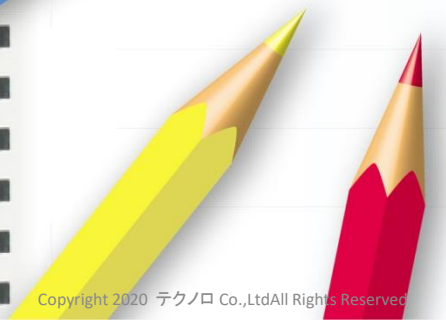
プログラミング教室のテクノロ

Pythonの道



本格的なゲーム開発②(後編)

もくじ

・パズルゲームをつくる（後編）

　複数のアルゴリズムを組み込む方法を学ぶ

タイトル画面とゲームオーバー画面

タイトル画面とゲームオーバー画面を作る処理を追加し、ゲームとしての一連の流れを完成させる。

indexという変数を用意し、その値が1のときにタイトル画面の処理
2~5の時にゲーム中の各処理、6の時にゲームオーバー画面の処理を行う

indexの値	処理の内容
0	タイトル画面の文字を表示し、index1の処理に移る。
1	ゲーム開始の入力を待つ。画面をクリックしたら、最初に落ちてくるネコをセットしてindex2の処理に移る。
2	【ゲーム中の処理1】 ネコを落下させる。全てのネコが落下したらindex3の処理に移る。
3	【ゲーム中の処理2】 ネコが3つ以上並んだかを調べ、そろったネコがあれば肉球に変える。index4の処理に移る。
4	【ゲーム中の処理3】 肉球に変わったマスがあれば、消してスコアを加算し、再びindex2の落下処理に移る。（消えたマスの上にあるネコを落とすため。）肉球に変わったマスがない場合、最上段まで積みあがっていなければindex5の入力待ちに移る。積み上がってしまったら、index6のゲームオーバに移る。
5	【ゲーム中の処理4】 プレイヤからの入力を待つ。マウスでカーソルを移動し、クリックしてネコを配置したらindex2のネコを落下させる処理に移る。
6	ゲームオーバ画面。変数で時間をカウントし、5秒後にindex0の処理に移る。

タイトル画面とゲームオーバー画面

ゲームオーバにならない間は、index2からindex5の処理が繰り返される。処理をわけるには、リアルタイム処理を行うgame_main()関数の中で以下のように記述する。

```
if index == 0:
    処理0
elif index == 1:
    処理1
elif index == 2:
    処理2
.
.
```

追加する関数

関数	意味
sweep_neko()	肉球を消す関数。いくつか消したかを数え、戻り値として返す。 この戻り値でスコアを計算する。
over_neko()	最上段まで積み上がったかを調べる。積み上がった場合はTrueを返す。
set_neko()	最上段にランダムにネコを配置する。
draw_txt(txt,x,y,siz,col,tg)	影付きの文字列を表示する関数。引数は文字列、xy座標、文字の大きさ、色、タグ。

タイトル画面とゲームオーバー画面

```
import tkinter  ・・・・tkinterモジュールの呼出し
import random   ・・・・randomモジュールの呼出し

index = 0       ・・・・ゲーム進行を管理する変数
timer = 0       ・・・・時間を管理する変数
score = 0       ・・・・スコア用の変数
tugi = 0        ・・・・次にセットするネコの値をいれる変数

cursor_x = 0    ・・・・カーソルの横方向の位置(左から何マス目にあるか)
cursor_y = 0    ・・・・カーソルの縦方向の位置(上から何マス目にあるか)
mouse_x = 0     ・・・・マウスポインタのX座標
mouse_y = 0     ・・・・マウスポインタのY座標
mouse_c = 0     ・・・・マウスボタンをクリックした時の変数（フラグ）

def mouse_move(e):  ・・・・マウスを動かした時に実行する関数
    global mouse_x , mouse_y ・・・・これらをグローバル変数として扱うと宣言
    mouse_x = e.x  ・・・・mouse_xにマウスポインタのX座標を代入
    mouse_y = e.y  ・・・・mouse_yにマウスポインタのY座標を代入

def mouse_press(e):  ・・・・マウスボタンをクリックした時に実行する関数
    global mouse_c   ・・・・この変数をグローバル変数として扱うと宣言
    mouse_c = 1      ・・・・mouse_cに1を代入

neko=[] ・・・・マス目を管理する二次元リスト
check=[] ・・・・判定用の二次元リスト
```

