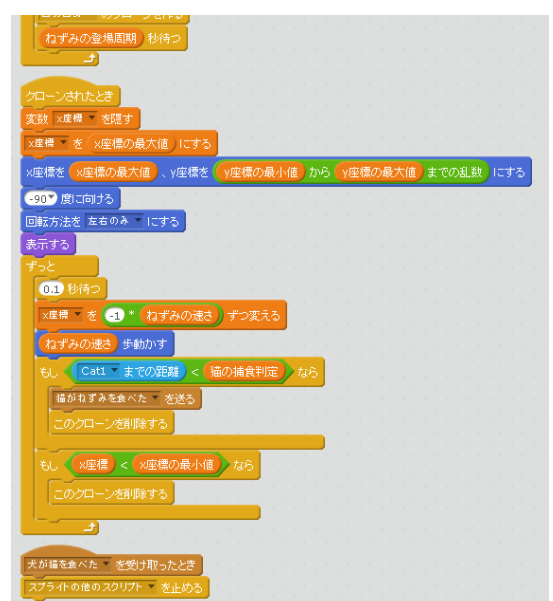


2019年度 理数系教員指導力向上研修

先生のためのScratch入門講座 —プログラミング教育の必修化に向けて—



(画像はプログラミングの例です。この講座で作成するプログラムとは異なります)

東京学芸大学 気象学研究室

佐藤尚毅

はじめに

小学校で 2020 年度からの実施が見込まれる新しい教育課程においては、プログラミングを必修とする方向で議論が進められています。「小学校段階におけるプログラミング教育の在り方について（議論の取りまとめ）」（小学校段階における論理的思考力や創造性、問題解決能力等の育成とプログラミング教育に関する有識者会議）には以下のように述べられています。

小学校におけるプログラミング教育が目指すのは、前述のように、子供たちが、コンピュータに意図した処理を行うよう指示することができるということを体験しながら、身近な生活でコンピュータが活用されていることや、問題の解決には必要な手順があることに気付くこと、各教科等で育まれる思考力を基盤としながら基礎的な「プログラミング的思考」を身に付けること、コンピュータの働きを自分の生活に生かそうとする態度を身に付けることである。

プログラミング教育の実施に当たっては、コーディングを覚えることが目的ではないことを明確に共有していくことが不可欠である。また、「主体的・対話的で深い学び」の実現に資するプログラミング教育とすることが重要であり、一人で黙々とコンピュータに向かっているだけで授業が終わったり、子供自身の生活や体験と切り離された抽象的な内容に終始したりすることがないように、留意が必要である。楽しく学んでコンピュータに触れることが好きになることが重要であるが、一方で、楽しいだけで終わっては学校教育としての学習成果に結びついたとは言えず、子供たちの感性や学習意欲に働きかけるためにも不十分である。

（文部科学省のウェブサイトより。下線は著者が付した。）

簡潔にまとめれば、

- ・児童が自分で考えてプログラムを作る。
- ・ただしコードは書かない。

という 2 つの条件を満たしたプログラミング教育を行なってくださいということです。この講座で扱う Scratch は、コードをキーボードでタイプするのではなく、画面上でブロックを組み合わせることによってプログラムを作成する教材であり、まさに、上記の目的に合ったものです。Scratch 以外にも類似した機能を持つ教材はありますが、基本的な考え方は同じです。今後、小学校でのプログラミング教育では、ブロックを組み合わせるという方法が標準になっていくと思います。この講座で学んだ内容は、利用する具体的な教材を問わず、小学校でのプログラミング教育において、普遍的に役立つものです。この講座は、初心者を前提にして、限られた時間の中で開講しているため、プログラムの例をあらかじめ示していますが、各自のアイデアで自分だけのプログラムを作成していくとさらに楽しく学べると思います。

第 1 部：アプリケーションのインストール

アプリケーションのダウンロード

まず、

<https://scratch.mit.edu/scratch2download/>

から Scratch 2 をダウンロードします。次に、

<https://get.adobe.com/jp/air/>

から Adobe AIR をダウンロードします(上記の Scratch のダウンロードサイトからリンクあり)。

※この講座では、CD-ROM または USB メモリで配布します。

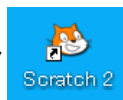
⇒ 配布したデータの中の「アプリケーション」というフォルダを開いてください。Windows の場合は「Windows」というフォルダに、macOS の場合は「Mac」というフォルダに移動してください。

アプリケーションのインストール

まず、Adobe AIR をインストールします。これは Scratch を動作させるために必要なアプリケーションです。インストールするためには、ダウンロードした AdobeAIRInstaller.exe (AdobeAIR.dmg) をダブルクリックします。次に、Scratch 2 をインストールします。インストールするためには、ダウンロードした Scratch-448.exe (Scratch-448.dmg) をダブルクリックします。

