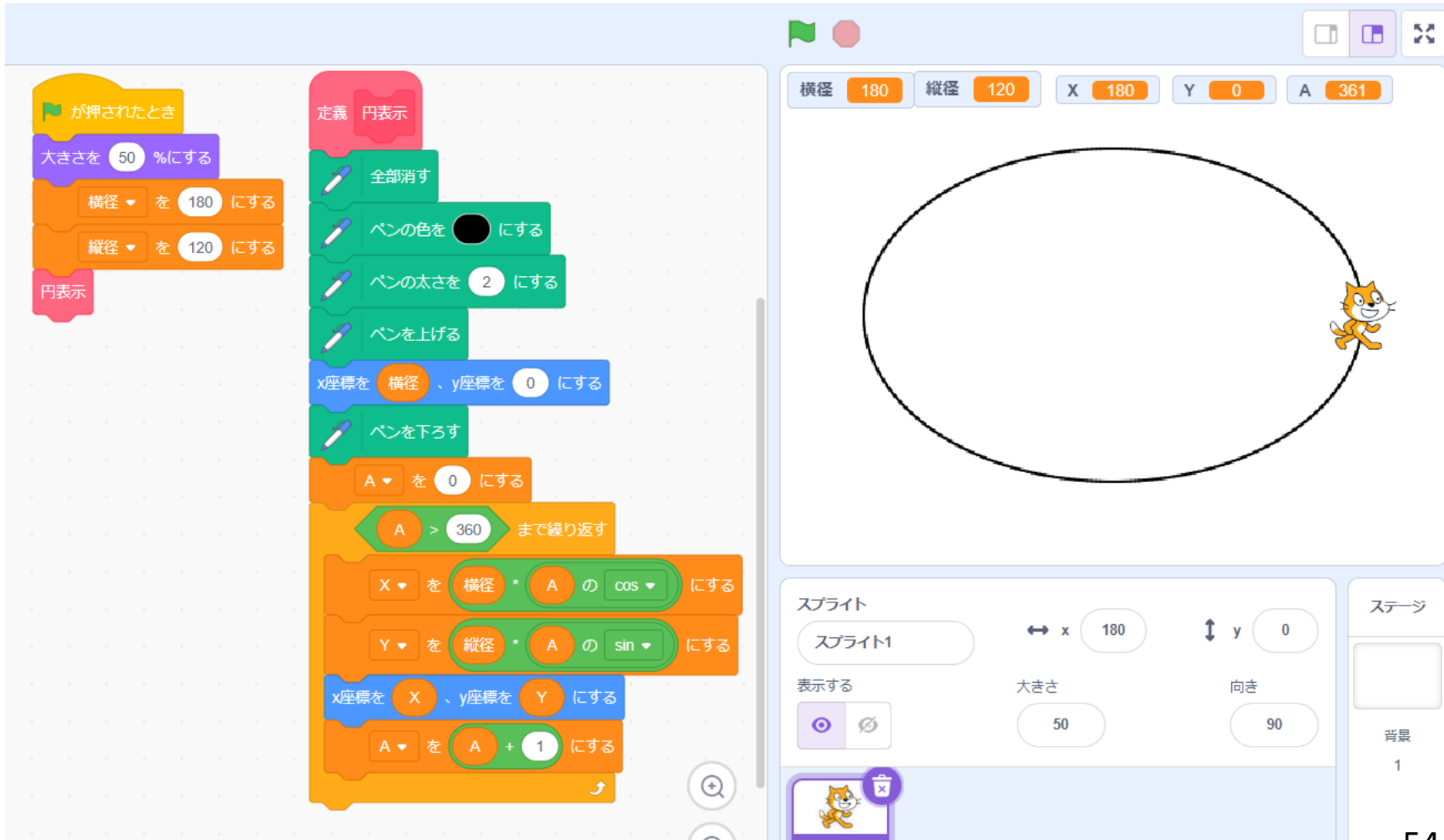
左端のメニューの最下部に拡張機能「ペン」が追加されています。



9. 拡張機能「ペン(2次元グラフィックス)」の使用例: 楕円の描画(1)

角度Aを0～360度に変化させ、 $X = (\text{横径}) \times \cos(A)$ 、 $Y = (\text{縦径}) \times \sin(A)$ で示される座標(X,Y)をプロットすることにより楕円形状を描画する。

"Scratch3_circle.sb3"



The image displays the Scratch3 interface with a script designed to draw an ellipse. The script is as follows:

```
when green flag clicked
  set size to 50%
  set width to 180
  set height to 120
  define circle drawing
  clear
  set pen color to black
  set pen thickness to 2
  lift pen
  set x coordinate to width, y coordinate to 0
  set pen down
  set A to 0
  loop until A > 360
    set X to width * A's cos
    set Y to height * A's sin
    set x coordinate to X, y coordinate to Y
    set A to A + 1
```

The stage view on the right shows a black ellipse with a width of 180 and a height of 120. The x and y coordinates are 180 and 0, respectively, and the angle A is 361. The sprite is a cat character.

9. 拡張機能「ペン」の使用例: 楕円の描画(2): 三角関数の使用方法



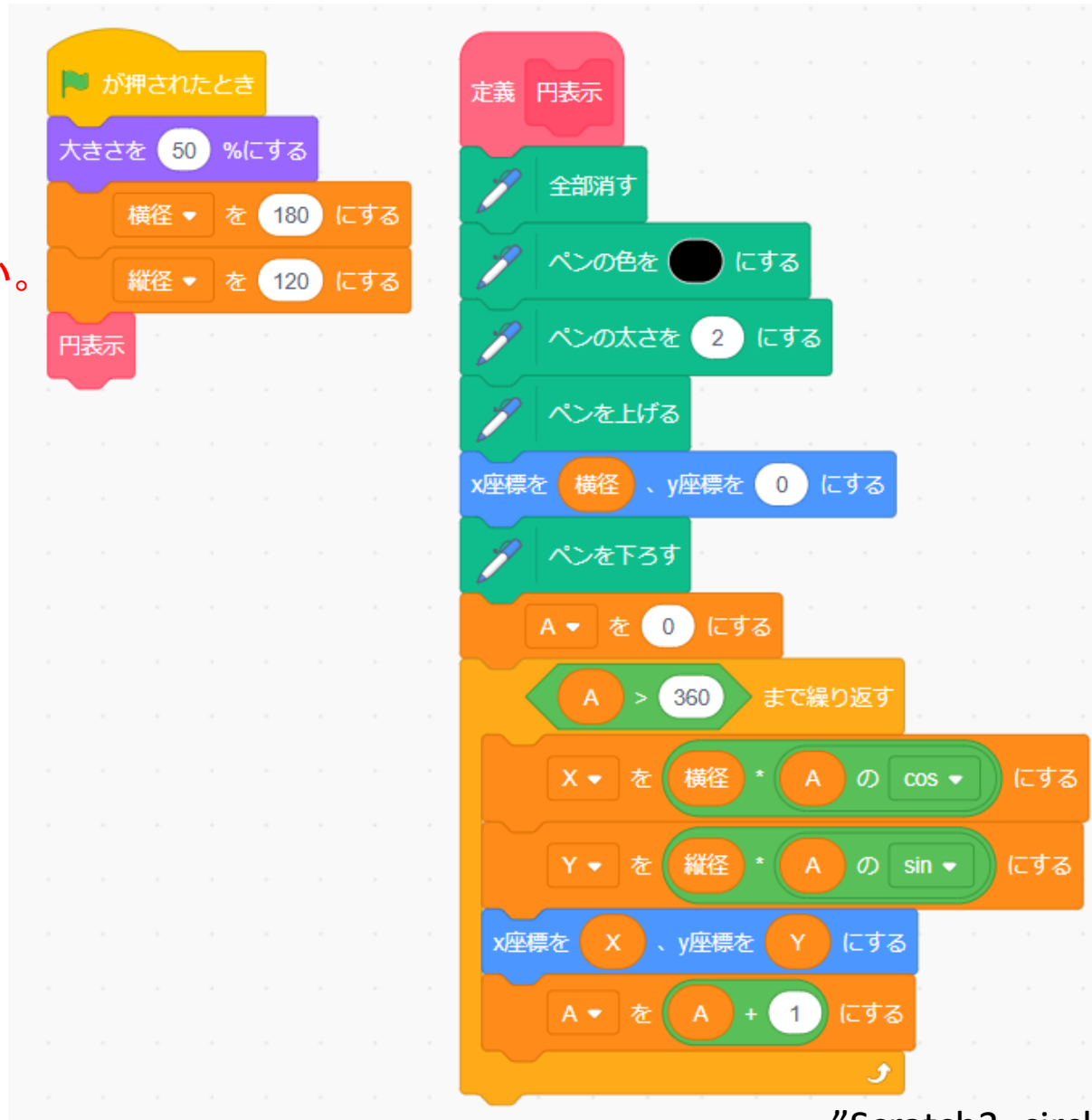
The image shows the Scratch web interface. On the left, the 'Code' tab is selected, and the 'Math' category is highlighted in the block palette. A script of green flag clicked blocks is visible: a 'say' block, a 'wait' block, and a 'say' block. The 'Math' category is expanded, showing various mathematical blocks. The 'sin' block is highlighted. A red text overlay explains the steps to select the 'sin' function from the 'Math' category.