次のページに続く

タイトル画面とゲームオーバー画面

```
for i in range(10):・・・・繰り返しとappend()命令でリストを初期化する
    neko.append([0,0,0,0,0,0,0,0])
    check.append([0,0,0,0,0,0,0,0])

def draw_neko():  ・・・・ネコを表示する関数
    cvs.delete("NEKO")・・・・一旦、ネコを消す
    for y in range(10):  ・・・繰り返し yは0から9まで１ずつ増える
        for x in range(8):  ・・・繰り返し xは0から7まで１ずつ増える
            cvs.create_image(x*72+60,y*72+60,image=img_neko[neko[y][x]],tag="NEKO")
                                        ・・・二次元リストから条件にあった画像を表示する

def check_neko():  ・・・ネコが縦、横、斜めに３つ以上並んだか調べる関数
    for y in range(10):  ・・・繰り返し yは0から9まで１ずつ増える
        for x in range(8):・・・繰り返し xは0から7まで１ずつ増える
            check[y][x]=neko[y][x]・・・判定用のリストにネコの値を入れる

    for y in range(1,9):・・・繰り返し yは1から8まで１ずつ増える
        for x in range(8):・・・繰り返し xは0から7まで１ずつ増える
            if check[y][x]>0:・・空白マスどうしは同じ値が入っていると認識させない
                if check[y-1][x] == check[y][x] and check[y+1][x] == check[y][x]:
                    neko[y-1][x]=7  ・・・マスにネコが配置されていて上下が
                    neko[y][x]=7    同じネコならそれらのマスを肉球に変える
                    neko[y+1][x]=7
```

次のページに続く

タイトル画面とゲームオーバー画面

```
    for y in range(10):  ・・・繰り返し yは0から9まで１ずつ増える
        for x in range(1,7):  ・・・繰り返し xは1から6まで１ずつ増える
            if check[y][x]>0:・・・・空白マスどうしは同じ値が入っていると認識させない
                if check[y][x-1] == check[y][x] and check[y][x+1] == check[y][x]:
                    neko[y][x-1]=7  ・・・マスにネコが配置されていて左右が同じネコなら
                    neko[y][x]=7    それらのマスを肉球に変える
                    neko[y][x+1]=7

    for y in range(1,9):・・・繰り返し yは1から8まで１ずつ増える
        for x in range(1,7):  ・・・繰り返し xは1から6まで１ずつ増える
            if check[y][x]>0:・・・・空白マスどうしは同じ値が入っていると認識させない
                if check[y-1][x-1] == check[y][x] and check[y+1][x+1] == check[y][x]:
                    neko[y-1][x-1]=7     ・・・マスにネコが配置されていて左上と右下が
                    neko[y][x]=7         同じネコならそれらのマスを肉球に変える
                    neko[y+1][x+1]=7
                if check[y+1][x-1] == check[y][x] and check[y-1][x+1] == check[y][x]:
                    neko[y+1][x-1]=7    ・・・マスにネコが配置されていて左下と右上が同
                    neko[y][x]=7        じネコならそれらのマスを肉球に変える
                    neko[y-1][x+1]=7
def sweep_neko():・・・そろったネコ(肉球)を消す関数
    num = 0       ・・・消した数をカウントする変数
    for y in range(10):  ・・・繰り返し yは0から9まで１ずつ増える
        for x in range(8):・・・繰り返し xは0から7まで１ずつ増える
            if neko[y][x] == 7:  ・・・マスが肉球になっていれば
                neko[y][x] = 0  肉球を消し、消した数を1増やす
                num = num + 1
    return num  ・・・消した数を戻り値として返す
```