起動してみよう

⇒ デスクトップ上のアイコン



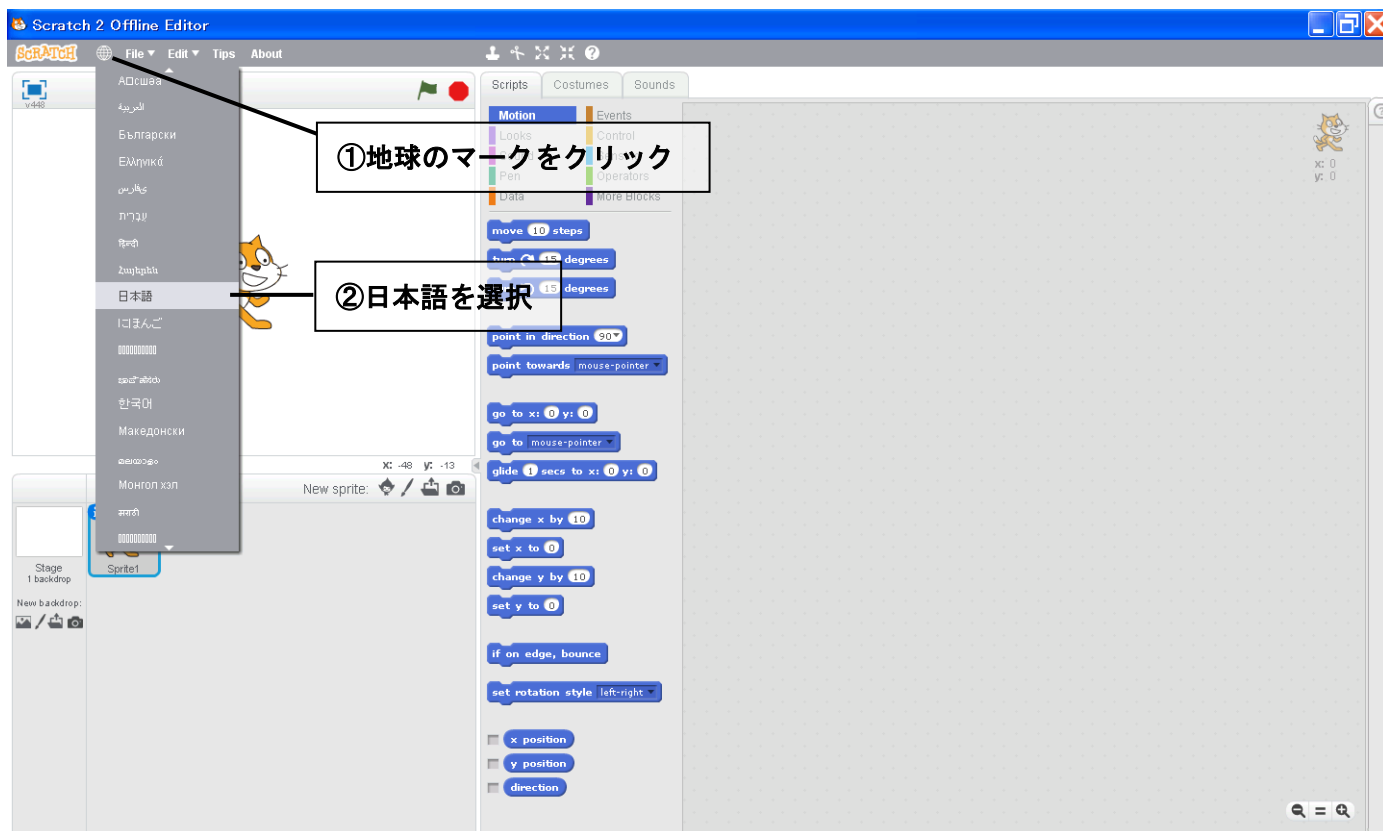
をダブルクリックします。

※終了するときは、メニューバーの「ファイル」をクリックして「終了」を選びます。

日本語の設定

もし日本語が表示されなかったら…

☞メニューバーにある地球のマークをクリックして「日本語」を選びます。

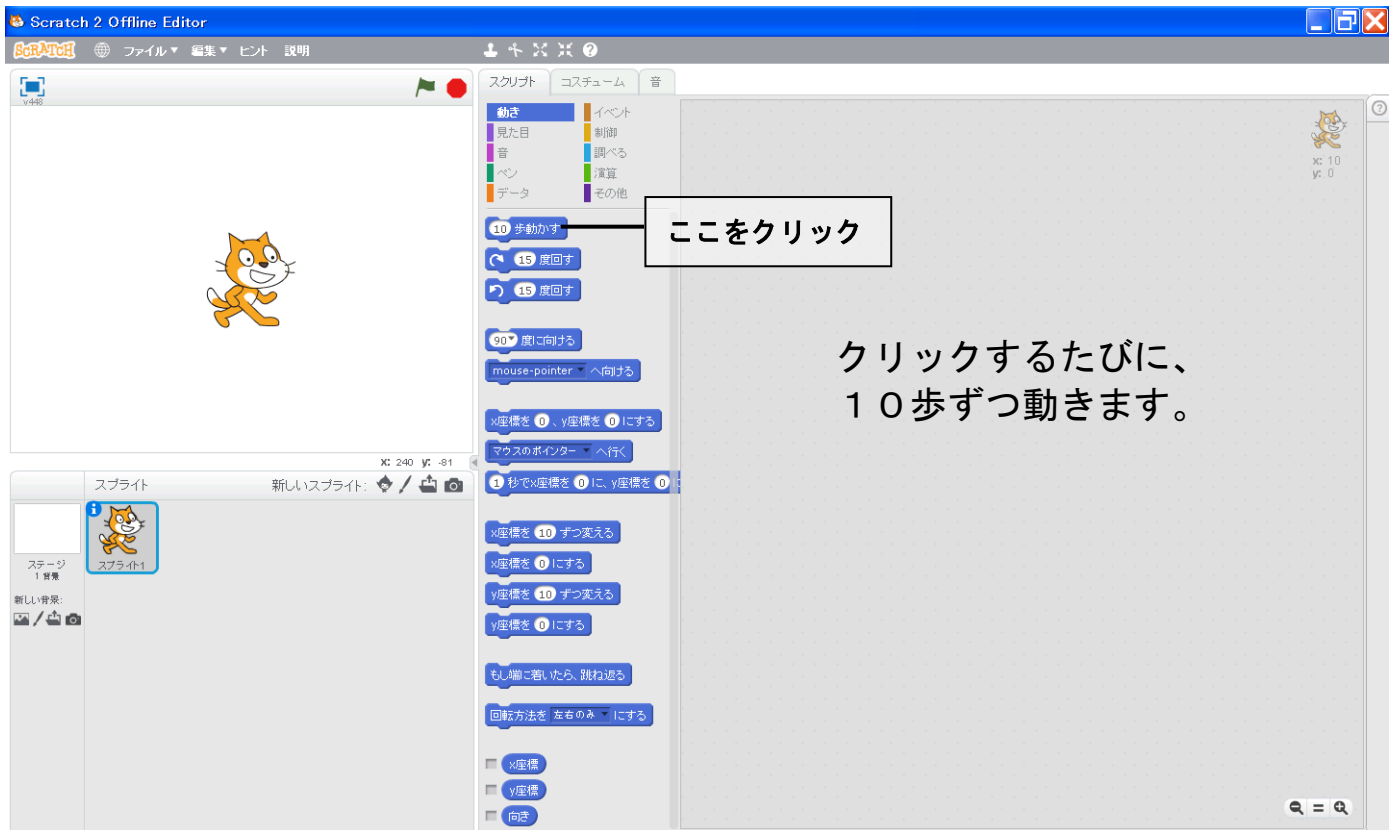


※低学年の児童の場合、「にほんご」を選ぶと、簡単な日本語表現にすることができます。

第2部：プログラミング

まずは、命令を試してみよう

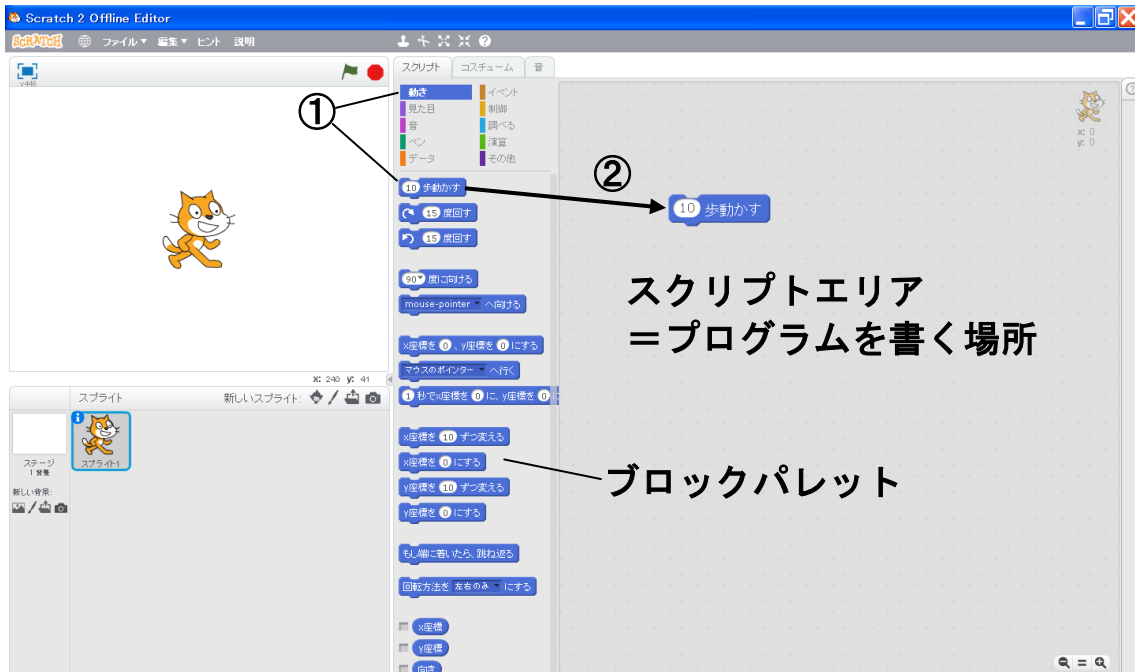
命令の内容：ネコを10歩動かす



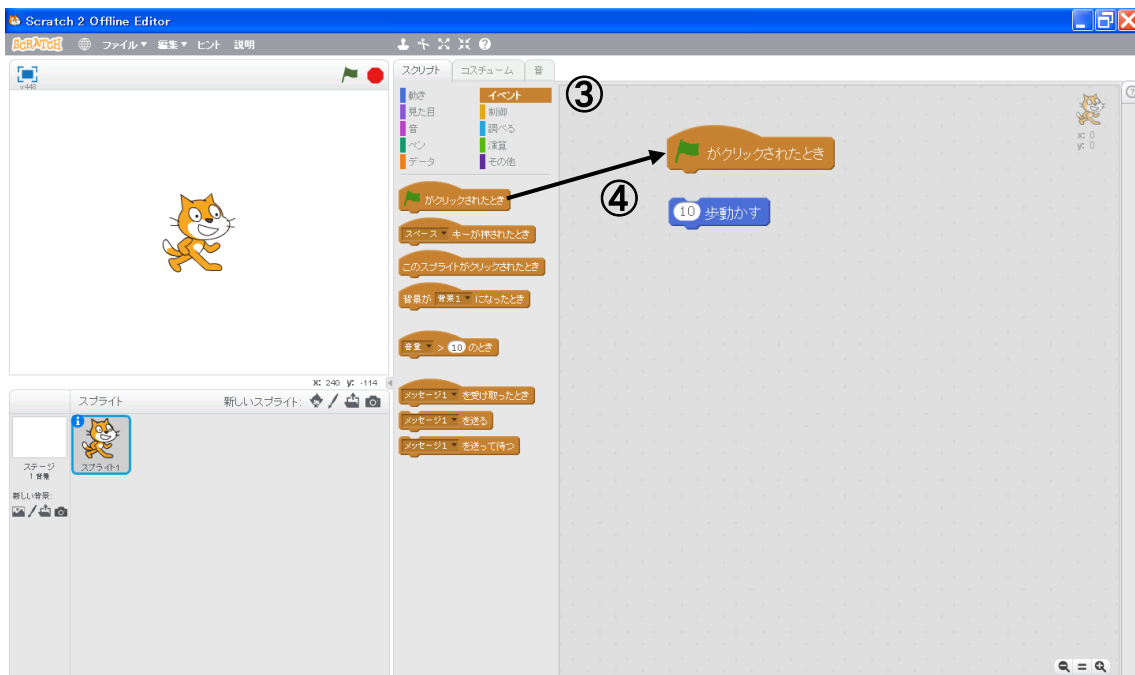
次に、簡単なプログラムを書いてみよう

プログラムの内容：ネコを10歩動かす

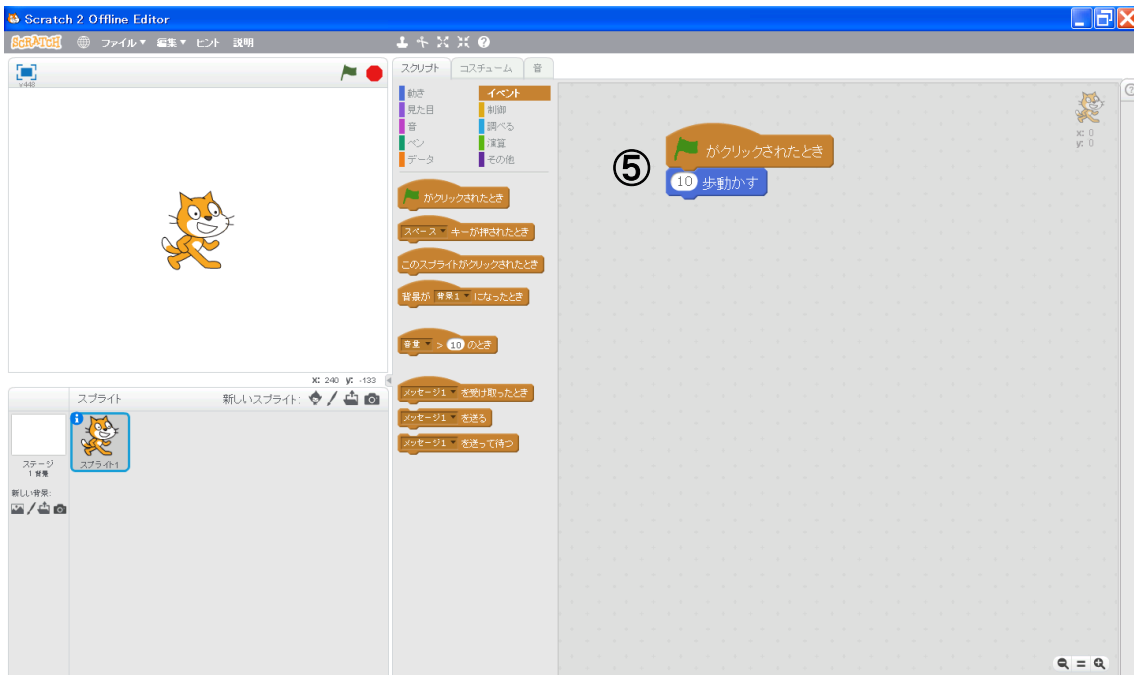
- ①中央のブロックパレットの中の動きから10歩動かすを選んで、
- ②右側のスクリプトエリアにドラッグする（クリックをしたまま引きずる）。



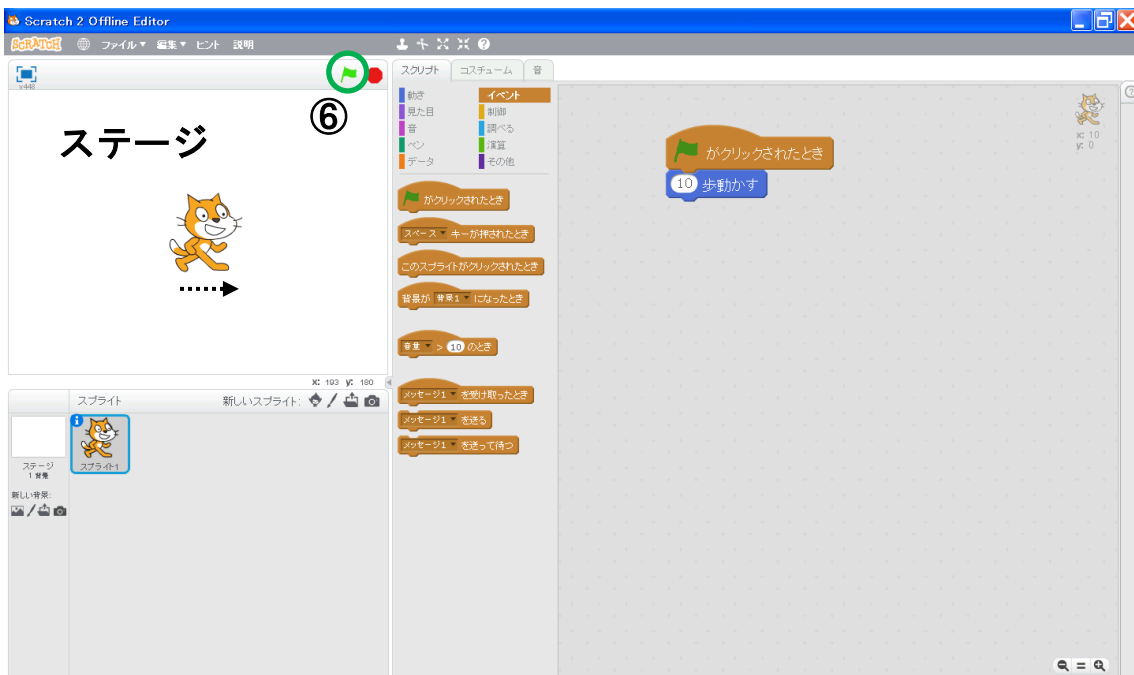
- ③中央のブロックパレットの中のイベントから緑の旗がクリックされたときを選んで、
- ④右側のスクリプトエリアにドラッグする。



⑤右側のスクリプトエリアで、10歩動かすと緑の旗がクリックされたときをくっつける。



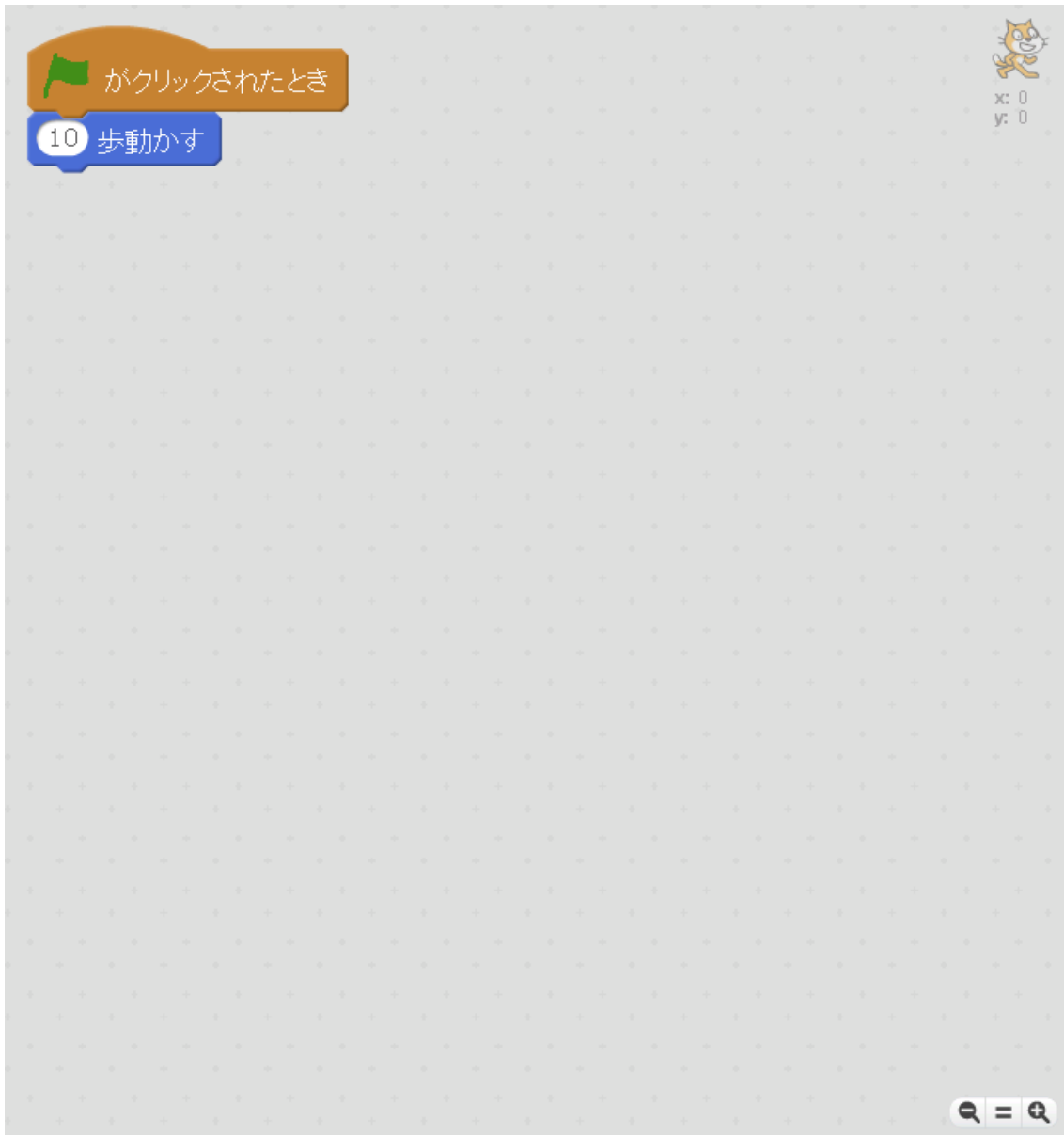
⑥左側のステージで緑の旗をクリックする。



※ステージ左上のアイコン  をクリックすると全画面表示にすることができます。

①ネコを動かそう

👉ブロックパレットからブロックを選んで、スクリプトエリアでプログラムを書こう。

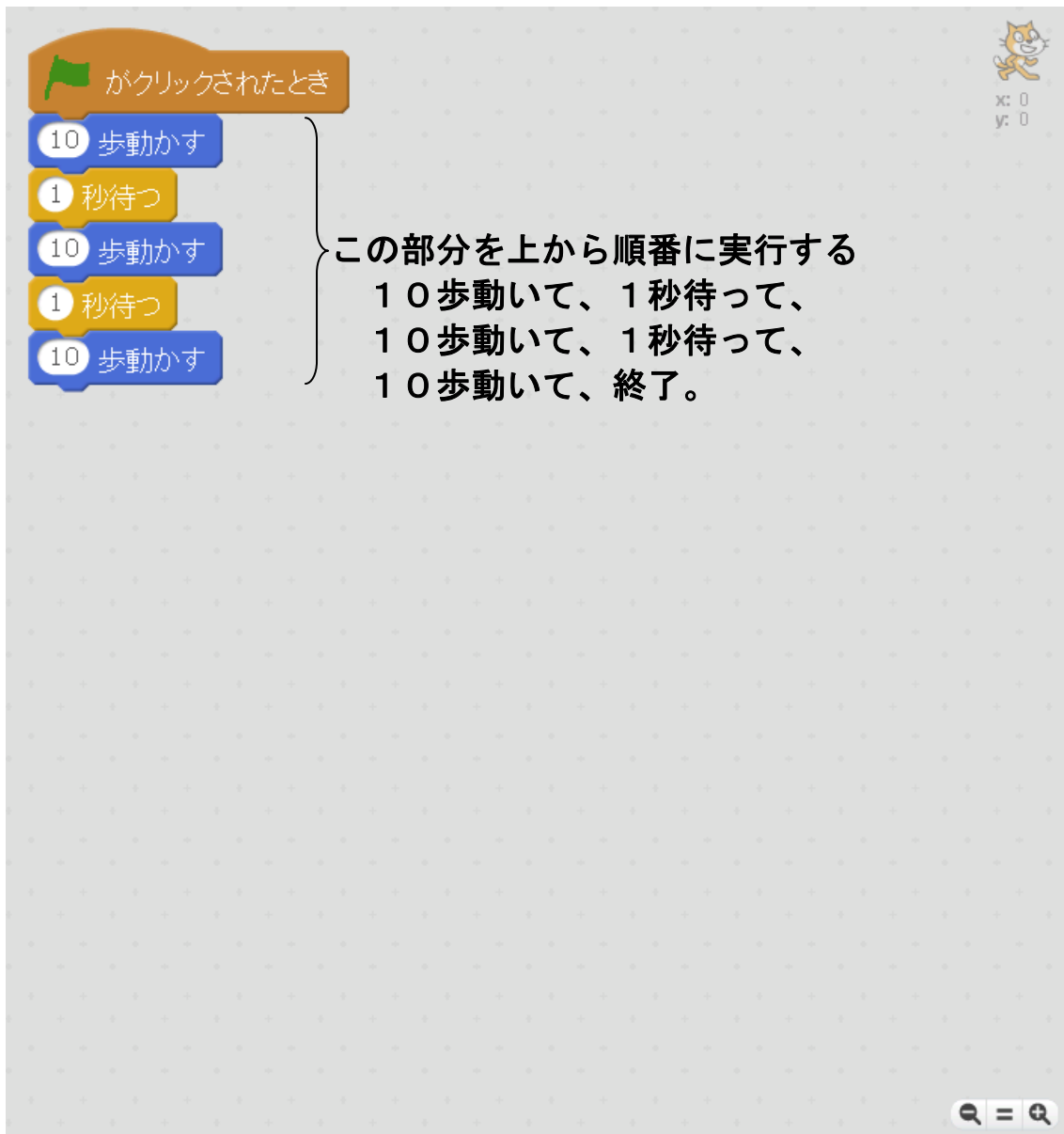


👉書き終わったら、ステージで**緑の旗**をクリックして実行してみよう。

👉実行し終わったら、メニューバーの「ファイル」から「名前をつけて保存」を選んで、プログラムを保存しよう。

②ネコを3回連続して動かそう

👉ブロックを追加してプログラムを完成しよう。



👉書き終わったら、ステージで**緑の旗**をクリックして実行してみよう。

👉実行し終わったら、メニューバーの「ファイル」から「名前をつけて保存」を選んで、プログラムを保存しよう。①とは別の名前で保存するとよいでしょう。

次に、10回連続して動かしてみましょう。「10歩動かす」「1秒待つ」を10回書いてもよいのですが、少し大変そうですね。。。 →次のページへ続く

③ネコを10回連続して動かそう

☞「〇回繰り返す」を追加して同じ処理を決められた回数だけ反復します。



※〇の中の数字は変更できます。数字の部分をクリックして、キーボードから入力してください。

不要になったブロックは、切り離れた後で、右クリックして「削除」を選べば消去できます。

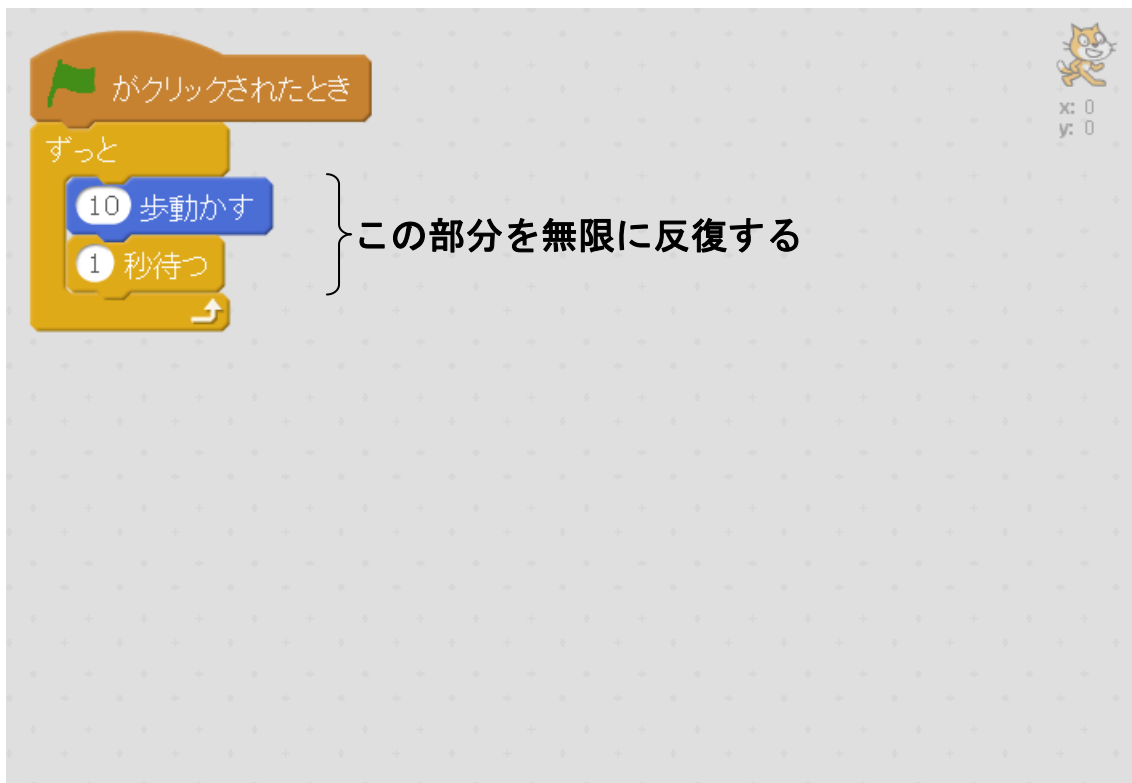
プログラミング言語では、たとえば、次のように書きます：

```
for (i=1; i<=10; i++) {  
    動かす (10. 0);  
    待つ (1. 0);  
}
```