左端メニュー「演算」より「の絶対値」を選択し、「絶対値」のプルダウンメニューより「sin」を選択します。

三角関数sinに渡す角度はラジアンではなく、360度単位で、出力は-1~+1の実数値です。

9. 拡張機能「ペン」の使用例: 楕円の描画(3): プログラム部拡大

横径、縦径は適当
に変更してください。



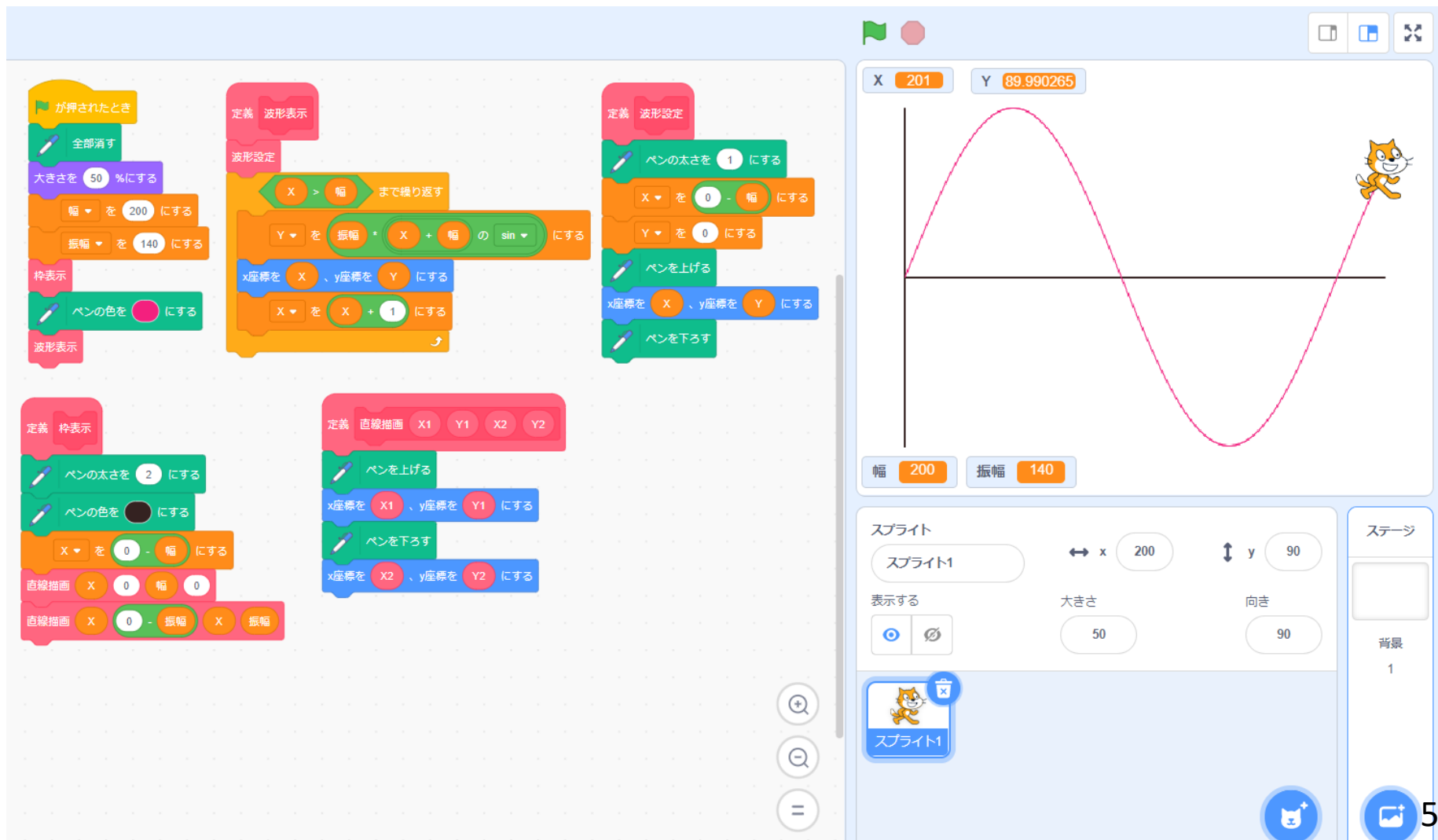
"Scratch3_circle.sb3"

9. 拡張機能「ペン(2次元グラフィックス)」の使用例(2): サインカーブ(1)

X1をペンでプロットするX座標 ($-(\text{幅}) \leq X1 \leq +(\text{幅})$) で、かつ角度とする。

$Y2 = (\text{振幅}) \times \sin(X1)$ で示される座標 $(X1, Y1)$ をプロットすることによりサインカーブを描画する。

”Scratch3_sine.sb3”



The image shows the Scratch3 interface with a script for drawing a sine wave. The script is divided into three main sections: initialization, wave drawing, and axis drawing.

Initialization (When Green Flag Clicked):

- 全部消す (Clear all)
- 大きさを 50 %にする (Set size to 50%)
- 幅 を 200 にする (Set width to 200)
- 振幅 を 140 にする (Set amplitude to 140)
- 枠表示 (Show frame)
- ペンの色を 黒 にする (Set pen color to black)
- 波形表示 (Show wave)

Wave Drawing (Define Wave Drawing):

- 定義 波形表示 (Define wave drawing)
- 波形設定 (Waveform settings)
- ペン設定 (Pen settings)
- ペン太さを 1 にする (Set pen thickness to 1)
- X を 0 - 幅 にする (Set X to 0 - width)
- Y を 0 にする (Set Y to 0)
- ペンを上げる (Lift pen)
- X座標を X 、Y座標を Y にする (Set X coordinate to X, Y coordinate to Y)
- X を X + 1 にする (Increase X by 1)
- ペンを下ろす (Lower pen)

Axis Drawing (Define Axis Drawing):

- 定義 枠表示 (Define frame drawing)
- ペンの太さを 2 にする (Set pen thickness to 2)
- ペンの色を 黒 にする (Set pen color to black)
- X を 0 - 幅 にする (Set X to 0 - width)
- 直線描画 X 0 幅 0 (Draw line from X=0 to X=width)
- 直線描画 X 0 - 振幅 X 振幅 (Draw line from X=0 to X=amplitude)

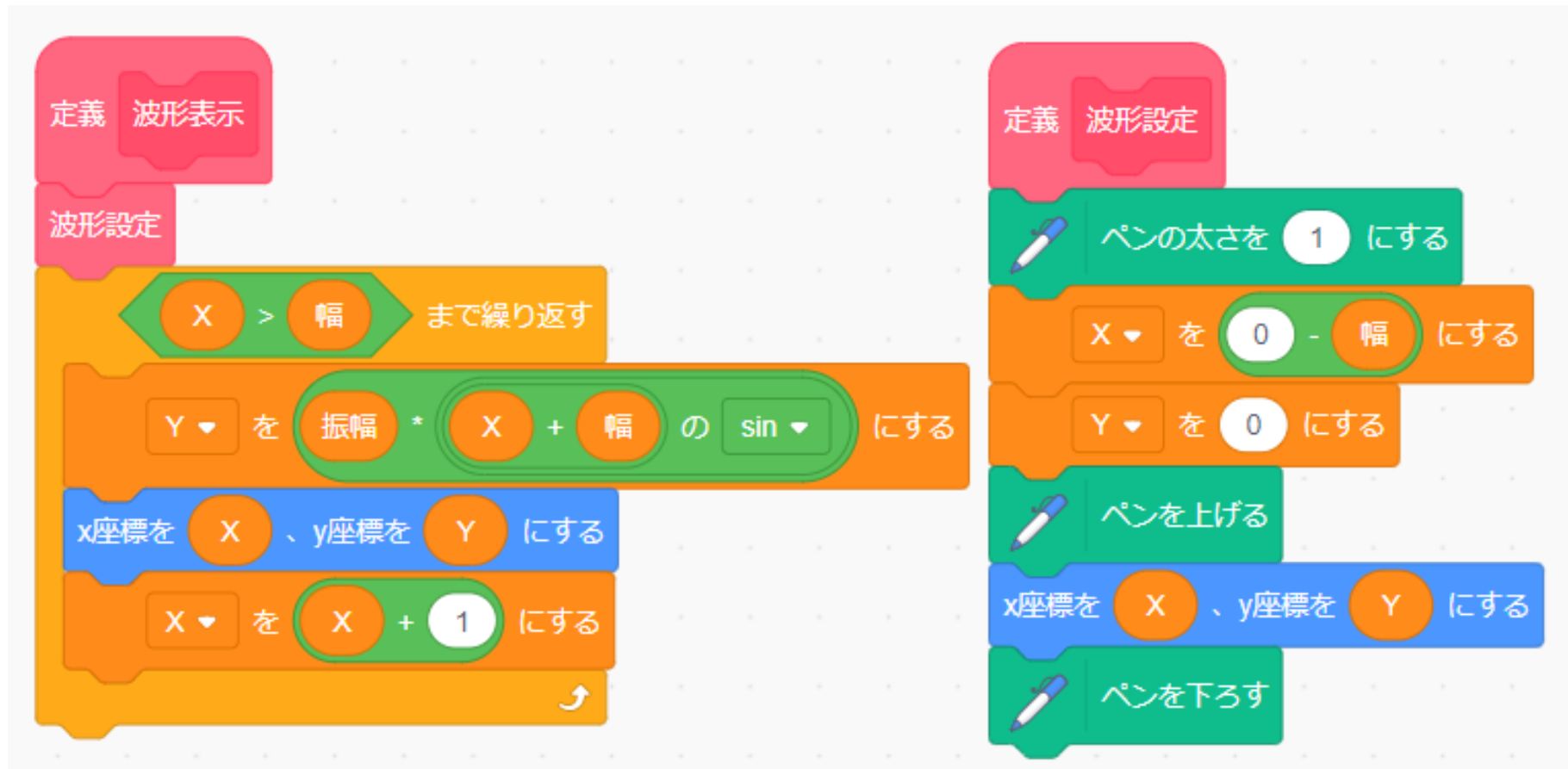
Interface Elements:

- Stage: X 201, Y 89.990265, Width 200, Amplitude 140
- Sprite: スプライト1 (Sprite 1)
- Background: 背景 1 (Background 1)

9. 拡張機能「ペン」の使用例(2):サインカーブ(2):プログラム部拡大(1)



9. 拡張機能「ペン」の使用例(2):サインカーブ(2):プログラム部拡大(2)



9. 拡張機能「ペン(2次元グラフィックス)」の使用例(3): 波形合成・うなり(1)

周期がわずかに異なる2種のサインカーブ(前ページのサインカーブの10周期と9周期分)を生成し、2種のサインカーブと合成波形をプロットする。音響再生はできないが、合成波形は1周期分のうなり音になっている。参考16-19ページ "Scratch3_beetw.sb3"

The image displays the Scratch 3.0 interface for a waveform synthesis project. The script editor on the left contains the following logic:

- When clicked:** Clear all, set size to 50%, set amplitude to 200, set frequency to 140.
- Pen display:** Set pen thickness to 2, color to black, and line width to 0.
- Pen settings:** Set pen color to red, blue, and green.
- Waveform display:** Set wave display to 10, 0, and 9.
- Pen up/down:** Move the pen up and down to start and end the drawing.
- Waveform settings:** Set wave settings to 1, 0, and 9.
- Waveform display:** Set wave display to 10, 0, and 9.
- Waveform settings:** Set wave settings to 1, 0, and 9.

The stage shows a plot of three sine waves: a green one with a period of 10, a blue one with a period of 9, and a purple one which is their sum, creating a beat pattern. The plot area has X and Y coordinates, and sliders for amplitude and frequency.

9. 拡張機能「ペン」の使用例(3): 波形合成・うなり(2) : プログラム部拡大(1)

The image displays a Scratch script for a waveform synthesis program, organized into two columns. The script begins with a 'when green flag clicked' event block, followed by a 'clear' block. It then sets the 'size' to 50%, 'width' to 200, and 'amplitude' to 140. A 'show ruler' block is used, followed by setting 'amplitude' to 70. The pen color is set to purple, and a 'waveform display' block is shown with 'x' at 10 and 'y' at 0. The pen color is then set to blue, and another 'waveform display' block is shown with 'x' at 9 and 'y' at 0. Finally, the pen color is set to green, and a third 'waveform display' block is shown with 'x' at 10 and 'y' at 9.

が押されたとき

全部消す

大きさを 50 %にする

幅 を 200 にする

振幅 を 140 にする

枠表示

振幅 を 70 にする

ペンの色を 紫色 にする

波形表示 10 0

ペンの色を 青 にする

波形表示 9 0

ペンの色を 緑 にする

波形表示 10 9

定義 枠表示

ペンの太さを 2 にする

ペンの色を 黒 にする

X を 0 - 幅 にする

直線描画 X 0 幅 0

直線描画 X 0 - 振幅 X 振幅

定義 直線描画 X1 Y1 X2 Y2

ペンを上げる

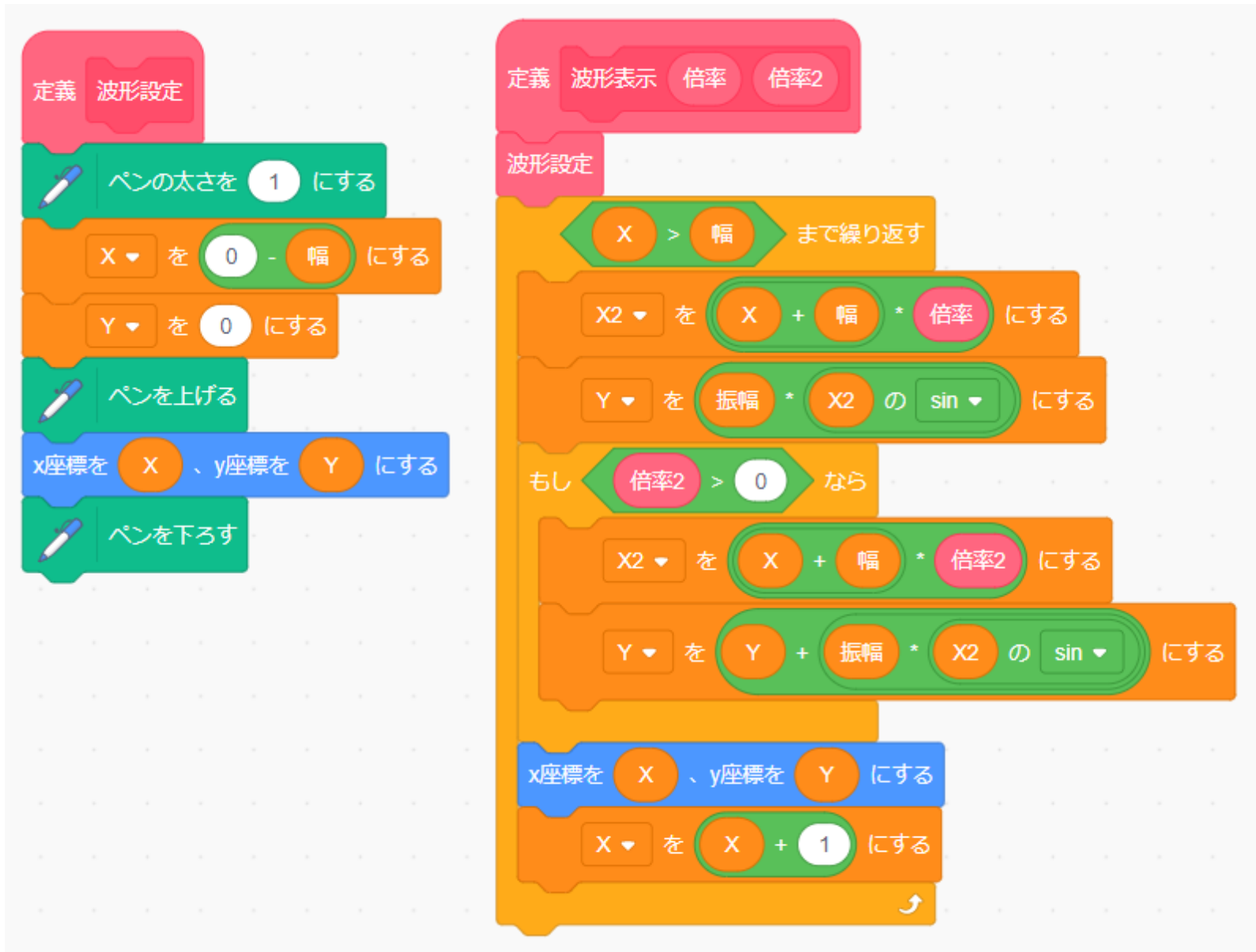
x座標を X1 、y座標を Y1 にする

ペンを下ろす

x座標を X2 、y座標を Y2 にする

Scratch3_beetw.sb3

9. 拡張機能「ペン」の使用例(3): 波形合成・うなり(2): プログラム部拡大(2)



"Scratch3_beetw.sb3"

9. 拡張機能「ペン」の使用例(3):お絵描き(1)

マウスをクリックした点(ペンダウン)から、再度クリックする点(ペンアップ)までの間、マウスを動かすことにより自由曲線を描くことができる(描画中はボタンを離した状態)。

”Scratch3_draw.sb3”