次のページに続く

タイトル画面とゲームオーバー画面

```
def drop_neko():  ・・・ネコを落下させる関数
    flg = False    ・・・落下したかを判断するフラグ（Falseは落下していない）
    for y in range(8,-1,-1): ・・・繰り返し yは8から0まで１ずつ減る
        for x in range(8):  ・・・繰り返し xは0から7まで１ずつ増える
            if neko[y][x] != 0 and neko[y+1][x] == 0:  ・・・ネコのあるマス
                neko[y+1][x] = neko[y][x]                の下が空白なら空白に
                neko[y][x] = 0                           ネコを入れ元のマスは
                flg = True ・・・落下したフラグを立てる     空白にする
    return flg ・・・フラグの値を戻り値として返す

def over_neko(): ・・・最上段に達したか調べる関数
    for x in range(8):・・・繰り返し xは0から7まで１ずつ増える
        if neko[0][x] > 0:    ・・・最上段にネコがあるなら
            return True       Trueを返す
    return False   ・・・最上段に達してないならFalseを返す

def set_neko():・・・最上段にネコをセットする関数
    for x in range(8):・・・繰り返し xは0から7まで１ずつ増える
        neko[0][x] = random.randint(0,6)・・・最上段にランダムにネコをセットする

def draw_txt(txt,x,y,siz,col,tg):・・・影付きの文字列を表示する関数
    fnt = ("Times New Roman",siz,"bold") ・・・フォントを指定
    cvs.create_text(x+2,y+2,text=txt,fill="black",font=fnt,tag=tg)
                                          ・・2ドットずらし黒い色で文字列を表示（影）
    cvs.create_text(x,y,text=txt,fill=col,font=fnt,tag=tg)
                                          ・・指定した色で文字列を表示
```

次のページに続く

タイトル画面とゲームオーバー画面

```
def game_main():・・・メインの処理（リアルタイム処理）を行う関数
    global index,timer,score,tugi  ・・・これらをグローバル変数として扱うと宣言
    global cursor_x,cursor_y,mouse_c ・・・これらをグローバル変数として扱うと宣言
    if index == 0:・・・index0の処理
        draw_txt("ねこねこ",312,240,100,"violet","TITLE")・・・タイトルロゴの表示
        draw_txt("クリックしてスタート",212,560,30,"orange","TITLE")
        index = 1  ・・indexの値を1にする          ・・・クリックしてスタートと表示
        mouse_c = 0 ・・クリックしたフラグを解除する
    elif index == 1:・・・index1の処理
        if mouse_c == 1:・・・マウスボタンをクリックしたら
            for y in range(10):・・・二重ループの
                for x in range(8):      繰り返しで
                    neko[y][x]=0          マスをクリア
            mouse_c = 0 ・・・クリックしたフラグを解除
            score = 0     ・・・スコアを0にする
            tugi = 0      ・・・次に配置するネコを一旦無し(値0)にする
            cursor_x = 0 ・・・カーソルの位置を左上にする
            cursor_y = 0 ・・・カーソルの位置を左上にする
            set_neko()    ・・・最上段にネコをセットする
            draw_neko()   ・・・ネコを表示する
            cvs.delete("TITLE")・・・TITLE画面の文字を消す
            index = 2 ・・indexの値を2にする
```

次のページに続く

タイトル画面とゲームオーバー画面

```
elif index == 2:・・・index2の処理
    if drop_neko() == False:・・・ネコを落下させ、落ちたネコが無いなら
        index = 3                    indexの値を3にする
    draw_neko()・・・ネコを表示する

elif index == 3:・・・index3の処理
    check_neko()  ・・・同じネコが並んだか調べる
    draw_neko()   ・・・ネコを表示
    index = 4     ・・・indexの値を4にする

elif index == 4:・・・index4の処理
    sc = sweep_neko()・・・肉球を消し、消した数をscに入れる
    score = score + sc*10・・・スコアを加算する
    if sc > 0:  ・・・消した肉球(ネコ)があれば
        index = 2   index2の処理に移る(再び落下)
    else:                          ・・・そうでなければ
        if over_neko() == False:       最上段に達していなければ
            tugi = random.randint(1,6)   次に配置するネコをランダムに決め
            index = 5                      indexの値を5にする
        else:       ・・・そうでなければ（最上段に達した）
            index = 6  indexの値を6にする
            timer = 0  timerの値を0にする
    draw_neko()・・・ネコを表示する
```