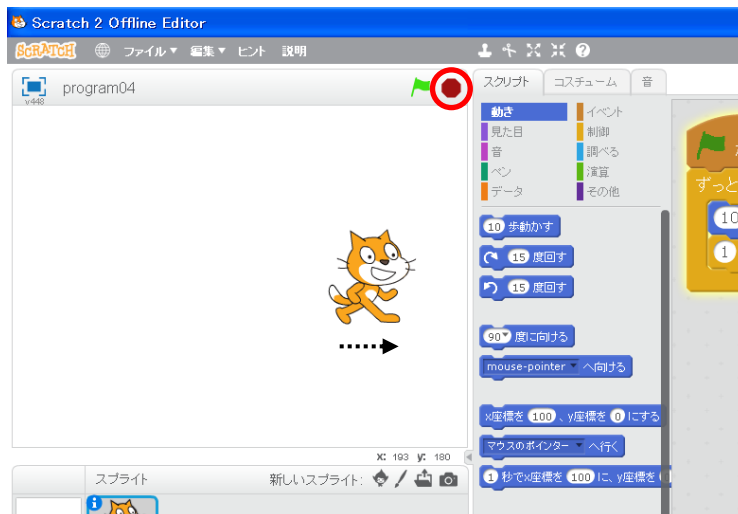
※代表的なプログラミング言語であるCを例にして、有限回の反復処理の書き方を概念的に示しています。日本語の部分は架空の文法です。

④ネコをずっと連続して動かそう

👉「ずっと」を追加して同じ処理を無限に（終了ボタンが押されるまで）反復します。



👉終了するときは、ステージの右上の**赤いボタン**（終了ボタン）をクリックする。



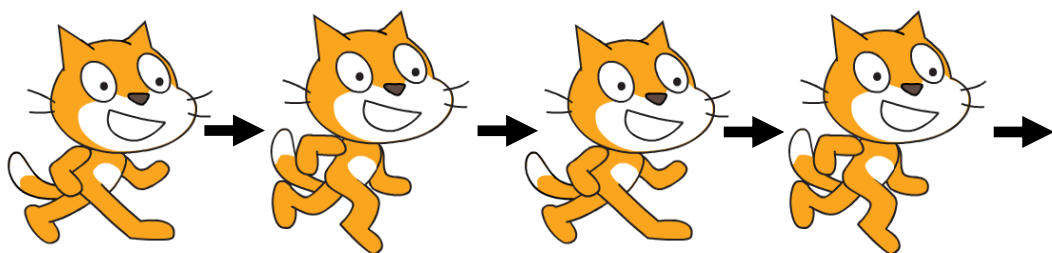
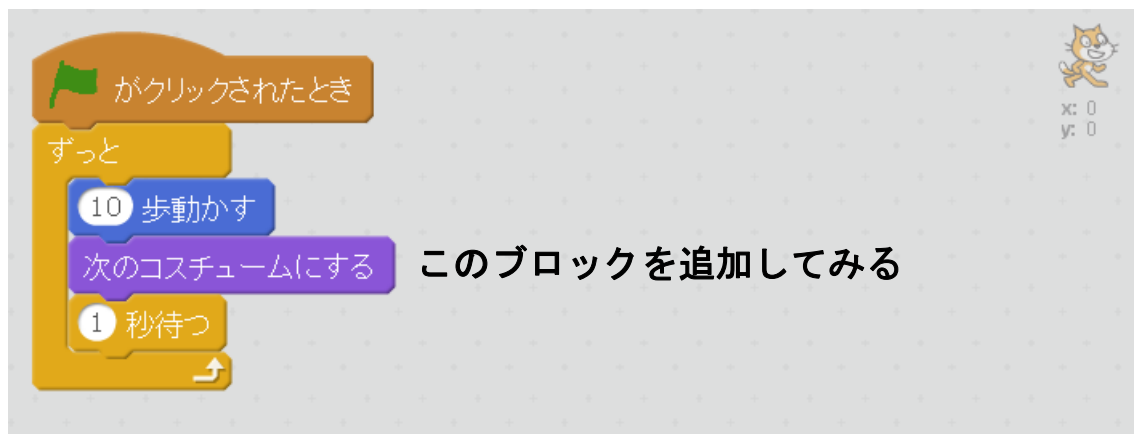
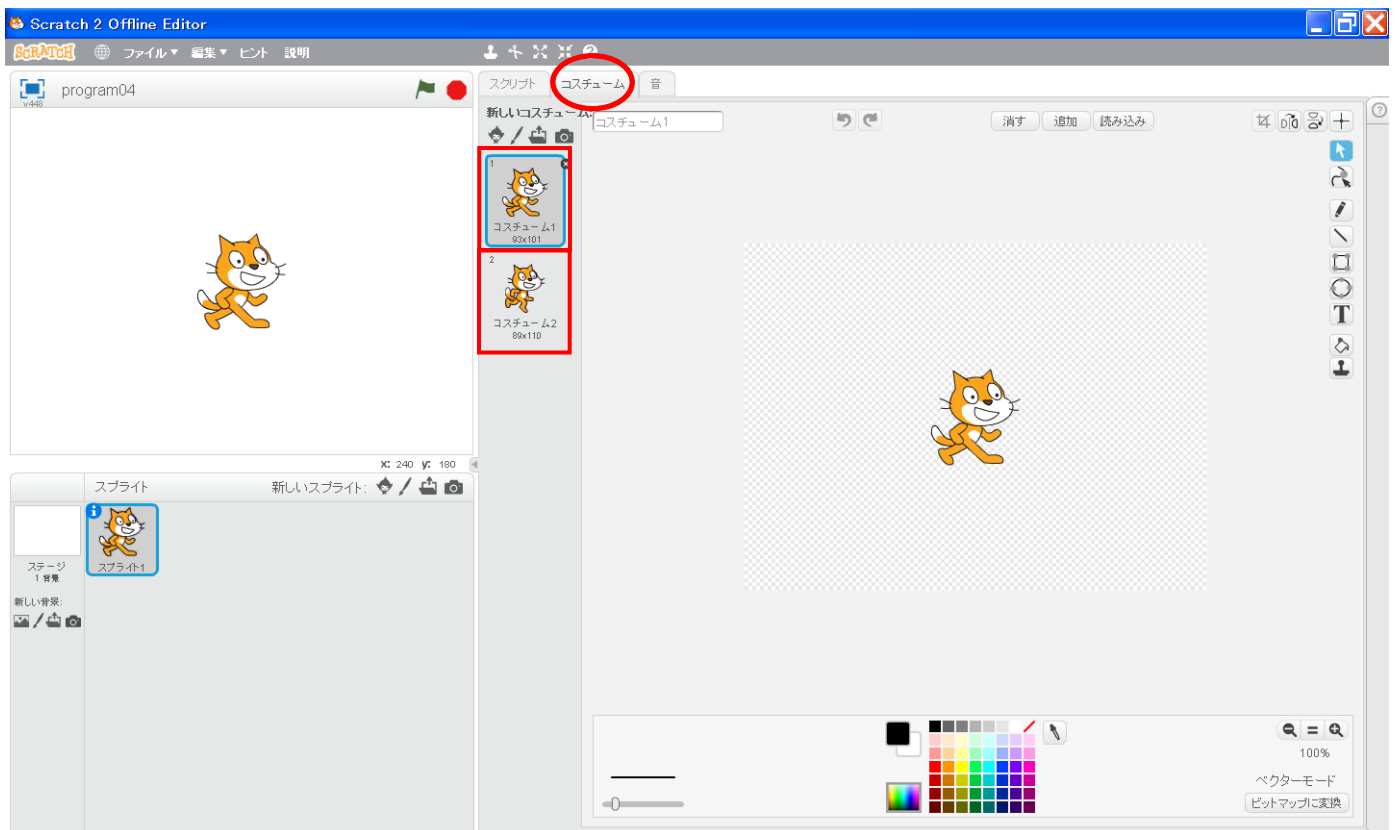
プログラミング言語では、たとえば、次のように書きます：

```
while (1) {  
    動かす (10. 0);  
    待つ (1. 0);  
}
```

※代表的なプログラミング言語であるCを例にして、無限回の反復処理の書き方を概念的に示しています。日本語の部分は架空の文法です。

寄り道：歩いているように見せる方法

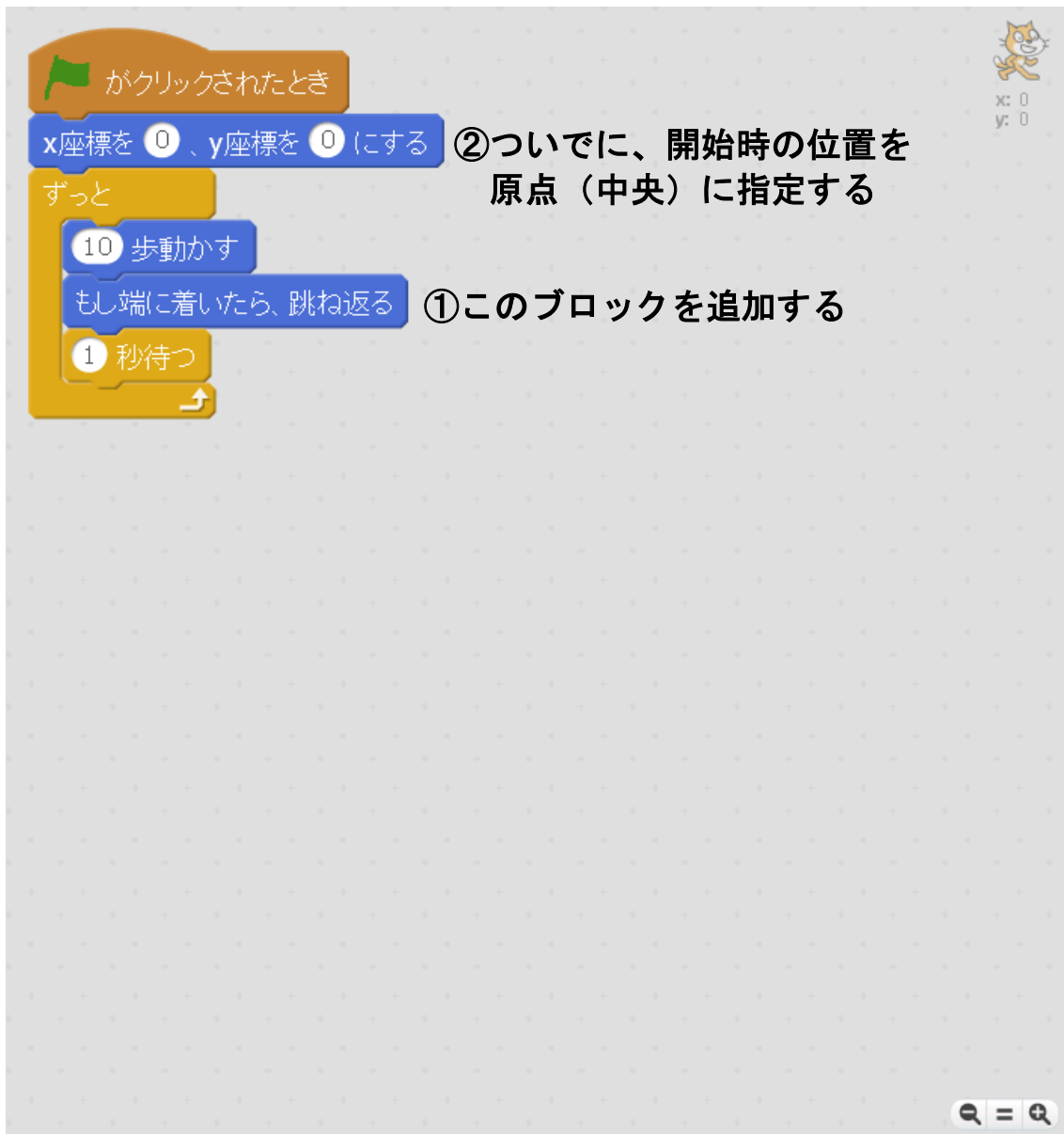
ネコ（スプライト1）には2つのコスチュームが用意されています。コスチューム1とコスチューム2を交互に表示することで歩いているように見せることができます。