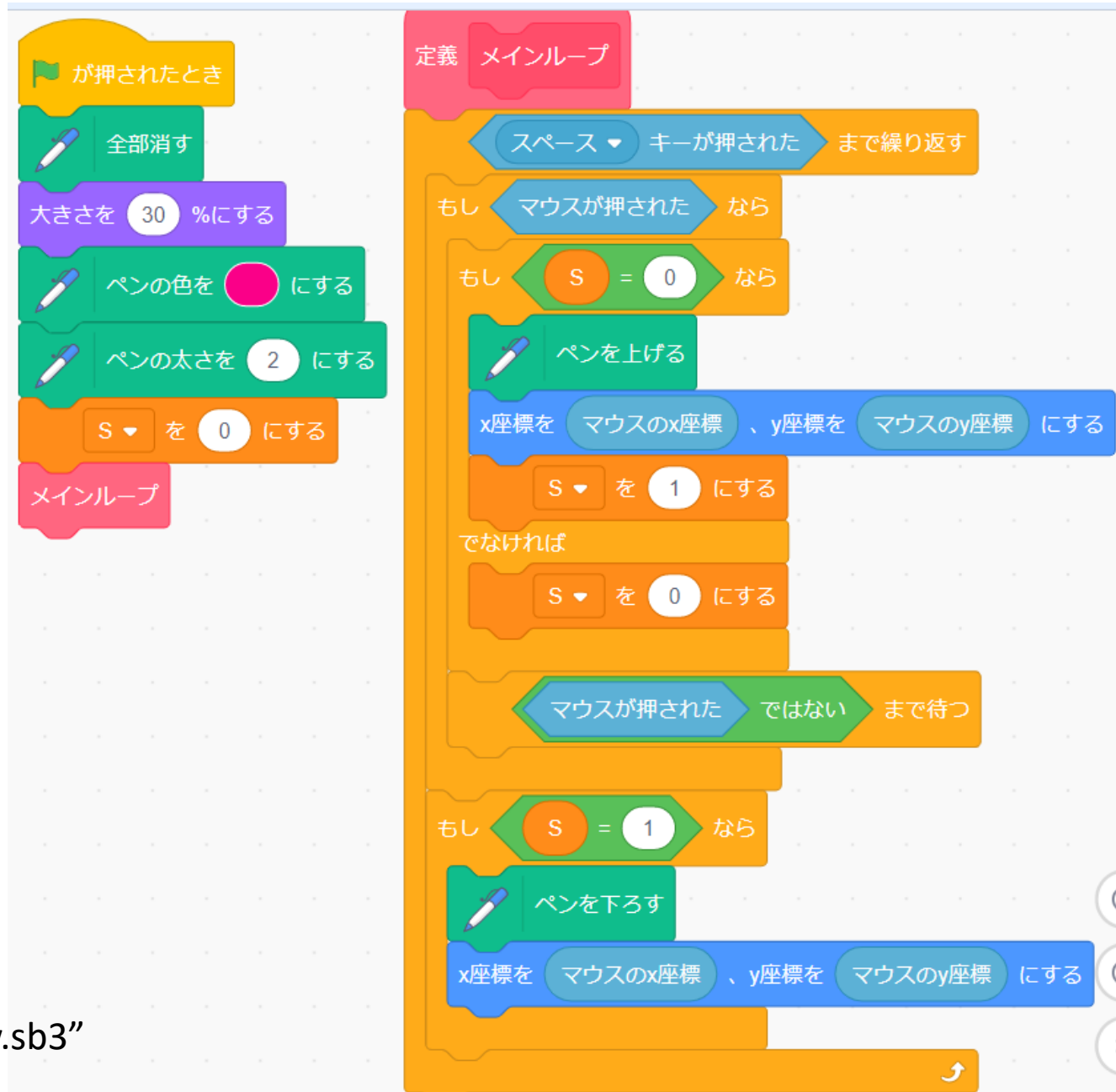
The image displays the Scratch3_draw.sb3 script and stage. The script is as follows:

```
when green flag clicked
  clear
  set brush size to 30%
  set brush color to magenta
  set brush thickness to 2
  set s to 0
  main loop
  when space key pressed
    loop until
      mouse clicked
    if mouse clicked
      if s = 0
        pen up
        set x coordinate to mouse x coordinate
        set y coordinate to mouse y coordinate
        set s to 1
      else
        set s to 0
      wait until mouse clicked
    if s = 1
      pen down
      set x coordinate to mouse x coordinate
      set y coordinate to mouse y coordinate
  loop
```

The stage shows the word "Scratch" drawn in magenta. The sprite panel shows the Scratch cat sprite. The stage background is white.

9. 拡張機能「ペン」の使用例(3):お絵描き(2):プログラム部拡大

ペンの色や太さは自由に変更できます。



"Scratch3_draw.sb3"

10. 拡張機能「ペン」と「音楽」の使用例:ピアノ鍵盤(1)

拡張機能「ペン」を用いてピアノ鍵盤を描画し、先の拡張機能「音楽」を用いてマウスでクリックすることにより対応するキーの音階を鳴らすことができる。

"Scratch3_piano.sb3"

The image displays the Scratch3 piano project interface, showing the code blocks on the left and the stage view on the right.

Code Blocks:

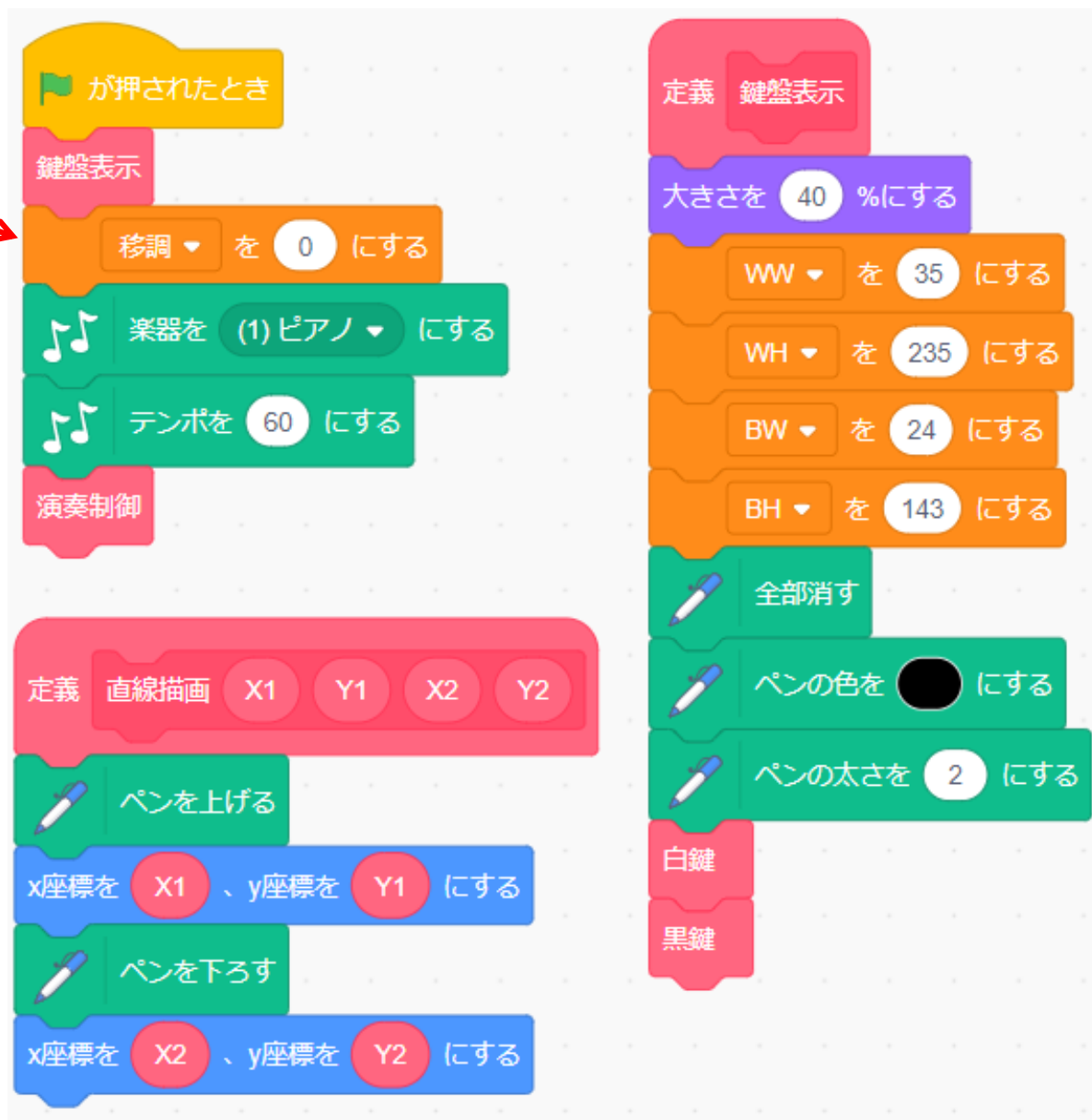
- When Green Flag Clicked:** Set stage display, set tempo to 60, set key to (1) Piano.
- Define: Stage Display:** Set size to 40%, set WW to 35, set WH to 235, set BW to 24, set BH to 143.
- Define: White Key:** Set X to 0 + WW / 4, set Y to 0 + WH / 2, set X2 to WW / 4, set Y2 to WH / 2. Draw white key, set pen color to black, set pen size to 2.
- Define: Black Key:** Set pen size to BW, set Y to Y2 + BH, set C to 1. Loop: set X2 to X + WW + C, draw black key, set C to C + 1. If C = 3 or C = 7, set C to C + 1.
- Define: Music Setup:** Loop: if space key clicked, loop: if mouse clicked, set X2 to mouse x - X, if mouse y > Y, draw black key, if white key, draw white key, set C to C + 1, message 1.
- Define: White Key Out:** Loop: if C < 3, set C to C - 1, if C < 7, set C to C - 1, if C < 10, set C to C - 1.
- Define: Black Key Out:** Loop: if C < 2, set C to C - 1, if C < 10, set C to C - 1.
- When Message Received:** Play sound C for 0.5 seconds.