次のページに続く

タイトル画面とゲームオーバー画面

```
    elif index == 5:・・・index5の処理          ・・マウスポインタの座標が盤面であれば
        if 24 <= mouse_x and mouse_x < 24+72*8 and 24 <= mouse_y and mouse_y < 24+72*10:
            cursor_x = int((mouse_x-24)/72)・ポインタのx座標からカーソルの横の位置を計算
            cursor_y = int((mouse_y-24)/72)・ポインタのy座標からカーソルの縦の位置を計算
            if mouse_c ==1:・・・マウスボタンをクリックしたら
                mouse_c = 0・・・クリックしたフラグを解除
                set_neko()   ・・・最上段にネコをセット
                neko[cursor_y][cursor_x] = tugi ・・・カーソルのマスにネコを配置
                tugi = 0・・・次に配置するネコ(風船内)をなしに
                index = 2 ・・・indexの値を2にする
        cvs.delete("CURSOR")・・・カーソルを消す
        cvs.create_image(cursor_x*72+60,cursor_y*72+60,image = cursor,tag = "CURSOR")
        draw_neko()・・ネコを表示する          ・・・新たな位置にカーソルを表示する
    elif index == 6:・・・index6の処理
        timer = timer + 1・・・timerの値を1増やす
        if timer == 1:・・・timerの値が1なら
            draw_txt("GAME OVER",312,348,60,"red","OVER")
        if timer == 50:・・・timerの値が50なら      ・・GAME OVERの文字を表示する
            cvs.delete("OVER")・・・GAME OVERの文字を消し
            index = 0・・・indexの値を0にする
    cvs.delete("INFO") ・・・一旦、スコアの表示を消す
    draw_txt("SCORE" + str(score),160,60,32,"blue","INFO")・・・スコアの表示する
    if tugi > 0: ・・・次に配置するネコの値がセットされていればそのネコを表示する
        cvs.create_image(752,128,image = img_neko[tugi],tag="INFO")
    root.after(100,game_main)・・0.1秒後に再びメインの処理を実行する 次のページに続く
```

タイトル画面とゲームオーバー画面

```
root=tkinter.Tk()・・・・ウィンドウのオブジェクトを作る
root.title("落ち物パズル「ねこねこ」")・・・・タイトルを指定
root.resizable(False,False)・・・・ウィンドウサイズを変更できないようにする
root.bind("<Motion>",mouse_move) ・・マウスが動いた時に実行する関数を指定
root.bind("<ButtonPress>",mouse_press)・マウスボタンをクリックした時に実行する関数を指定
cvs=tkinter.Canvas(root,width=912,height=768)・・・キャンバスの部品を作る
cvs.pack()・・キャンバスを配置する

bg = tkinter.PhotoImage(file=neko_bg.png")・・・背景画像の読み込み
cursor = tkinter.PhotoImage(file=neko_cursor.png")・・カーソル画像の読み込み
img_neko=[・・・リストで複数のネコの画像を管理
    None,・・・何もない値を意味する
    tkinter.PhotoImage(file=neko1.png"),
    tkinter.PhotoImage(file=neko2.png"),
    tkinter.PhotoImage(file=neko3.png"),
    tkinter.PhotoImage(file=neko4.png"),
    tkinter.PhotoImage(file=neko5.png"),
    tkinter.PhotoImage(file=neko6.png"),
    tkinter.PhotoImage(file=neko_niku.png")
]

cvs.create_image(456,384,image=bg)・・・キャンバス上に背景を描く
game_main() ・・・メインの処理を行う関数を呼び出す
root.mainloop()・・・ウィンドウを表示する
```

マス上のデータを管理する

「Run Module」またはF5キーを押してプログラムを実行する。

ゲームとして完成させる

一通り動くようになったゲームに難易度選択とハイスコアの処理を入れ、落ち物パズルを完成させる。

先程作成したプログラムは6種類のネコ（ブロック）がランダムに落ちてくる。ブロックの色や柄をそろえるゲームはブロックの種類が多いほど難しくなる。

そこでネコの種類を「かんたん」モードでは４つ、「ふつう」モードでは５つ、「むずかしい」モードでは６つとする。さらにハイスコアを保持する変数を用意し、現在のスコアがハイスコアを上回った時にハイスコアを更新する処理を追加してみよう。

ゲームとして完成させる

```
import tkinter
import random

index = 0
timer = 0
score = 0
hisc = 1000 ・・・・ハイスコアを保持する変数
difficulty = 0 ・・・・難易度の値を入れる変数
tugi = 0

cursor_x = 0
cursor_y = 0
mouse_x = 0
mouse_y = 0
mouse_c = 0

def mouse_move(e):
    global mouse_x , mouse_y
    mouse_x = e.x
    mouse_y = e.y

def mouse_press(e):
    global mouse_c
    mouse_c = 1
```