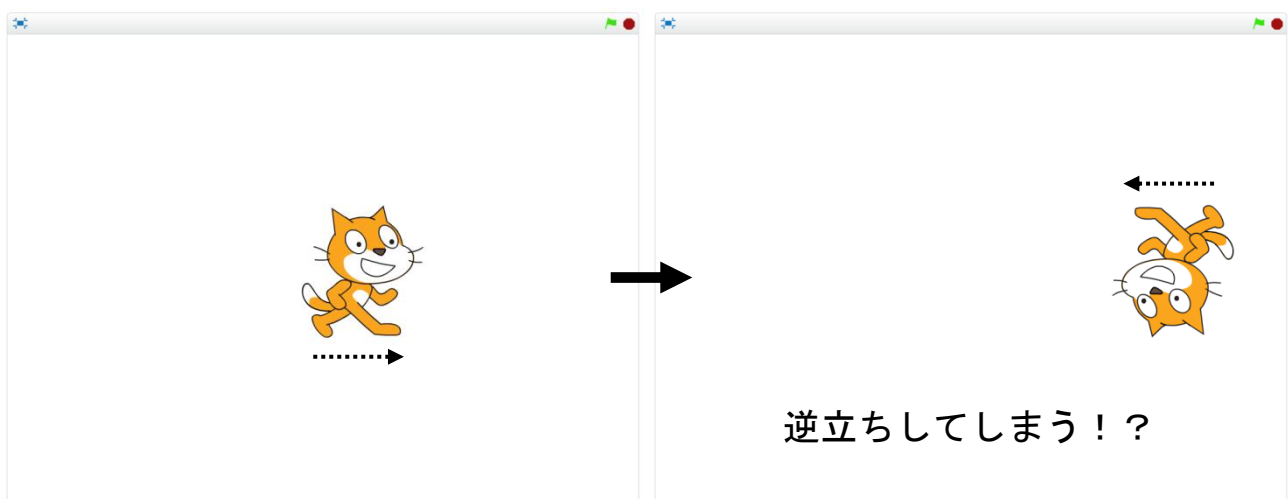
昔のテレビゲームは実際にこんな感じで動きを表現していたようです。

⑤端に着いたら、戻ってこよう

👉 ステージの端に着いたら、向きを変えて戻ってくるようにします。



実行例

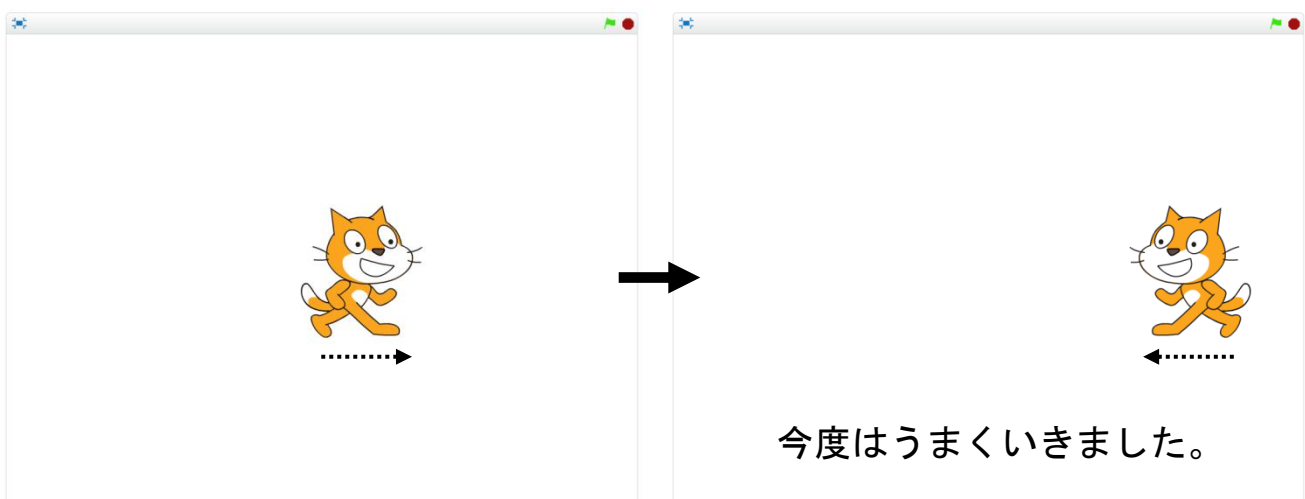


⑥逆立ちしないようにしよう

☞向きを変えたときに逆立ちしてしまわないように、開始時に設定を変更します。

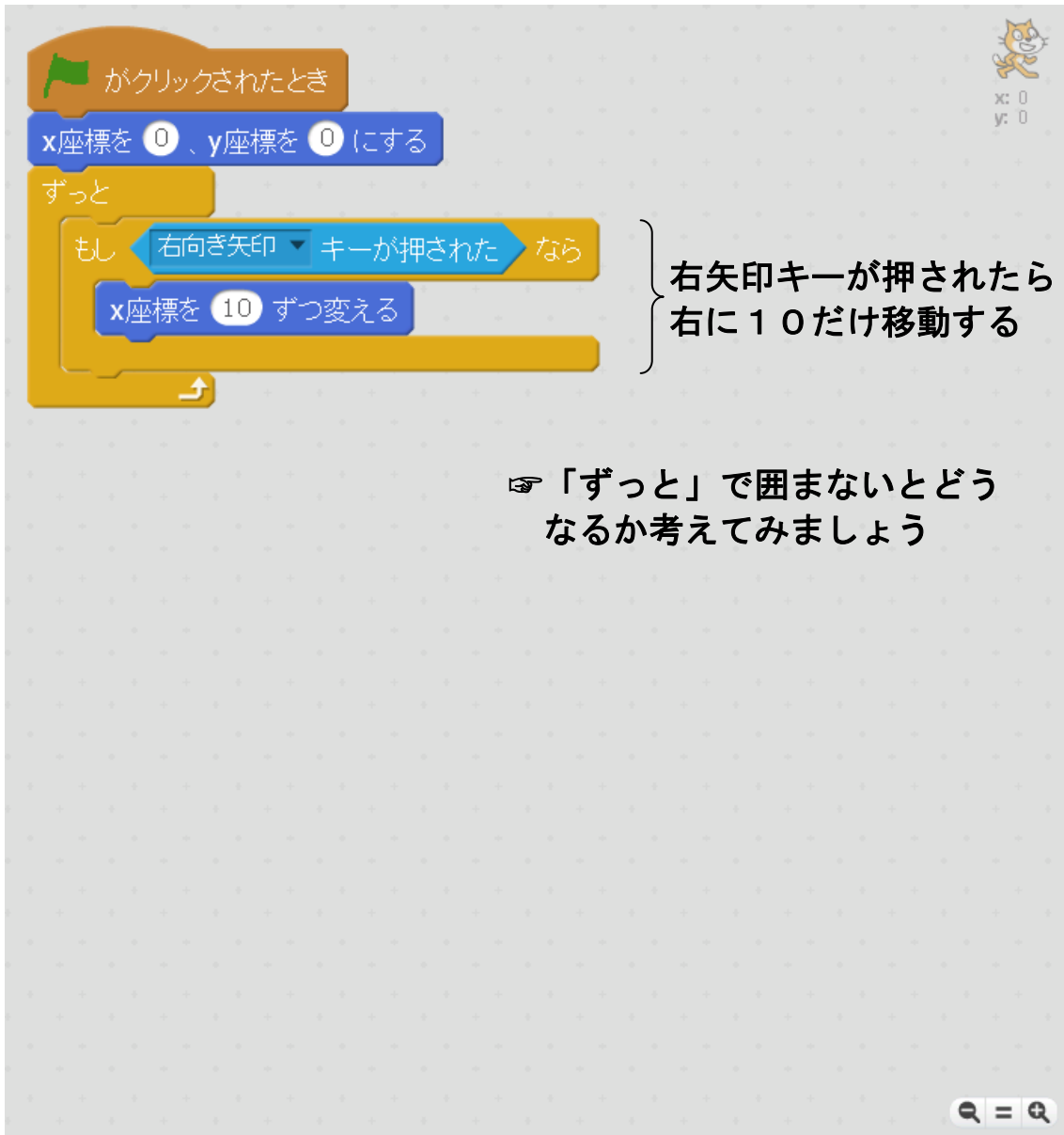


実行例



⑦キーボードで右に動かそう

☞ネコが自動で動くのではなく、キーを押したときだけ動くようにします。



The image shows a Scratch script on a cat sprite. The script starts with a 'when clicked' event block, followed by a 'set x to 0, set y to 0' block. Then, there is a 'loop forever' block containing an 'if right arrow key pressed' block and an 'x increases by 10' block. A bracket on the right side of the loop block is labeled '右矢印キーが押されたら 右に10だけ移動する' (When the right arrow key is pressed, move 10 to the right). Below the script, a text box asks '☞「ずっと」で囲まないとう なるか考えてみましょう' (Let's think about whether it will work without the 'loop forever' block).

がクリックされたとき

x座標を 0、y座標を 0 にする

ずっと

もし 右向き矢印 キーが押された なら

x座標を 10 ずつ変える

右矢印キーが押されたら 右に10だけ移動する

☞「ずっと」で囲まないとう なるか考えてみましょう

プログラミング言語では、たとえば、次のように書きます：

```
x = 0.0;
y = 0.0;
while (1) {
    if (右矢印キーが押された) {
        x = x + 10.0;
    }
}
```

※代表的なプログラミング言語であるCを例にして、反復処理や条件分岐の書き方を概念的に示しています。日本語の部分は架空の文法です。

⑧キーボードで4方向に動かそう

👉 4方向に動けるようにします。



The image shows a Scratch script for a character to move in four directions using a keyboard. The script starts with a 'when clicked' event block, followed by an 'initialize x and y coordinates to 0' block. Then, a 'forever' loop contains four 'if key pressed' blocks for the right, left, up, and down arrow keys. Each key press triggers a 'change x or y coordinate by 10' block. A bracket on the right side of the script indicates that the four 'if key pressed' blocks should be added to the loop.

```
when clicked
  x座標を 0、y座標を 0 にする
  ずっと
    もし 右向き矢印 キーが押された なら
      x座標を 10 ずつ変える
    もし 左向き矢印 キーが押された なら
      x座標を -10 ずつ変える
    もし 上向き矢印 キーが押された なら
      y座標を 10 ずつ変える
    もし 下向き矢印 キーが押された なら
      y座標を -10 ずつ変える
  上へ
```

この部分を追加する

⑨ネズミを登場させよう

☞ネズミを登場させます。登場人物（スプライト）を追加し、ネズミのスクリプトを書きます。
ネコ（スプライト1）

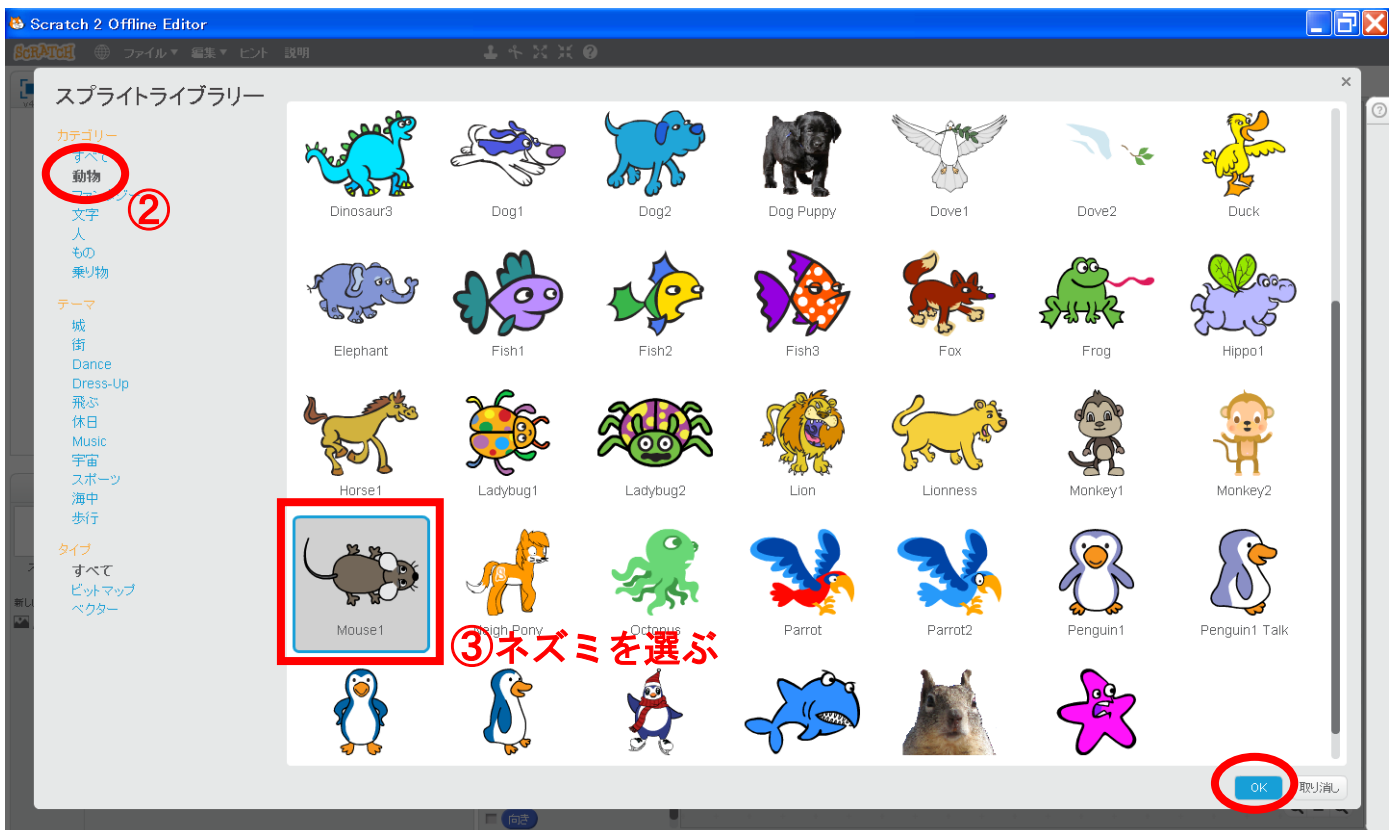
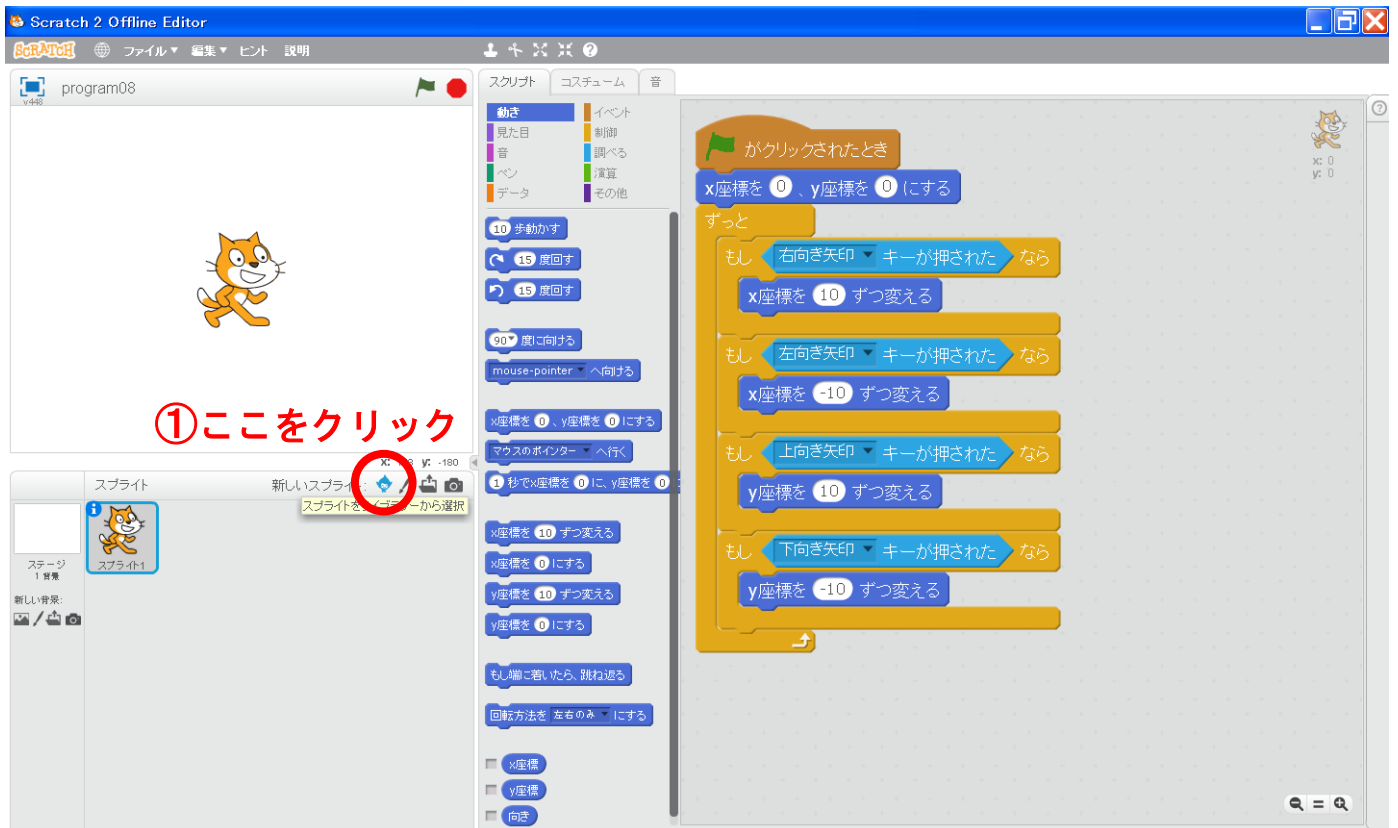


The image shows a Scratch script for a cat sprite. The script starts with a 'when clicked' event block, followed by a 'set x to 0, set y to 0' block. Then, there is a 'forever' loop containing four 'if' blocks. Each 'if' block checks for a specific arrow key being pressed (right, left, up, or down) and, if true, moves the cat 10 units in that direction. The script is written in Japanese.