Stage View:

- Stage:** WW 35, WH 235, BW 24, BH 143, C 67.
- Stage Dimensions:** X -140, Y -25.5, X2 165, Y2 117.5, Shift 0.
- Sprite:** Sprite 1, size 40, direction 90.
- Background:** Background 1.

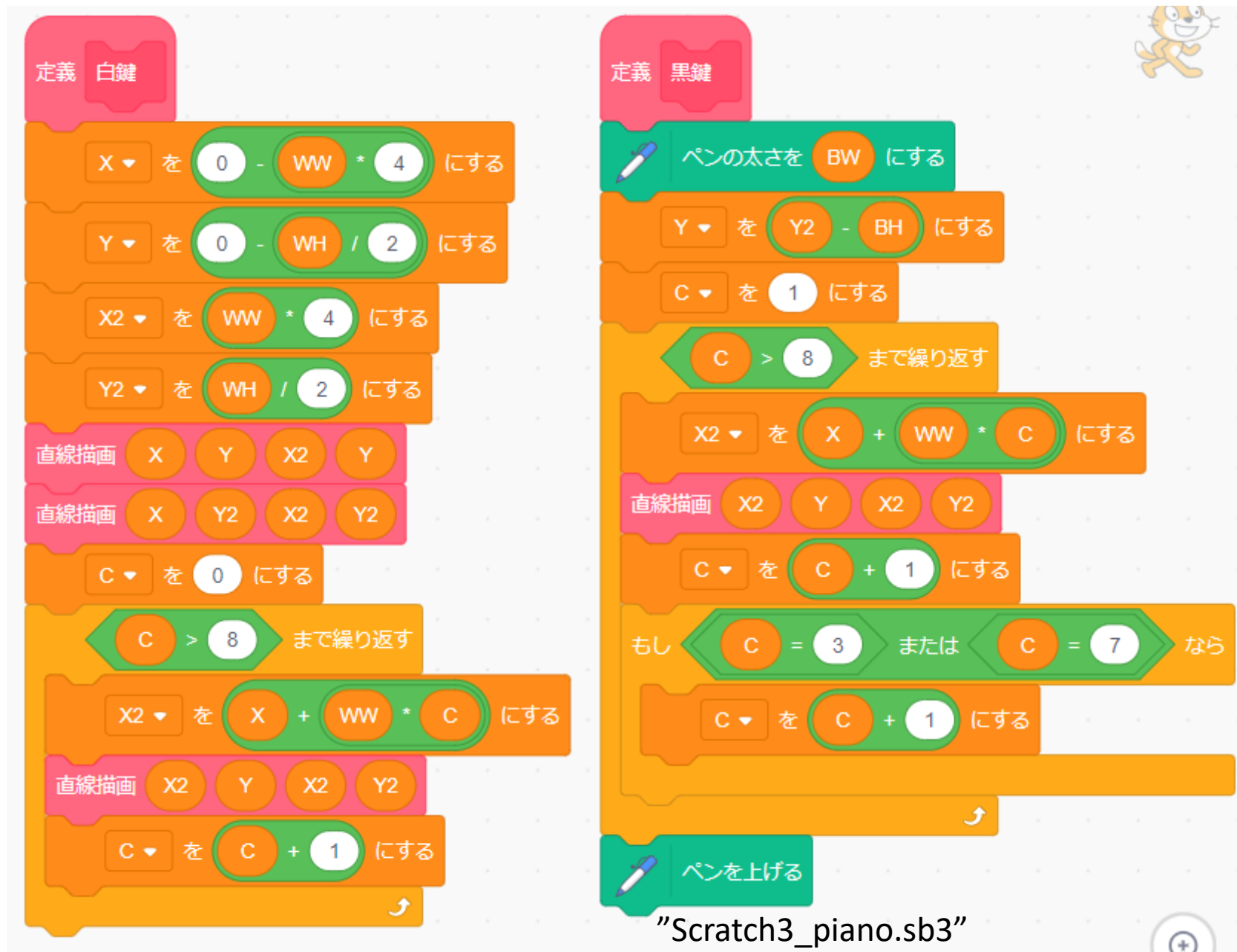
10. 拡張機能「ペン」と「音楽」の使用例:ピアノ鍵盤(2):プログラム部拡大(1)

「移調」設定によりキーを半音単位で上下させることができます。



"Scratch3_piano.sb3"

10. 拡張機能「ペン」と「音楽」の使用例:ピアノ鍵盤(3):プログラム部拡大(2)

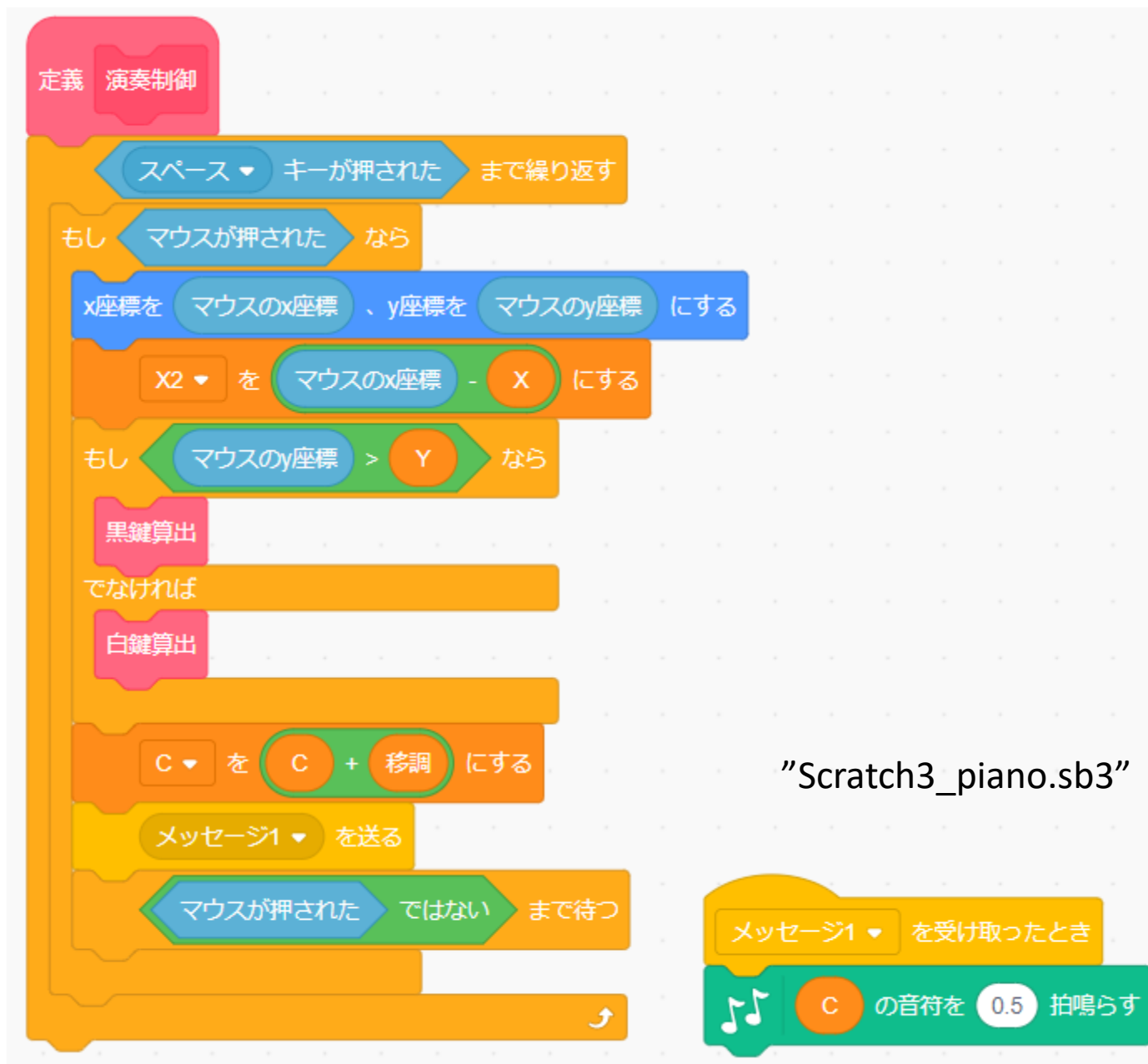


The image displays two Scratch code blocks for drawing a piano keyboard. The left block, titled '定義 白鍵' (Define White Key), sets up the initial coordinates and width for white keys. It defines X as $0 - WW * 4$, Y as $0 - WH / 2$, X2 as $WW * 4$, and Y2 as $WH / 2$. It then draws a vertical line from (X, Y) to (X2, Y2) and a horizontal line from (X, Y2) to (X2, Y2). A counter C is set to 0, and a loop 'C > 8' repeats the process of updating X2 to $X + WW * C$, drawing the lines, and incrementing C by 1.