次のページに続く

ゲームとして完成させる

```
neko=[]
check=[]
for i in range(10):
    neko.append([0,0,0,0,0,0,0,0])
    check.append([0,0,0,0,0,0,0,0])

def draw_neko():
    cvs.delete("NEKO")
    for y in range(10):
        for x in range(8):
            cvs.create_image(x*72+60,y*72+60,image=img_neko[neko[y][x]],tag="NEKO")

def check_neko():
    for y in range(10):
        for x in range(8):
            check[y][x]=neko[y][x]

    for y in range(1,9):
        for x in range(8):
            if check[y][x]>0:
                if check[y-1][x] == check[y][x] and check[y+1][x] == check[y][x]:
                    neko[y-1][x]=7
                    neko[y][x]=7
                    neko[y+1][x]=7
```

次のページに続く

ゲームとして完成させる

```
for y in range(10):
    for x in range(1,7):
        if check[y][x]>0:
            if check[y][x-1] == check[y][x] and check[y][x+1] == check[y][x]:
                neko[y][x-1]=7
                neko[y][x]=7
                neko[y][x+1]=7

for y in range(1,9):
    for x in range(1,7):
        if check[y][x]>0:
            if check[y-1][x-1] == check[y][x] and check[y+1][x+1] == check[y][x]:
                neko[y-1][x-1]=7
                neko[y][x]=7
                neko[y+1][x+1]=7
          if check[y+1][x-1] == check[y][x] and check[y-1][x+1] == check[y][x]:
                neko[y+1][x-1]=7
                neko[y][x]=7
                neko[y-1][x+1]=7
```

次のページに続く

ゲームとして完成させる

```
def sweep_neko():
    num = 0
    for y in range(10):
        for x in range(8):
            if neko[y][x] == 7:
                neko[y][x] = 0
                num = num + 1
    return num

def drop_neko():
    flg = False
    for y in range(8,-1,-1):
        for x in range(8):
            if neko[y][x] != 0 and neko[y+1][x] == 0:
                neko[y+1][x] = neko[y][x]
                neko[y][x] = 0
                flg = True
    return flg

def over_neko():
    for x in range(8):
        if neko[0][x] > 0:
            return True
    return False
```

次のページに続く

ゲームとして完成させる

```
def set_neko():
    for x in range(8):
        neko[0][x] = random.randint(0,difficulty)
                                        ・・・・難易度に応じてランダムにネコをセットする
def draw_txt(txt,x,y,siz,col,tg):
    fnt = ("Times New Roman",siz,"bold")
    cvs.create_text(x+2,y+2,text=txt,fill="black",font=fnt,tag=tg)
    cvs.create_text(x,y,text=txt,fill=col,font=fnt,tag=tg)

def game_main():
    global index,timer,score,hisc,difficulty,tugi ・・グローバル変数として扱うと宣言
    global cursor_x,cursor_y,mouse_c
    if index == 0:
        draw_txt("ねこねこ",312,240,100,"violet","TITLE")
        cvs.create_rectangle(168,384,456,456,fill="skyblue",width=0,tag="TITLE")
        draw_txt("かんたん",312,420,40,"white","TITLE")  ・・かんたんの文字を表示
        cvs.create_rectangle(168,528,456,600,fill="lightgreen",width=0,tag="TITLE")
        draw_txt("ふつう",312,564,40,"white","TITLE") ・・ふつうの文字を表示
        cvs.create_rectangle(168,672,456,744,fill="orange",width=0,tag="TITLE")
        draw_txt("むずかしい",312,708,40,"white","TITLE")
                                        ・・むずかしいの文字を表示

        index = 1
        mouse_c = 0
```

次のページに続く

ゲームとして完成させる

```
    elif index == 1:
        difficulty = 0 ・・difficultyの値に0を代入する
        if mouse_c == 1:・・マウスボタンをクリックする
            if 168 < mouse_x and mouse_x < 456 and 384 < mouse_y and mouse_y < 456:
                difficulty = 4・クリックされた場所が「かんたん」の場合、difficultyに4を代入
            if 168 < mouse_x and mouse_x < 456 and 528 < mouse_y and mouse_y < 600:
                difficulty = 5・クリックされた場所が「ふつう」の場合、difficultyに5を代入
            if 168 < mouse_x and mouse_x < 456 and 672 < mouse_y and mouse_y < 744:
                difficulty = 6・クリックされた場所が「むずかしい」の場合、difficultyに6を代入
        if difficulty > 0:・・difficultyの値がセットされ場合、以下の処理を行う
            for y in range(10):
                for x in range(8):
                    neko[y][x]=0
            mouse_c = 0
            score = 0
            tugi = 0
            cursor_x = 0
            cursor_y = 0
            set_neko()
            draw_neko()
            cvs.delete("TITLE")
            index = 2
```