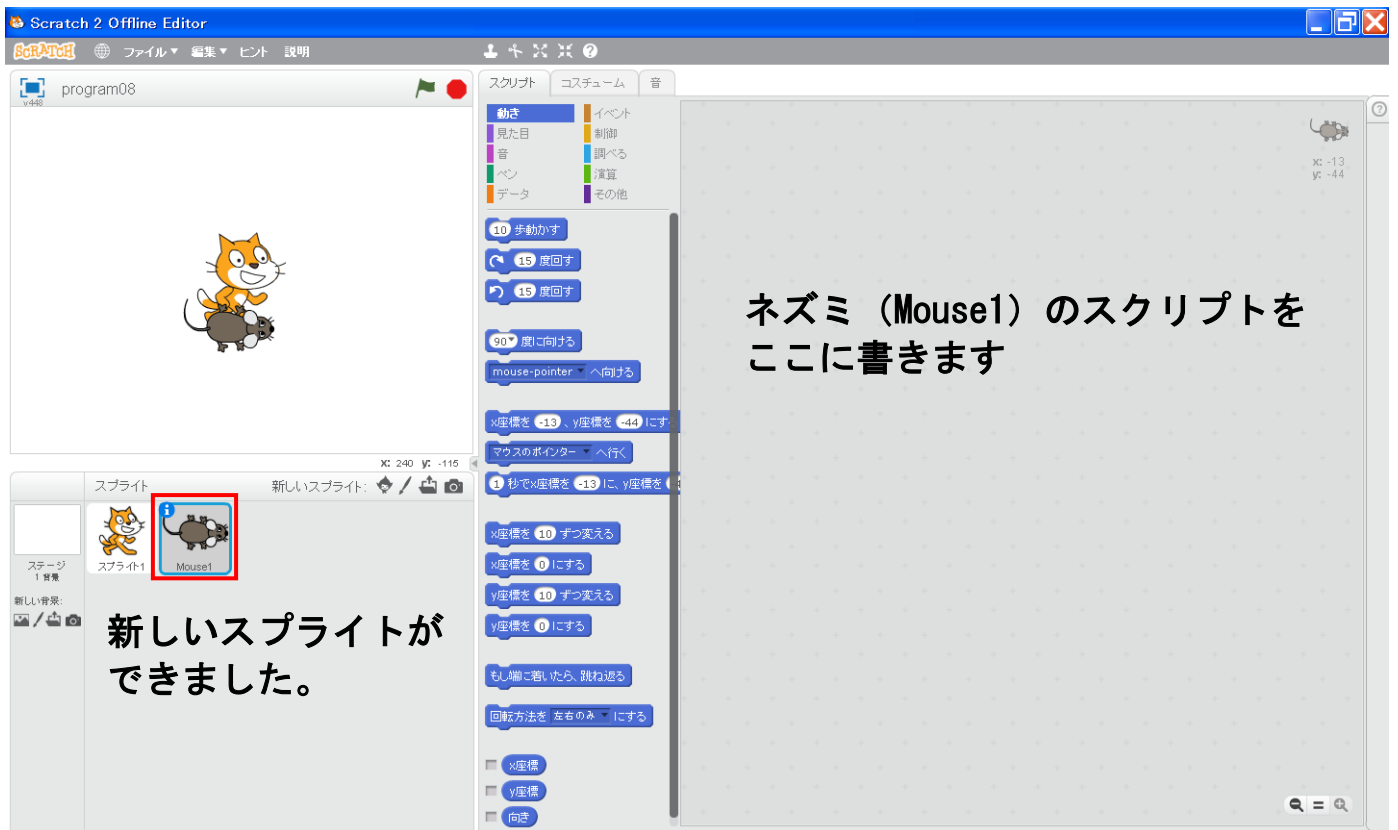
```
when clicked
  set x to 0, set y to 0
  loop
    if right arrow key pressed then
      move 10 units right
    if left arrow key pressed then
      move 10 units left
    if up arrow key pressed then
      move 10 units up
    if down arrow key pressed then
      move 10 units down
```

ネコのスクリプトに変更はありません

登場人物（スプライト）を追加する

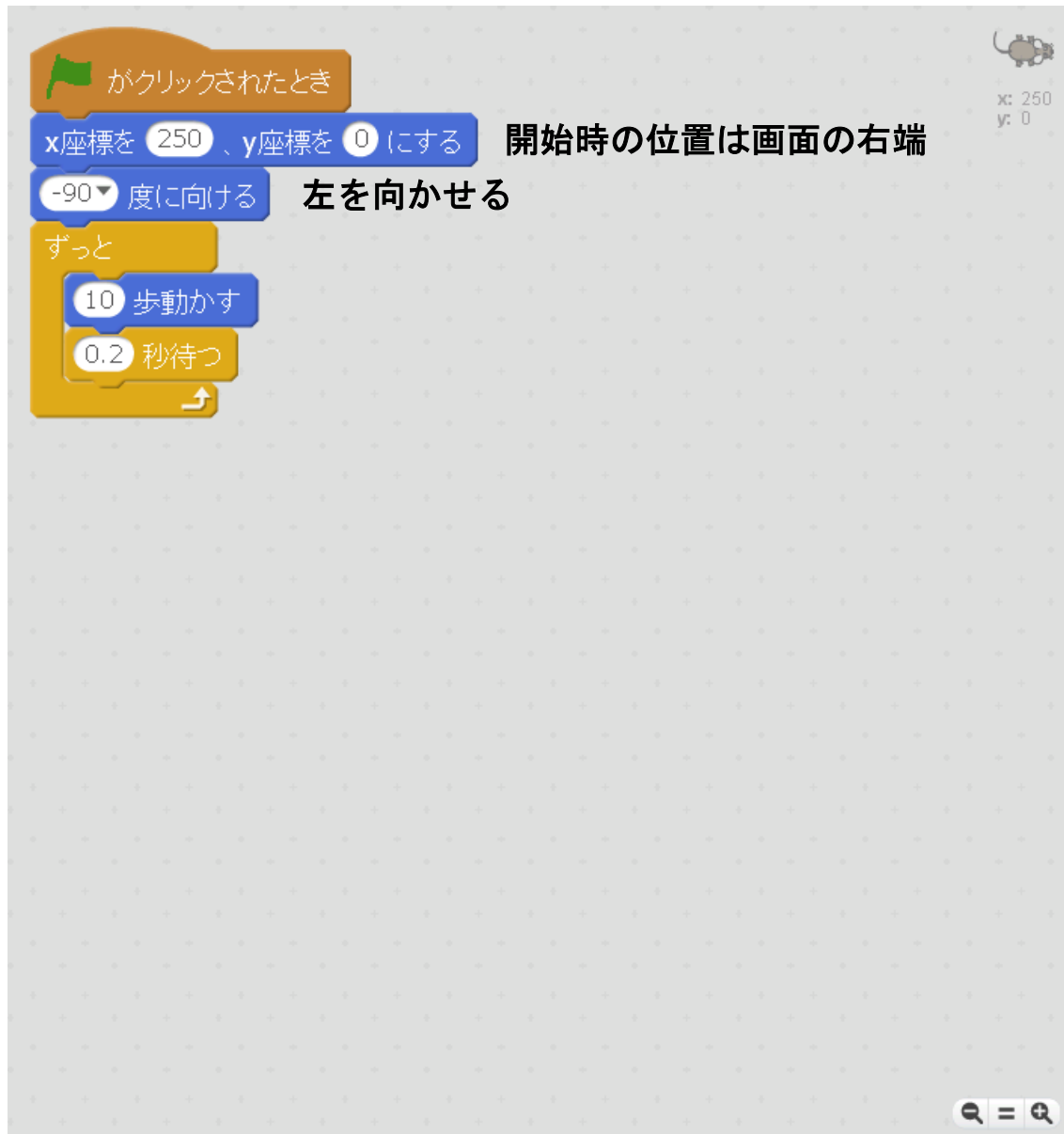


④OKをクリック

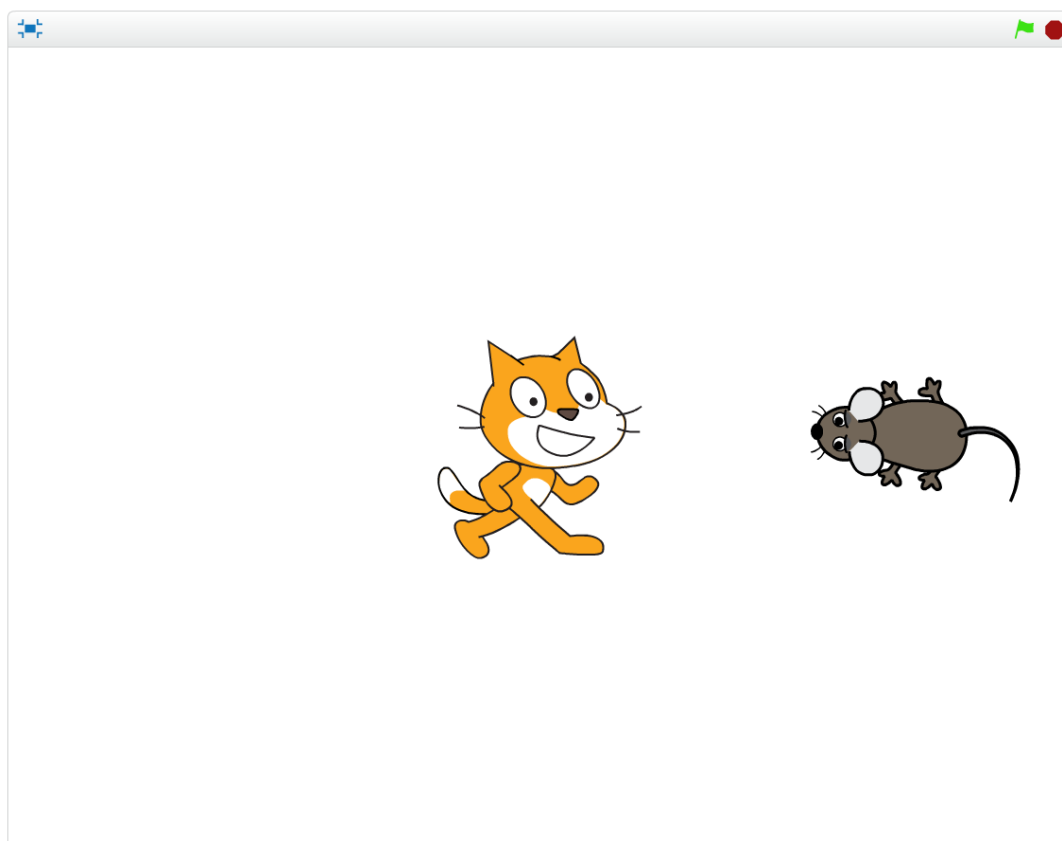


ネズミ (Mouse 1)

ネズミのスクリプトを新しく書きます。



実行例



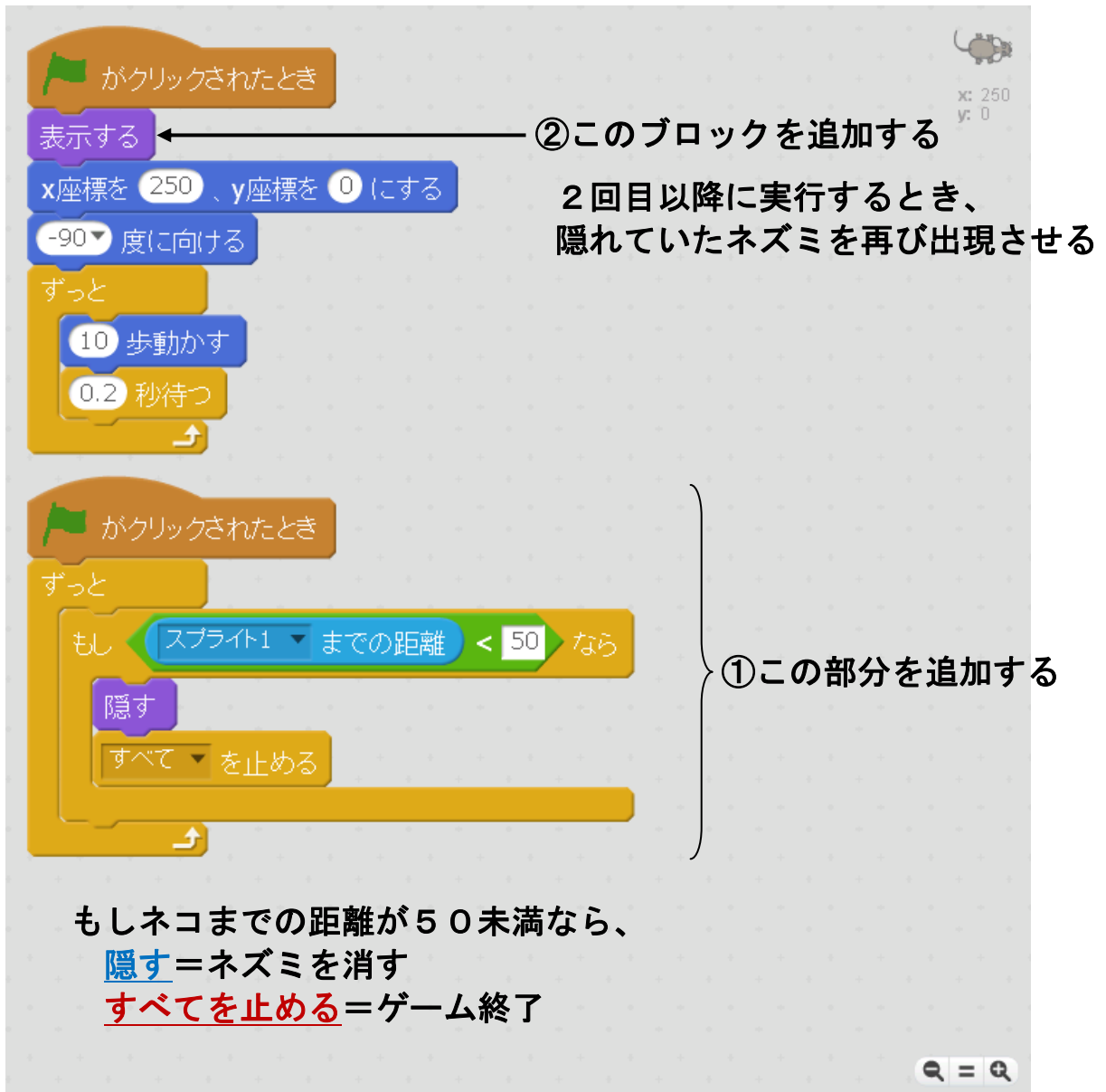
⑩ネズミを食べよう

ネコがネズミをつかまえて食べるようにします。

ネコ（スプライト1）

変更はありません。

ネズミ（Mouse1）



The image shows a Scratch script for the mouse sprite (Mouse1). The script is divided into two main sections. The first section, labeled ②, starts with a 'when clicked' event block, followed by a 'show' block, then a 'set x to 250, y to 0' block, a 'turn 90 degrees' block, and a 'repeat' loop with 'forever' as the condition. Inside the loop are 'move 10 steps' and 'wait 0.2 seconds' blocks. The second section, labeled ①, also starts with a 'when clicked' event block, followed by a 'repeat' loop with 'forever' as the condition. Inside this loop is an 'if' block with the condition 'distance to sprite 1 < 50'. If the condition is true, the 'hide' block is executed, followed by a 'stop all' block. Annotations with arrows point to these sections, explaining their purpose: ② is for reappearing the mouse when clicked, and ① is for hiding the mouse when the cat is close enough to eat it.