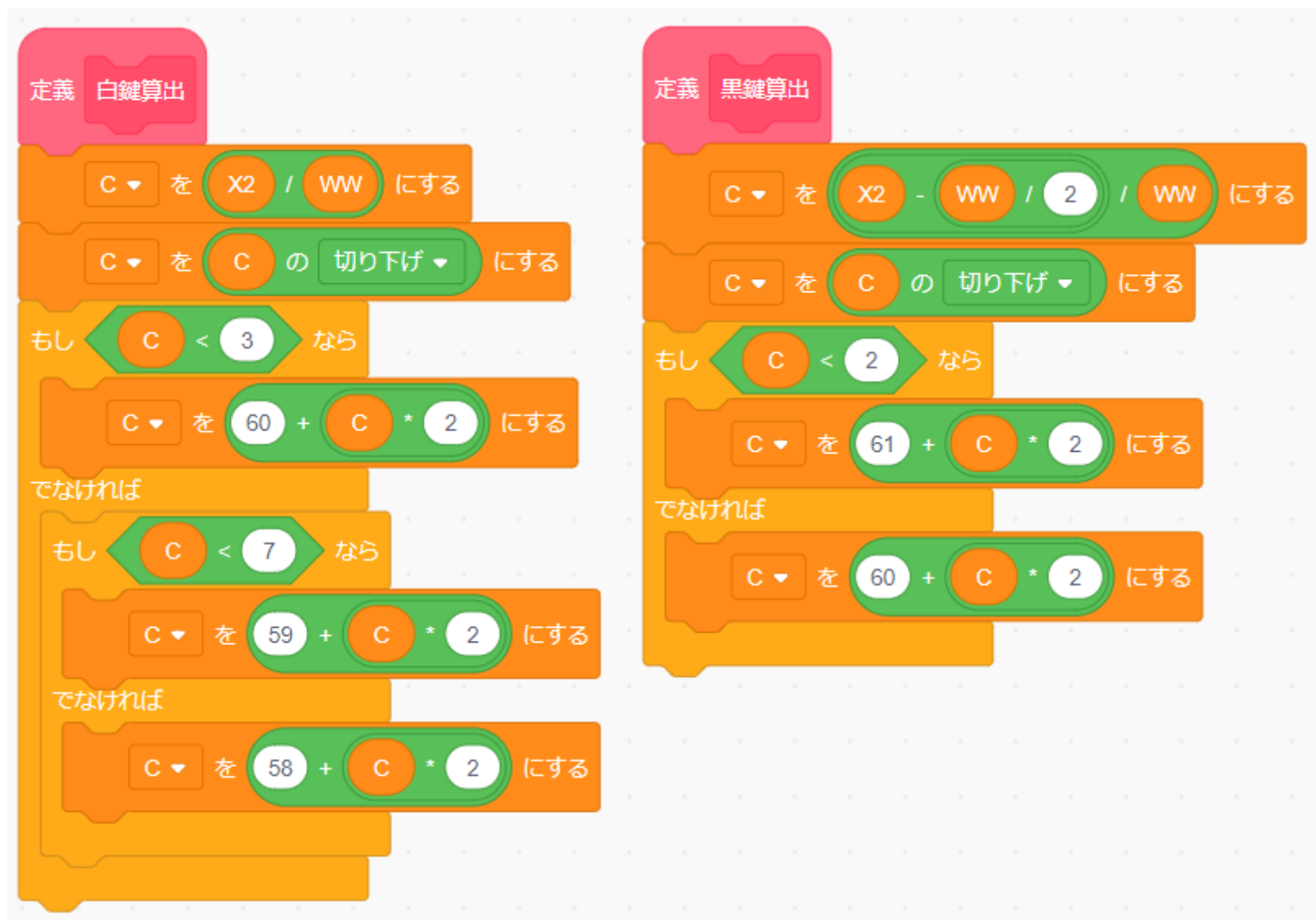
The right block, titled '定義 黒鍵' (Define Black Key), sets up the initial coordinates and width for black keys. It defines the pen thickness as BW, Y as $Y2 - BH$, and C as 1. A loop 'C > 8' repeats the process of updating X2 to $X + WW * C$, drawing a vertical line from (X2, Y) to (X2, Y2), and incrementing C by 1. A conditional block checks if C is 3 or 7, and if so, increments C by 1. The block ends with 'ペンを上げる' (Lift pen).

Scratch3_piano.sb3

10. 拡張機能「ペン」と「音楽」の使用例:ピアノ鍵盤(3):プログラム部拡大(3)



10. 拡張機能「ペン」と「音楽」の使用例:ピアノ鍵盤(3):プログラム部拡大(4)



10. 拡張機能「ペン」と「音楽」の使用例(2): 自動演奏ピアノ(1)

先に拡張機能「ペン」と「音楽」を用いて作成したピアノ鍵盤に対して、あらかじめ演奏する音符を定義して、鍵盤を自動的にクリックして自動演奏させることができる。

"Scratch3_apiano.sb3"

The image displays a Scratch3 project titled "Scratch3_apiano.sb3". The main stage shows a simplified piano keyboard with 12 keys. The keys are represented by black rectangles on a white background. The keys are labeled with their corresponding notes: C, D, E, F, G, A, B, C, D, E, F, G. The keyboard is positioned on the right side of the stage.

On the left side, there are several custom blocks for the "ペン" (Pen) and "音楽" (Music) extensions. These blocks are used to define the notes and their durations for the automatic piano performance.

The "音楽データ" (Music Data) block is a table with 12 rows, each representing a note and its duration. The columns are labeled "演奏" (Performance) and "データ" (Data).

演奏	データ
演奏 60	0.5
演奏 61	0.25
演奏 62	0.25
演奏 63	0.25
演奏 64	0.25
演奏 65	0.25
演奏 66	0.25
演奏 67	0.25
演奏 68	0.25
演奏 69	0.25
演奏 70	0.25
演奏 71	0.25
演奏 72	0.5

The "音楽" (Music) block is a custom block that takes a note number (e.g., 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72) and a duration (e.g., 0.5, 0.25) as input. It is used to define the notes and their durations for the automatic piano performance.

The "ペン" (Pen) block is a custom block that takes a note number (e.g., 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72) and a duration (e.g., 0.5, 0.25) as input. It is used to define the notes and their durations for the automatic piano performance.

The "音楽" (Music) block is a custom block that takes a note number (e.g., 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72) and a duration (e.g., 0.5, 0.25) as input. It is used to define the notes and their durations for the automatic piano performance.

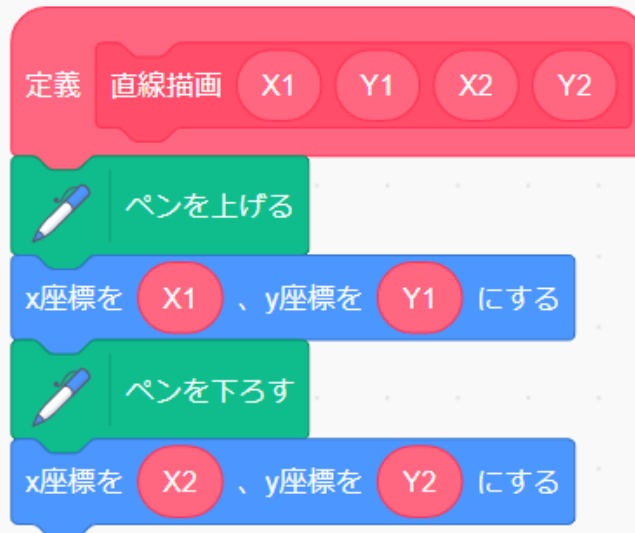
The "ペン" (Pen) block is a custom block that takes a note number (e.g., 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72) and a duration (e.g., 0.5, 0.25) as input. It is used to define the notes and their durations for the automatic piano performance.