次のページに続く

ゲームとして完成させる

```
    elif index == 2:
        if drop_neko() == False:
            index = 3
        draw_neko()

    elif index == 3:
        check_neko()
        draw_neko()
        index = 4
    elif index == 4:
        sc = sweep_neko()
        score = score + sc*difficulty*2 ・・スコアを加算する
        if score > hisc: ・・スコアがハイスコアを超えたら、
            hisc = score    ハイスコアを更新する
        if sc > 0:
            index = 2
        else:
            if over_neko() == False:
                tugi = random.randint(1,difficulty) ・・次に配置するネコをランダムにきめる
                index = 5
            else:
                index = 6
                timer = 0
        draw_neko()
```

次のページに続く

ゲームとして完成させる

```
elif index == 5:
    if 24 <= mouse_x and mouse_x < 24+72*8 and 24 <= mouse_y and mouse_y < 24+72*10:
        cursor_x = int((mouse_x-24)/72)
        cursor_y = int((mouse_y-24)/72)
        if mouse_c ==1:
            mouse_c = 0
            set_neko()
            neko[cursor_y][cursor_x] = tugi
            tugi = 0
            index = 2
    cvs.delete("CURSOR")
    cvs.create_image(cursor_x*72+60,cursor_y*72+60,image = cursor,tag = "CURSOR")
    draw_neko()
elif index == 6:
    timer = timer + 1
    if timer == 1:
        draw_txt("GAME OVER",312,348,60,"red","OVER")
    if timer == 50:
        cvs.delete("OVER")
        index = 0
cvs.delete("INFO")
draw_txt("SCORE" + str(score),160,60,32,"blue","INFO")
draw_txt("HISC" +str(hisc),450,60,32,"yellow","INFO")
```

・・ハイスコアを表示

次のページに続く

ゲームとして完成させる

```
    if tugi > 0:
        cvs.create_image(752,128,image = img_neko[tugi],tag="INFO")
    root.after(100,game_main)

root=tkinter.Tk()
root.title("落ち物パズル「ねこねこ」")
root.resizable(False,False)
root.bind("<Motion>",mouse_move)
root.bind("<ButtonPress>",mouse_press)
cvs=tkinter.Canvas(root,width=912,height=768)
cvs.pack()

bg = tkinter.PhotoImage(file="neko_bg.png")
cursor = tkinter.PhotoImage(file="neko_cursor.png")
img_neko=[
    None,
    tkinter.PhotoImage(file="neko1.png"),
    tkinter.PhotoImage(file="neko2.png"),
    tkinter.PhotoImage(file="neko3.png"),
    tkinter.PhotoImage(file="neko4.png"),
    tkinter.PhotoImage(file="neko5.png"),
    tkinter.PhotoImage(file="neko6.png"),
    tkinter.PhotoImage(file="neko_niku.png")
]
```

次のページに続く

ゲームとして完成させる

```
cvs.create_image(456,384,image=bg)
game_main()
root.mainloop()
```

ゲームとして完成させる

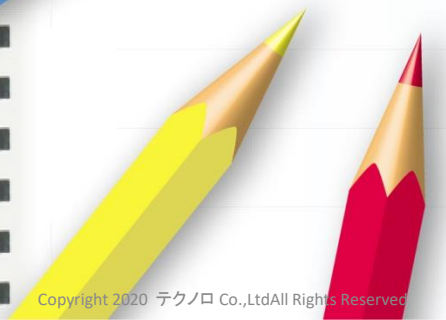
「Run Module」またはF5キーを押してプログラムを実行する。

復習＆チャレンジ

ここまで習ったことをScratchでもできるかチャレンジしてみよう。
その過程でScratchでできること、Pythonでないとできないことを整理してみよう。

メモ

プログラミング教室のテクノロ



なまえ：