②このブロックを追加する
2回目以降に実行するとき、
隠れていたネズミを再び出現させる

①この部分を追加する

もしネコまでの距離が50未満なら、
隠す＝ネズミを消す
すべてを止める＝ゲーム終了

⑪ネズミを食べたら、ニャーと鳴こう

👉ゲームを楽しくするために、音も出るようにします。

「ニャー」と鳴くのは、ネズミではなくネコです。そこで次のような手順でネコを鳴かせます。

1. ネズミの側で「猫に食べられた」と判定したら、
2. ネズミからネコに「猫に食べられた」というメッセージを送る。
3. ネコの側で「猫に食べられた」というメッセージを受け取ったら、
4. ネコが「ニャー」と鳴く。

ネコ（スプライト1）

＜次のページを先にやってからこのページに戻ります＞

がクリックされたとき

x座標を 0、y座標を 0 にする

ずっと

もし 右向き矢印 ▼ キーが押された なら

x座標を 10 ずつ変える

もし 左向き矢印 ▼ キーが押された なら

x座標を -10 ずつ変える

もし 上向き矢印 ▼ キーが押された なら

y座標を 10 ずつ変える

もし 下向き矢印 ▼ キーが押された なら

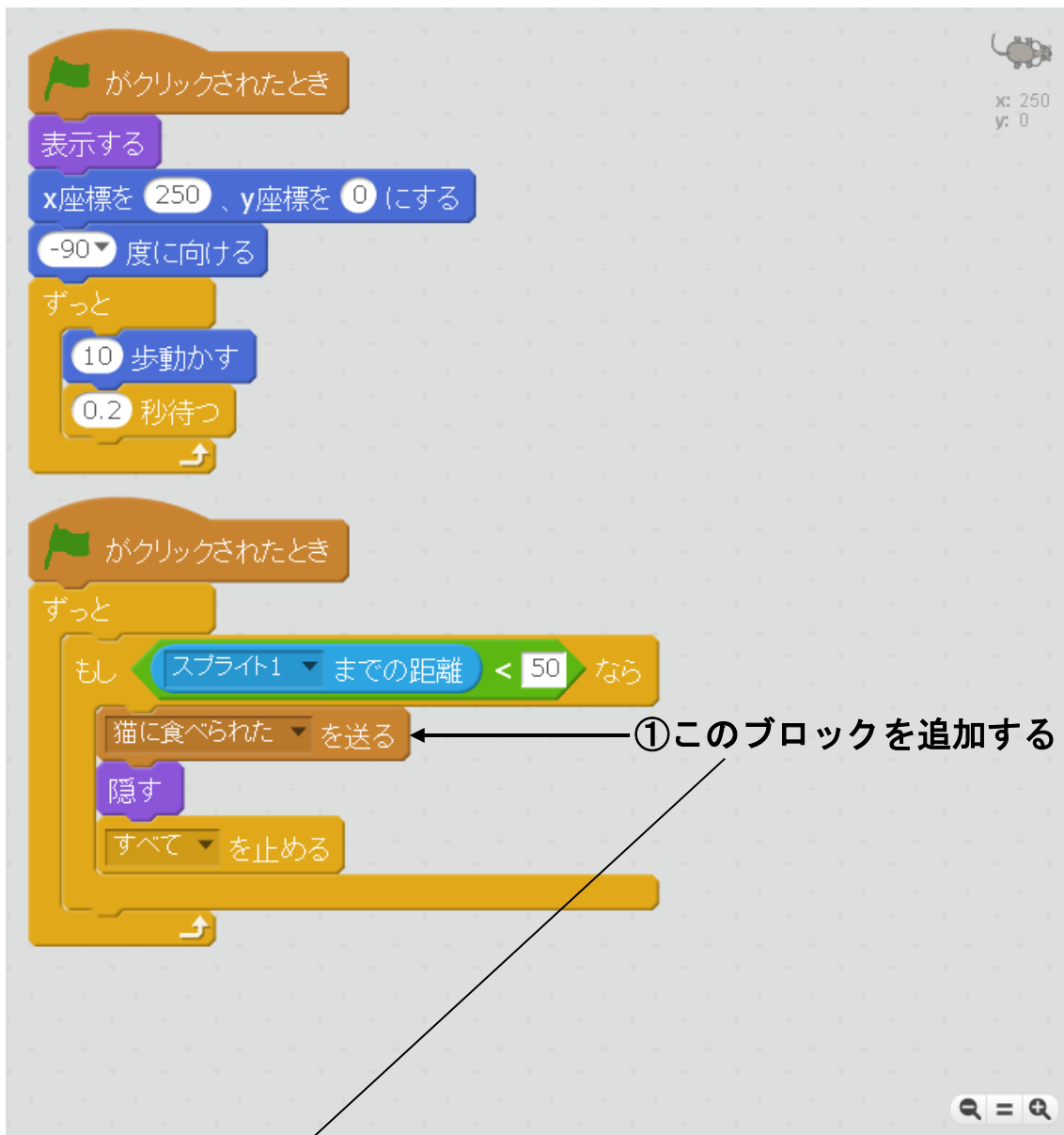
y座標を -10 ずつ変える

猫に食べられた ▼ を受け取ったとき

ニャー ▼ の音を鳴らす

②この部分を追加する

ネズミ (Mouse 1)



※メッセージを定義する方法



⑫ネズミをランダムな位置から登場させよう

☞いつも同じ場所ではなく、ランダムな場所からネズミが出てくるようにします。

ネコ（スプライト1）

変更はありません。

ネズミ（Mouse 1）

がクリックされたとき

表示する

x座標を 250 、y座標を -150 から 150 までの乱数 にする

-90 度に向ける

ずっと

10 歩動かす

0.2 秒待つ

この部分に**乱数**のブロックを入れる

乱数＝適当な数がランダムに決まる

がクリックされたとき

ずっと

もし スプライト1 までの距離 < 50 なら

猫に食べられた を送る

隠す

すべて を止める

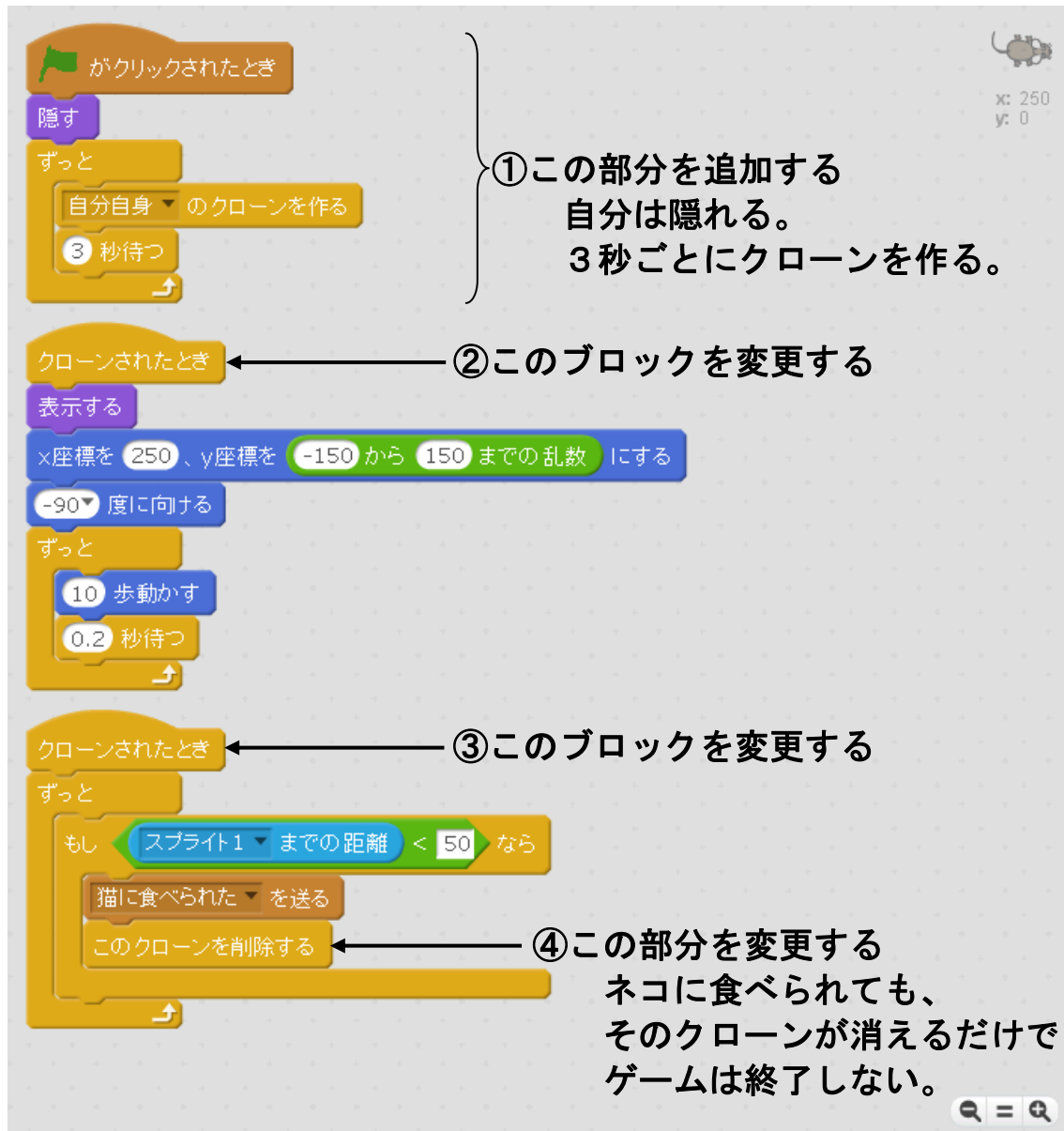
⑬ネズミをたくさん登場させよう

ネズミが1匹だけでは物足りないので、たくさん出てくるようにします。

ネコ（スプライト1）

変更はありません。

ネズミ（Mouse 1）



The image shows a Scratch script for a mouse sprite. The script is divided into three main sections, each starting with a 'when clicked' event block. The first section (labeled ①) contains a 'hide' block, a 'forever' loop with 'create a clone of myself' and a 'wait 3 seconds' block. The second section (labeled ②) starts with a 'when cloned' event, followed by 'show', 'set x and y coordinates to random values between -150 and 150', 'set direction to -90 degrees', and a 'forever' loop with 'move 10 steps' and 'wait 0.2 seconds'. The third section (labeled ③) starts with a 'when cloned' event, followed by a 'forever' loop. Inside the loop is an 'if' block: 'if distance to sprite 1 is less than 50, then send a message "eaten by cat" and delete this clone' (labeled ④). A mouse icon in the top right corner shows coordinates x: 250, y: 0.