10. 拡張機能「ペン」と「音楽」の使用例(2): 自動演奏ピアノ(2): プログラム部(1)

「移調」設定によりキーを半音単位で上下させることができます。



例題の半音階では面白みがないので、皆さんがご存じのメロディー(単旋律)を打ち込んでみてください。



"Scratch3_apiano.sb3"

10. 拡張機能「ペン」と「音楽」の使用例(3): 自動演奏ピアノ(3): プログラム部(2)

定義 白鍵

X を $0 - WW * 4$ にする

Y を $0 - WH / 2$ にする

X2 を $WW * 4$ にする

Y2 を $WH / 2$ にする

直線描画 X Y X2 Y

直線描画 X Y2 X2 Y2

C を 0 にする

もし $C > 8$ まで繰り返す

X2 を $X + WW * C$ にする

直線描画 X2 Y X2 Y2

C を $C + 1$ にする

定義 黒鍵

ペンの太さを BW にする

Y を $Y2 - BH$ にする

C を 1 にする

もし $C > 8$ まで繰り返す

X2 を $X + WW * C$ にする

直線描画 X2 Y X2 Y2

C を $C + 1$ にする

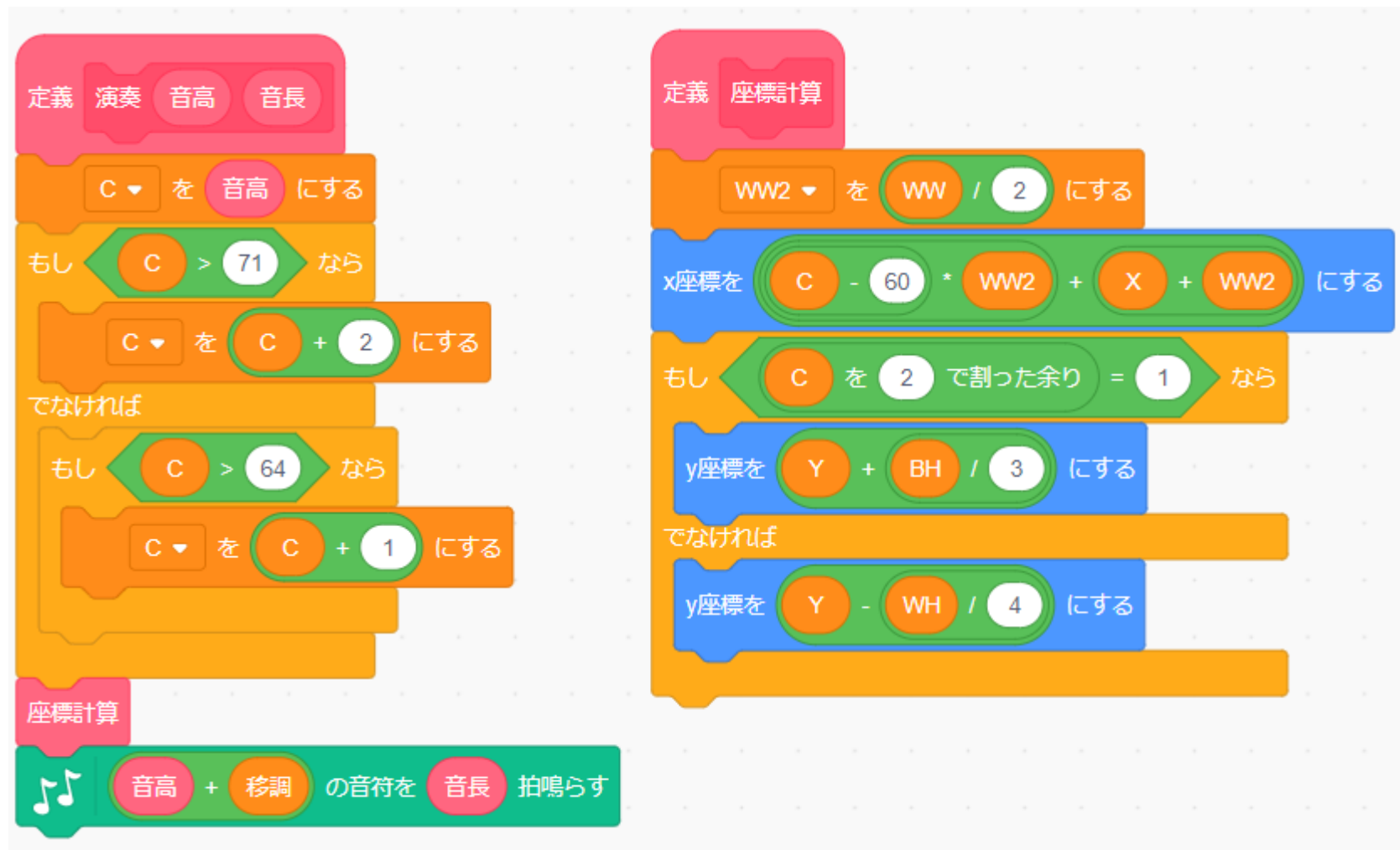
もし $C = 3$ または $C = 7$ なら

C を $C + 1$ にする

ペンを上げる

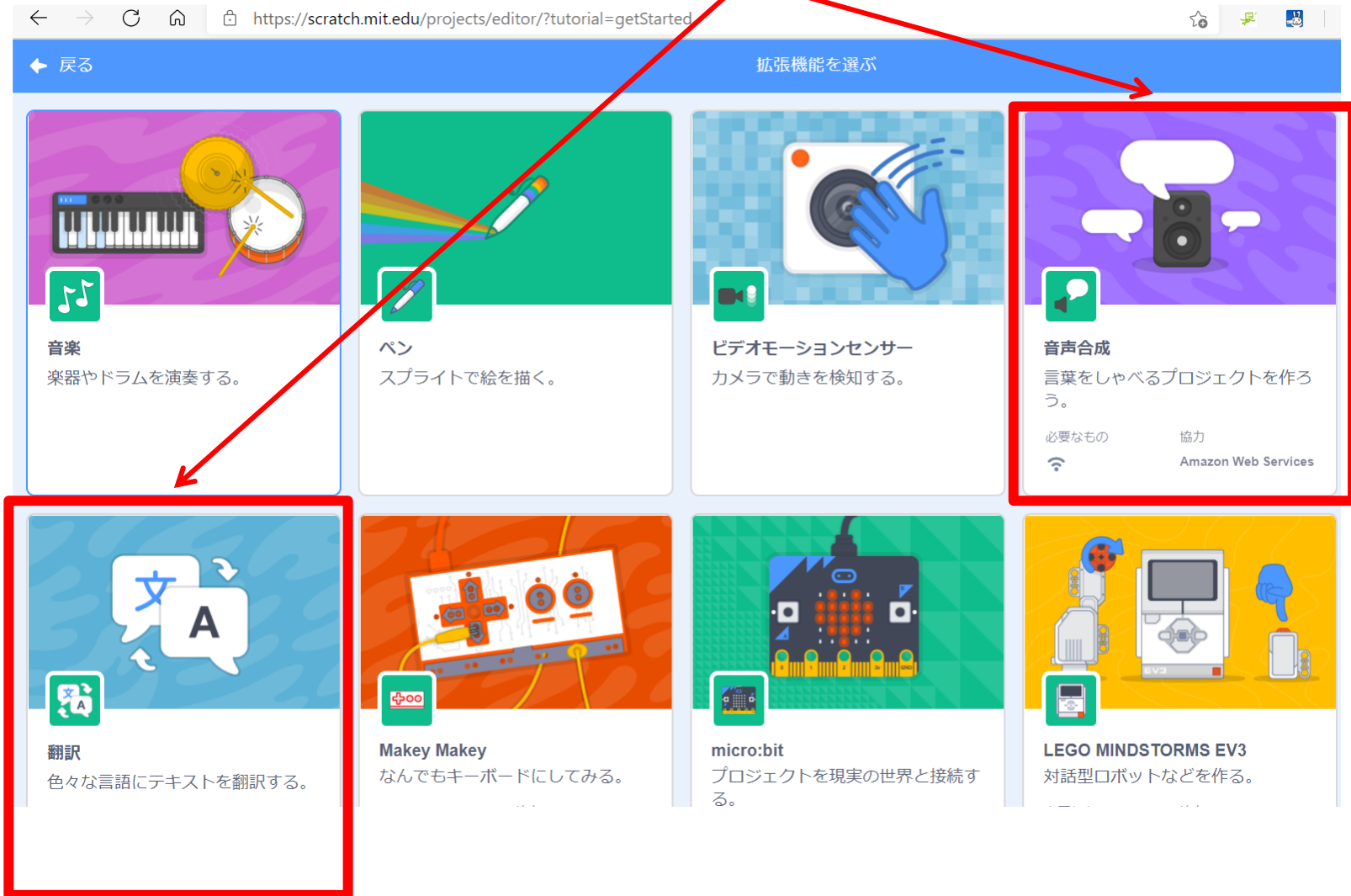
"Scratch3_apiano.sb3"

10. 拡張機能「ペン」と「音楽」の使用例(3): 自動演奏ピアノ(3): プログラム部(3)



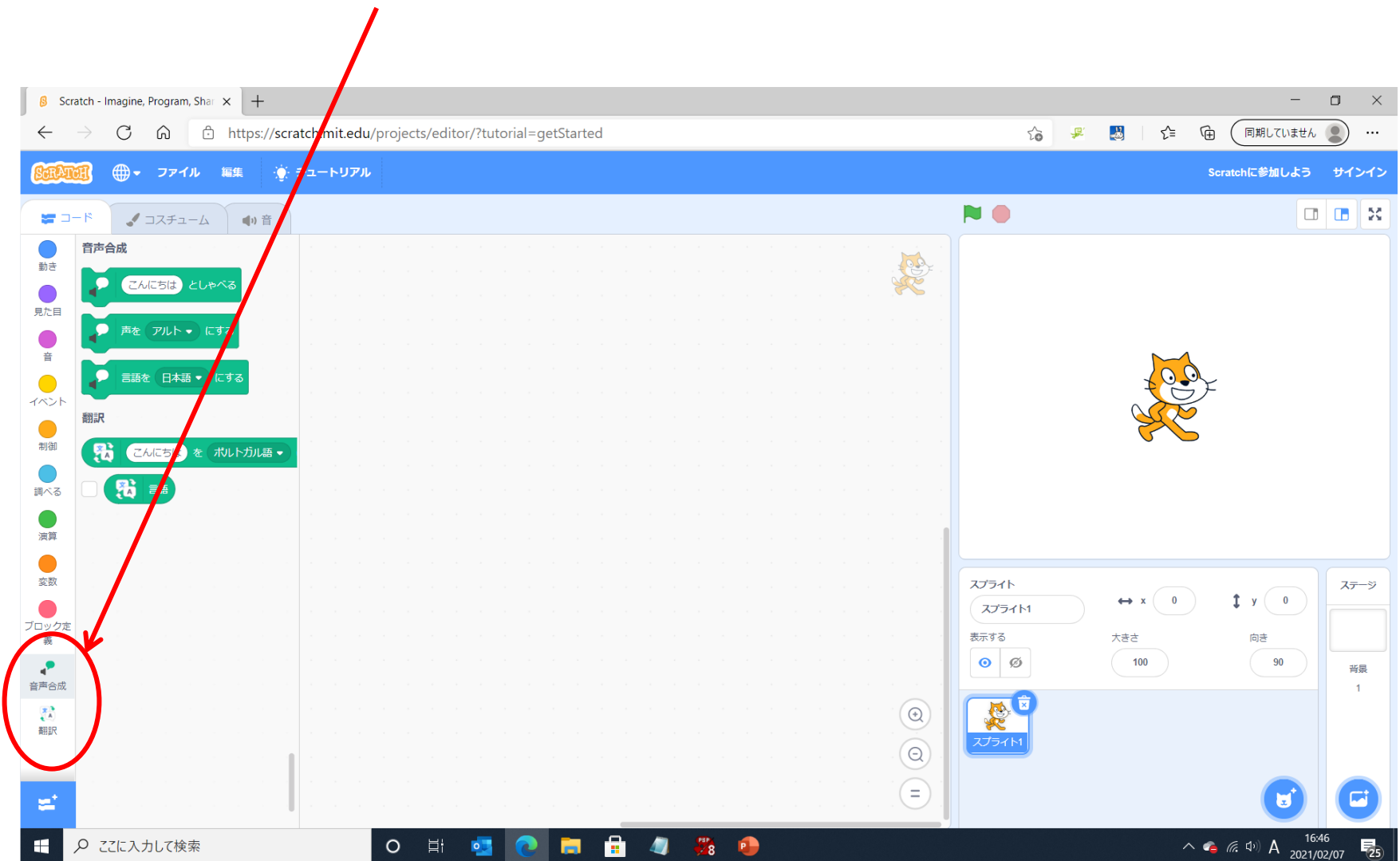
11. 拡張機能「音声合成・翻訳」の使用方法(1)

下記の拡張機能メニューで「音声合成」と「翻訳」を順次選択します。



11. 拡張機能「音声合成・翻訳」の使用方法(2)

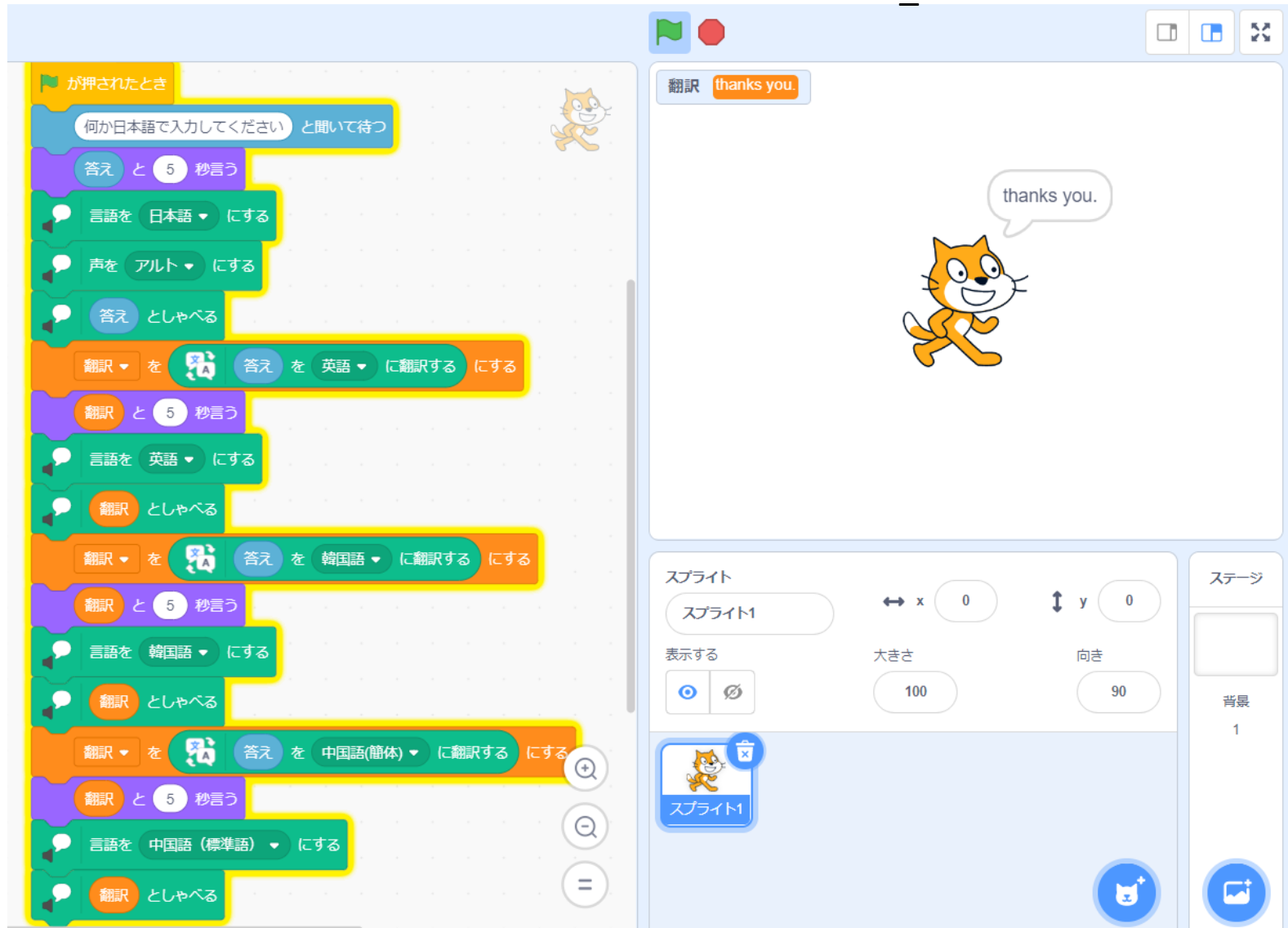
左端のメニューの最下部に拡張機能「音声合成」と「翻訳」が追加されています。



11. 拡張機能「音声合成・翻訳」の使用例：多言語翻訳機(1)

日本語でメッセージを入力すると、原語、英語、韓国語、中国語の順にキャラクターにしゃべらせ音声で読み上げる。

"Scratch3_trans.sb3"



11. 拡張機能「音声合成・翻訳」の使用例：多言語翻訳機(2):プログラム拡大

"Scratch3_trans.sb3"



入力文字列の言語および、翻訳先の外国語は、英語、韓国語、中国語に限らず多種類用意されていますので、種々の外国語に変更して試してください。

11. 拡張機能「音声合成・翻訳」の使用例: 多言語翻訳機(3): 実行部

"Scratch3_trans.sb3"

[文字列の入力画面(日本語)]



音声で読み上げられる
(現状、音声認識は不可)

[翻訳結果の出力画面(英語)]



翻訳文がテキスト表示され、
音声で読み上げられる