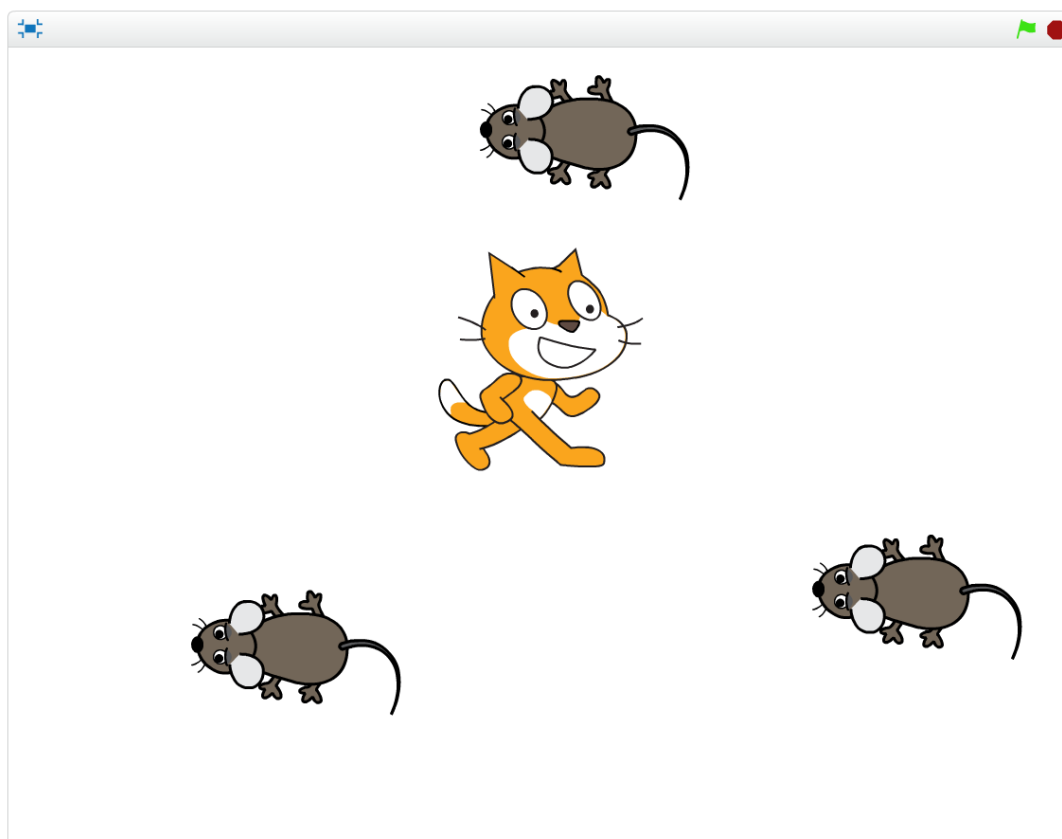
①この部分を追加する
自分は隠れる。
3秒ごとにクローンを作る。

②このブロックを変更する

③このブロックを変更する

④この部分を変更する
ネコに食べられても、
そのクローンが消えるだけで
ゲームは終了しない。

実行例



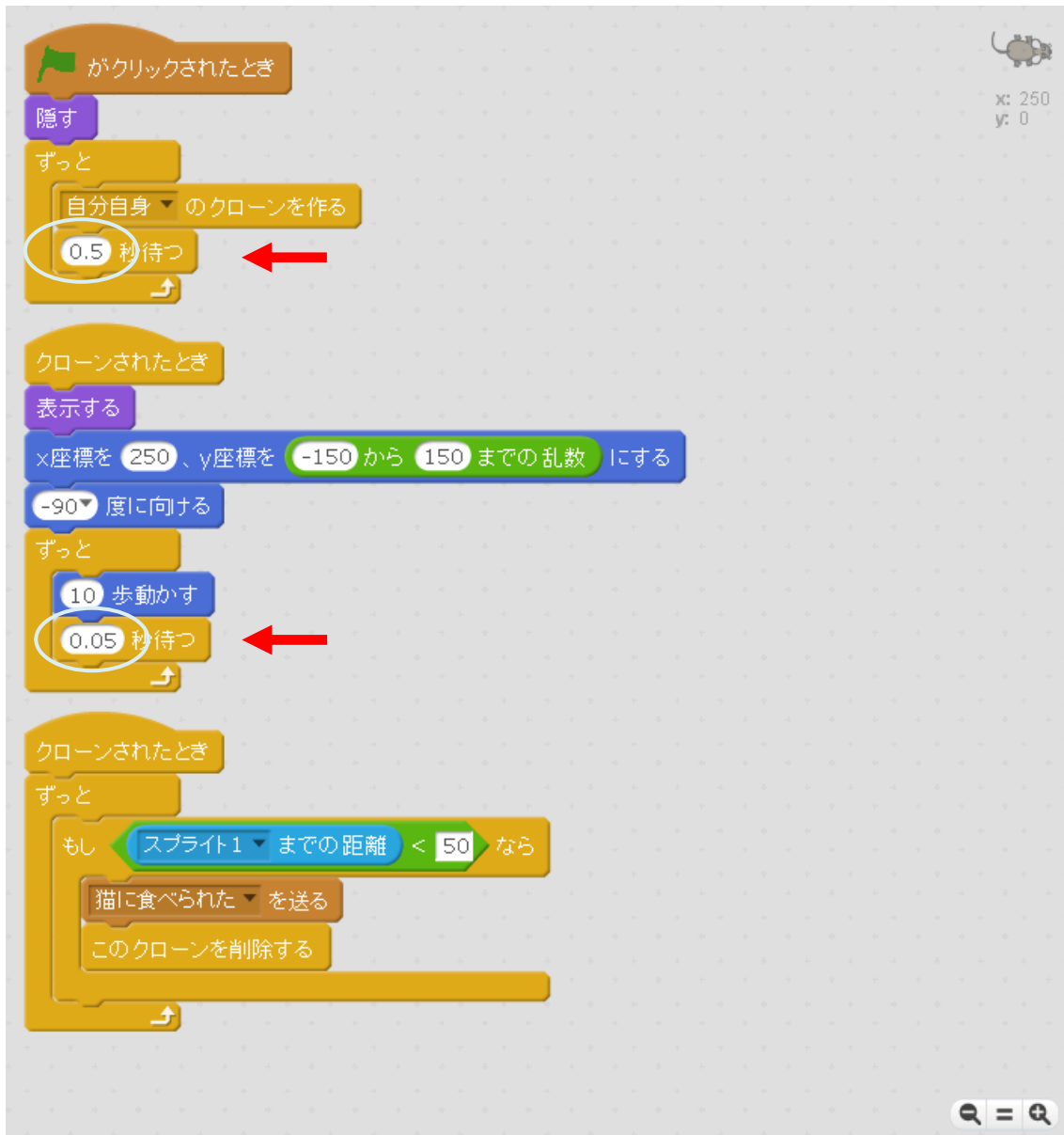
⑭ゲームの進行を速くしよう

☞設定を変えて、スリルのあるゲームにします。

ネコ（スプライト1）

変更はありません。

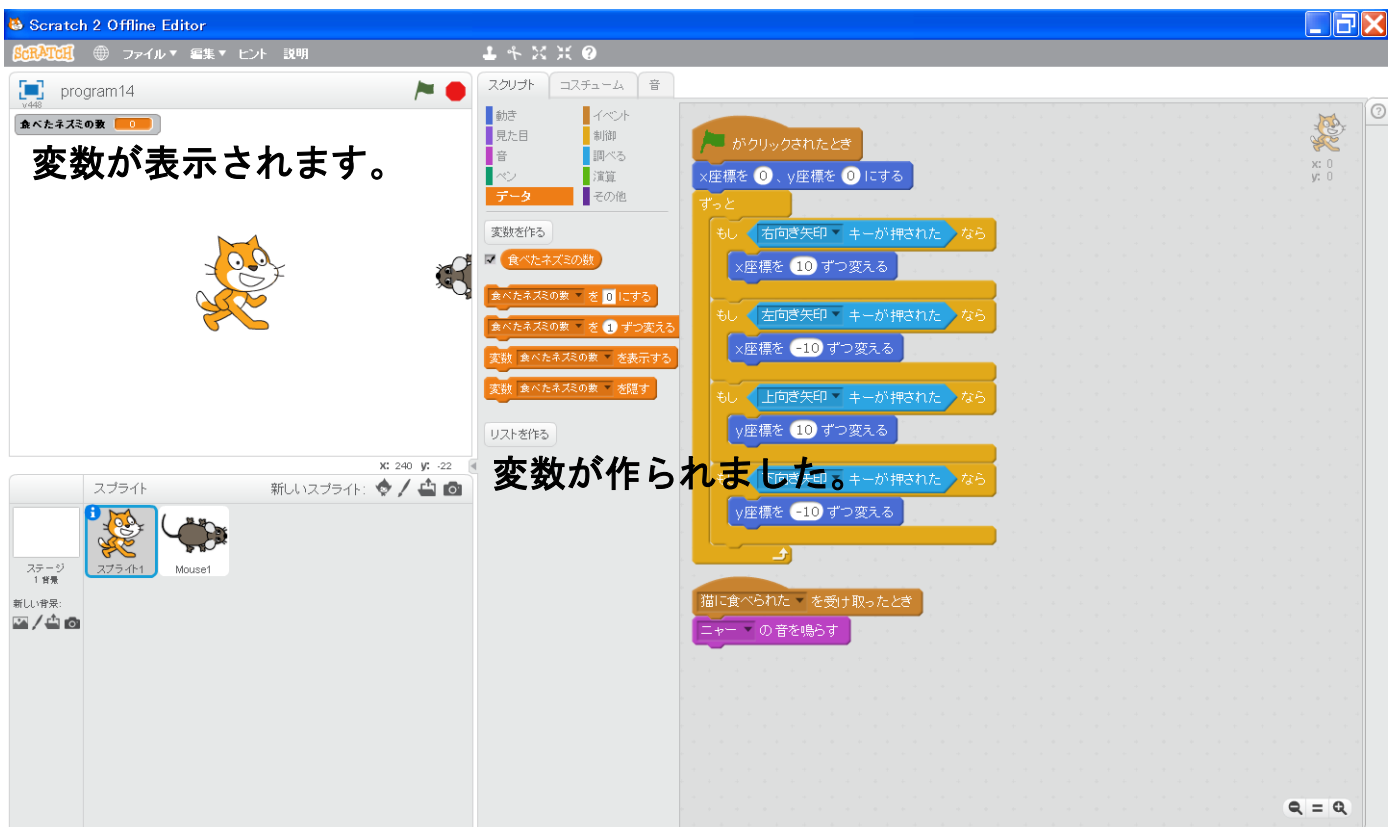
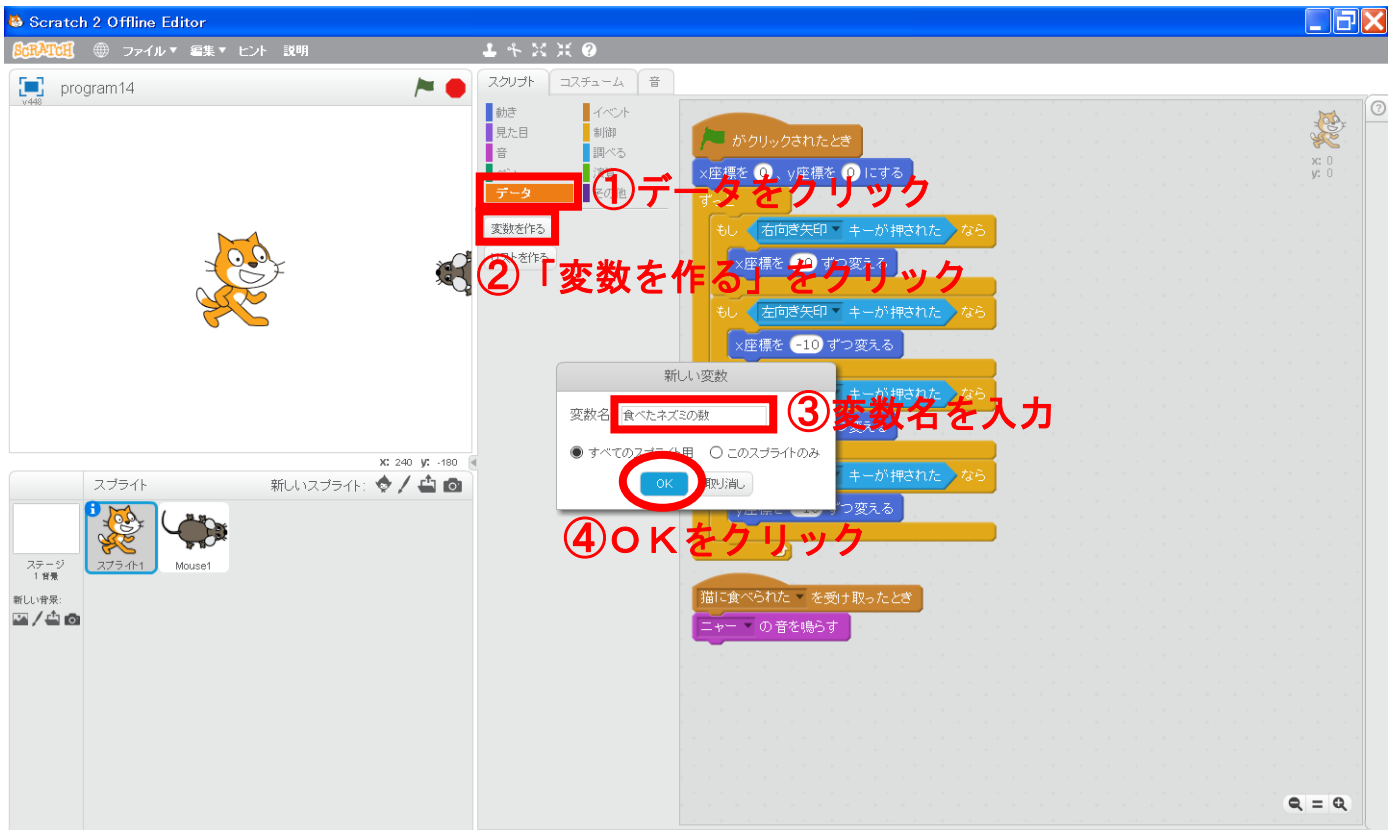
ネズミ（Mouse 1）



⑮食べたネズミの数を数えよう

☞変数を定義して、ネコに食べられたネズミの数を数えます。

変数を定義する



ネコ（スプライト1）

がクリックされたとき

食べたネズミの数 を 0 にする ①このブロックを追加する

x座標を 0、y座標を 0 にする

ずっと

もし 右向き矢印 キーが押された なら

x座標を 10 ずつ変える

もし 左向き矢印 キーが押された なら

x座標を -10 ずつ変える

もし 上向き矢印 キーが押された なら

y座標を 10 ずつ変える

もし 下向き矢印 キーが押された なら

y座標を -10 ずつ変える

猫に食べられた を受け取ったとき

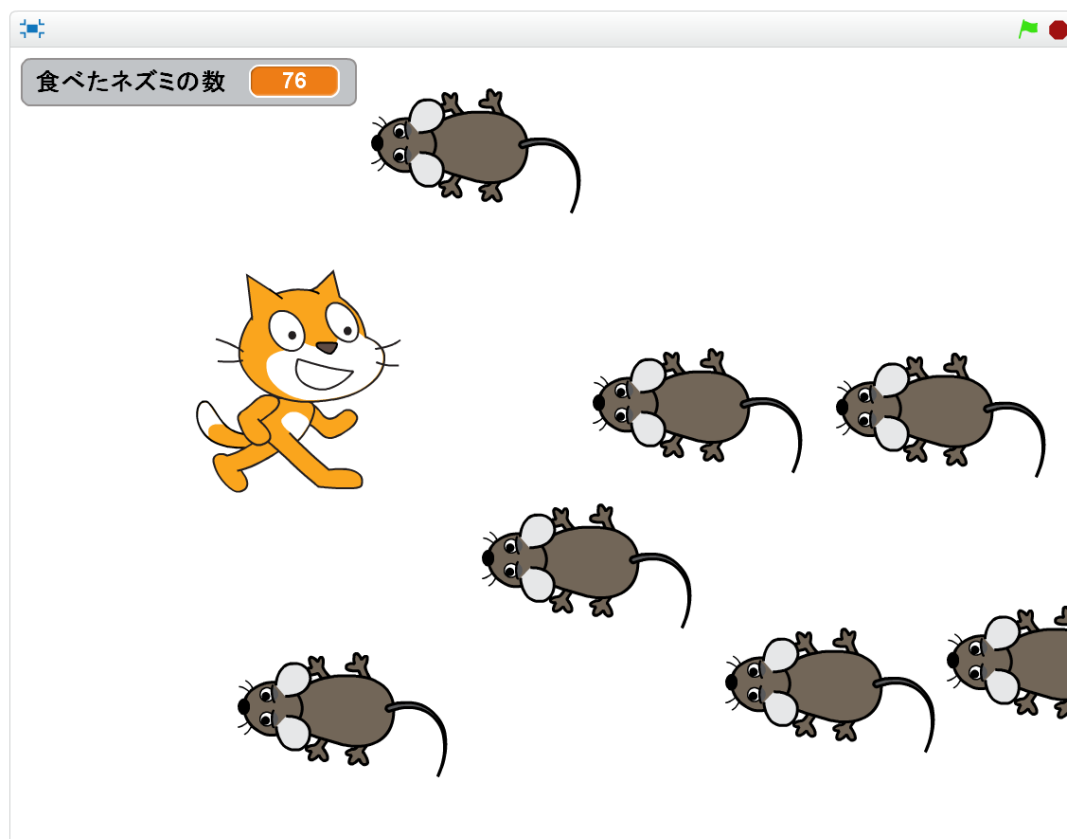
食べたネズミの数 を 1 ずつ変える ②このブロックを追加する

ニャー の音を鳴らす

ネズミ（Mouse1）

変更はありません。

実行例

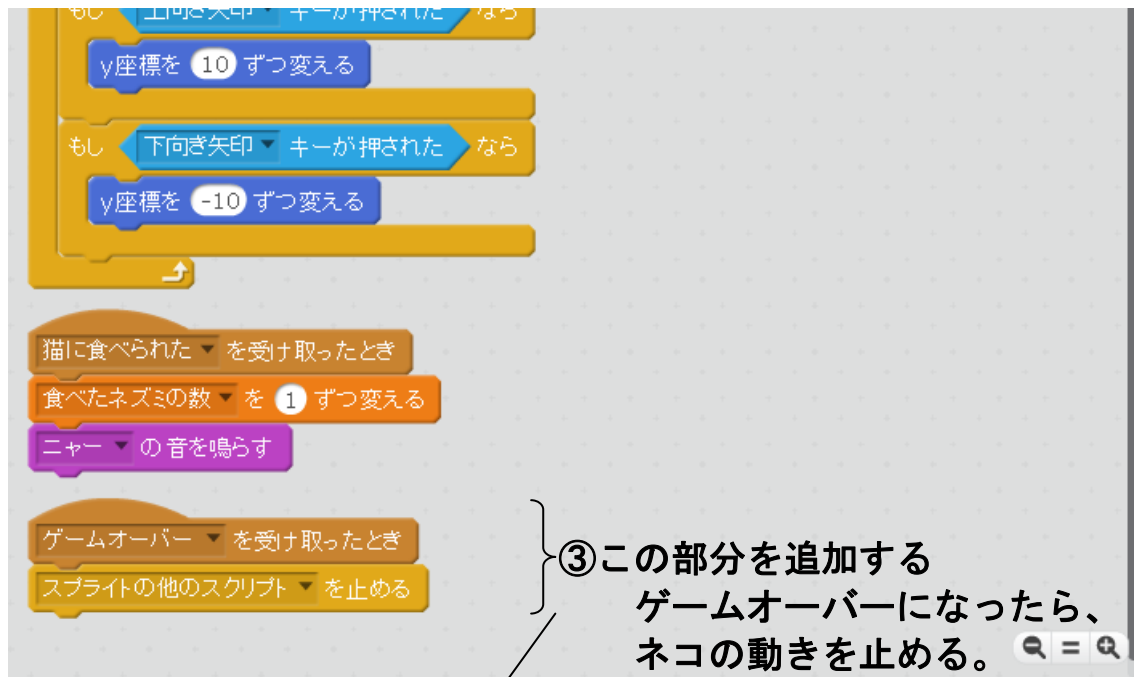


⑩ネズミを取り逃がしたらゲームオーバーにしよう

いよいよ完成です。ネズミが画面の左端に来たら、ゲームを止めるようにします。

ネコ（スプライト1）

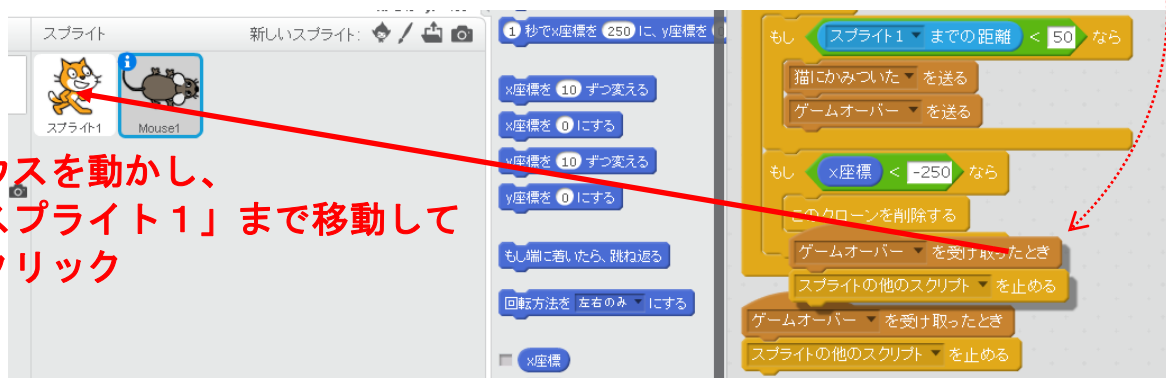
＜次のページを先にやってからこのページに戻ります＞



※③は、ネズミのほうで作った②をコピーしてくると便利です。



③マウスを動かし、
「スプライト1」まで移動して
左クリック



④ネコ（スプライト1）のスクリプトエリアに移動して、
ブロックのレイアウトを調整する
（スクリプトエリアの最初の部分にコピーされています）

ネズミ (Mouse 1)

0.5 秒待つ

クローンされたとき

表示する

x座標を 250、y座標を -150 から 150 までの乱数 にする

-90 度に向ける

ずっと

10 歩動かす

0.05 秒待つ

クローンされたとき

ずっと

もし スプライト 1 までの距離 < 50 なら

猫に食べられた を送る

このクローンを削除する

もし x座標 < -250 なら

ゲームオーバー を送る

①この部分を追加する
左端まで来たら、
ゲームオーバーを宣言する。

ゲームオーバー を受け取ったとき

スプライトの他のスクリプト を止める

②この部分を追加する
ゲームオーバーになったら、
ネズミの動きを止める。

1匹でも取り逃がすとゲームオーバーです。簡単なゲームですが、なかなかスリルがありますね。

これでいちおう完成しましたが、少しだけ手を加えて、さらに楽しんでみましょう。

⑰ネズミから逃げるゲームにしてみましょう

☞ネズミが苦手なネコもいるかもしれません。ネズミを取り逃がしたときではなく、ネコがネズミに捕まったら、ゲームを止めるようにします。

はじめに、変更前のプログラムを保存しておきましょう。

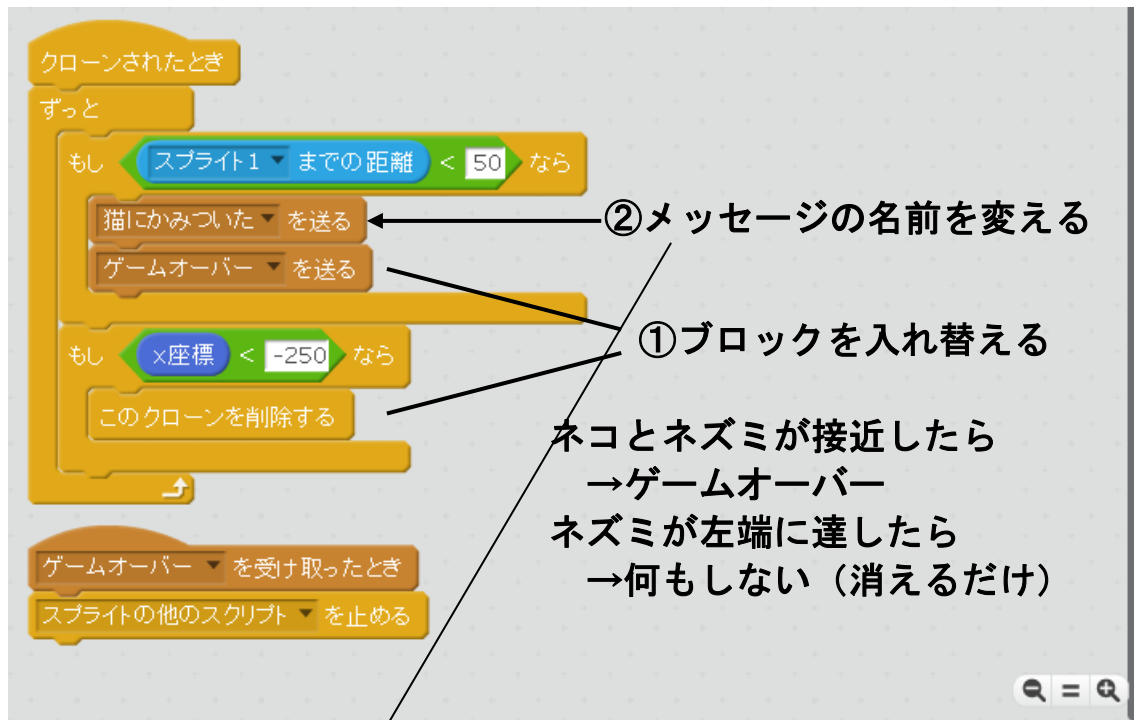
一旦、scratch を終了して、プログラムを右クリックして「コピー」、「貼り付け」で。

保存し終わったら…

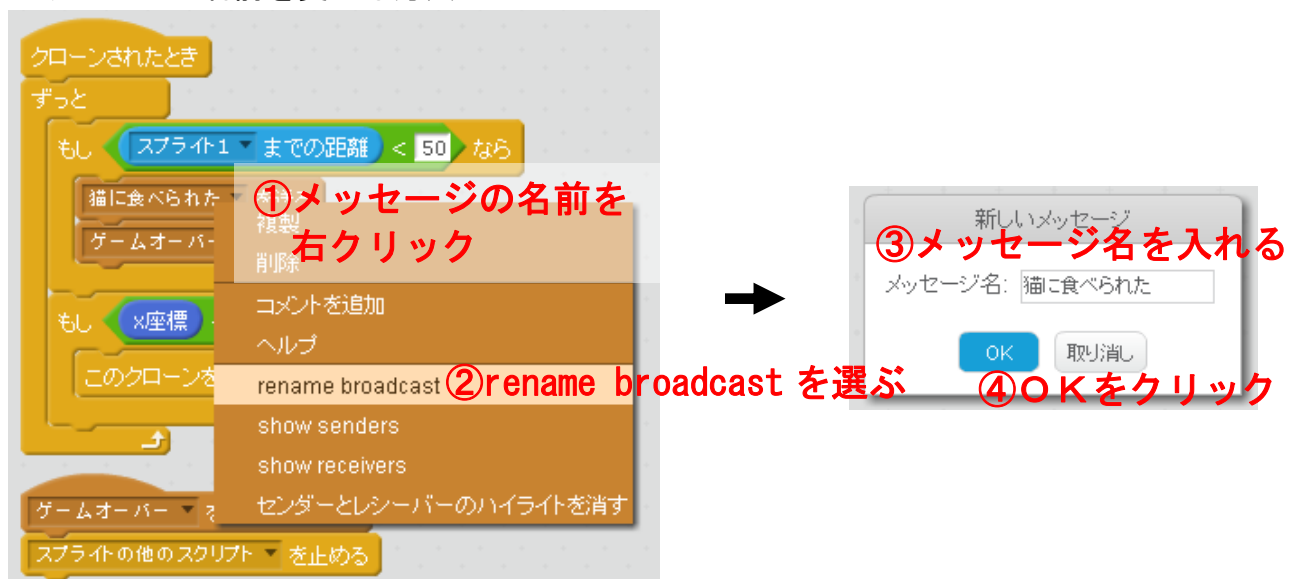
ネコ（スプライト1）

変更はありません。

ネズミ (Mouse 1)



※メッセージの名前を変える方法



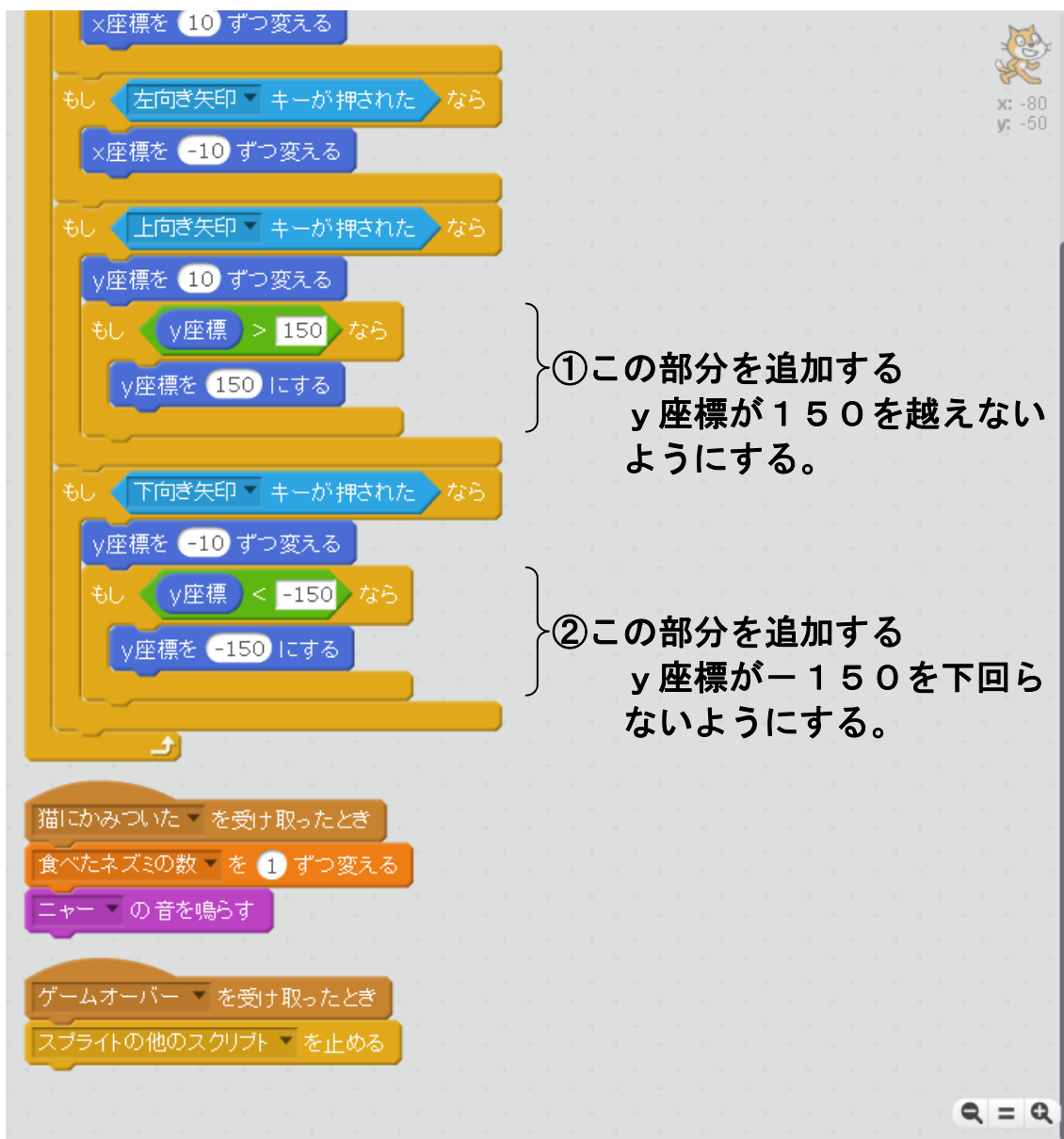
ネズミはこれまでと同じように登場します。実際にゲームをやってみると、少し変ですね。

- ・すべてのネズミをよけるのは難しい。 →どんなにがんばってもゲームオーバー。。
- ・画面の上下どちらかの端に逃げれば、絶対にネズミに捕まらない。 →つまらない裏技。。

設定を調整してみましょう。

⑩設定を調整しよう

👉設定を調整します。ネコが動ける範囲を制限して、それから、当たり判定を厳しくします。
ネコ（スプライト1）



The image shows a Scratch script for a cat sprite. The script is organized into three main sections. The first section, highlighted by a bracket and labeled ①, contains movement logic for the left, up, and down arrow keys. The second section, highlighted by a bracket and labeled ②, contains boundary checks for the y-coordinate. The third section contains logic for when the cat catches a mouse and when the game is over.

**①この部分を追加する
y座標が150を越えない
ようにする。**

**②この部分を追加する
y座標が-150を下回ら
ないようにする。**

Script details:

- Initial position: x座標を 10 ずつ変える
- Left arrow key pressed: x座標を -10 ずつ変える
- Up arrow key pressed: y座標を 10 ずつ変える
- Up arrow key pressed and y座標 > 150: y座標を 150 にする
- Down arrow key pressed: y座標を -10 ずつ変える
- Down arrow key pressed and y座標 < -150: y座標を -150 にする
- When cat catches mouse: 食べたネズミの数 を 1 ずつ変える, ニャー の音を鳴らす
- When game over: スプライトの他のスクリプト を止める

ネズミ (Mouse 1)

クローンされたとき

表示する

x座標を 250、y座標を -150 から 150 までの乱数 にする

-90 度に向ける

ずっと

10 歩動かす

0.05 秒待つ

クローンされたとき

ずっと

もし スプライト1 までの距離 < 25 なら

猫にかみついた を送る

ゲームオーバー を送る

もし x座標 < -250 なら

このクローンを削除する

ゲームオーバー を受け取ったとき

スプライトの他のスクリプト を止める

③当たり判定を厳しくする。
距離が25未満にならないと捕まらない。

これで普通の難易度のゲームになったと思います。

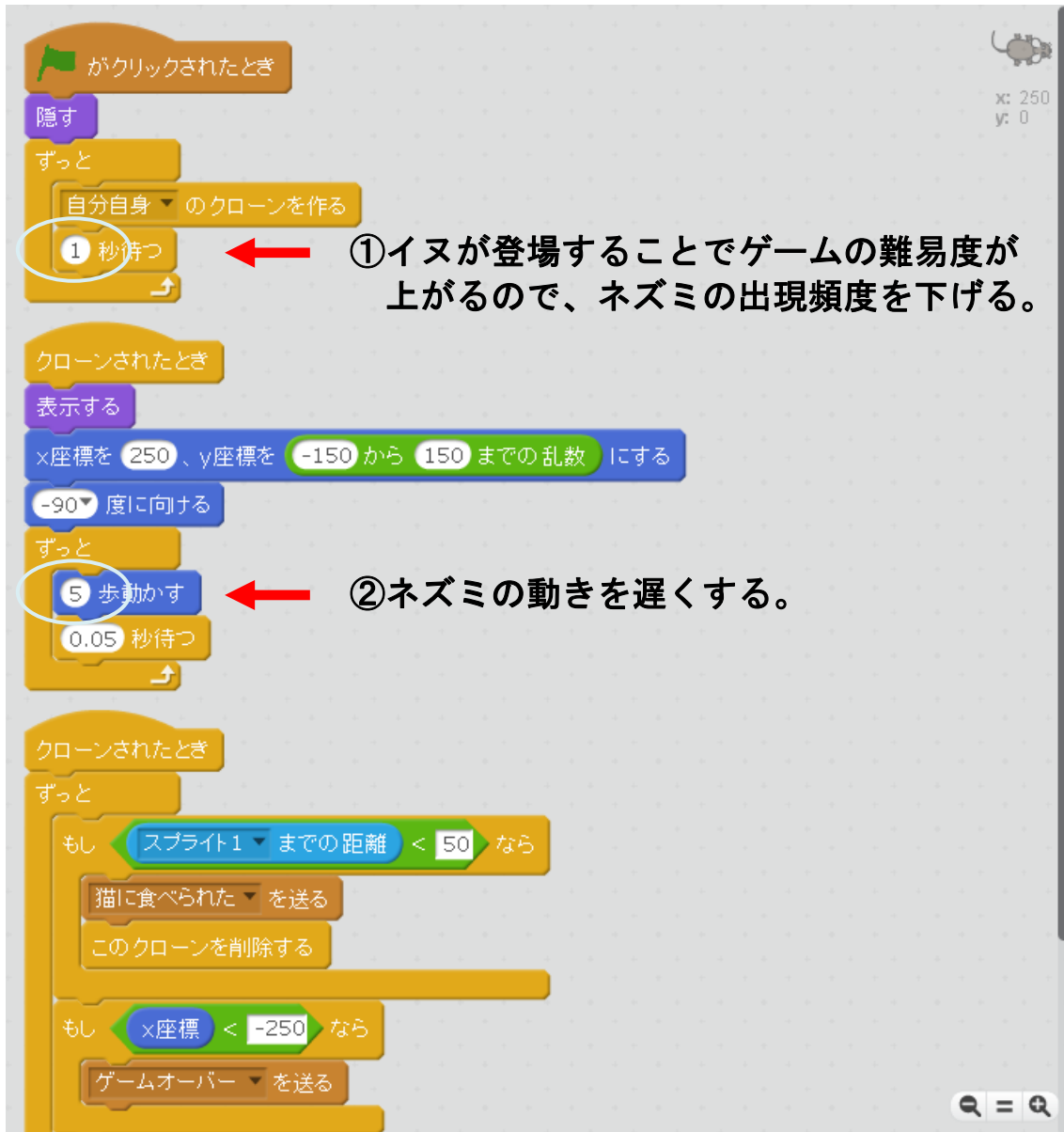
⑪イヌをよけながらネズミを捕まえるゲームにしよう

⑩で書いたプログラムを参考に、イヌをよけながらネズミを捕まえるゲームを作りましょう。
ネコ（スプライト1）

⑬に戻します。

ネズミ（Mouse1）

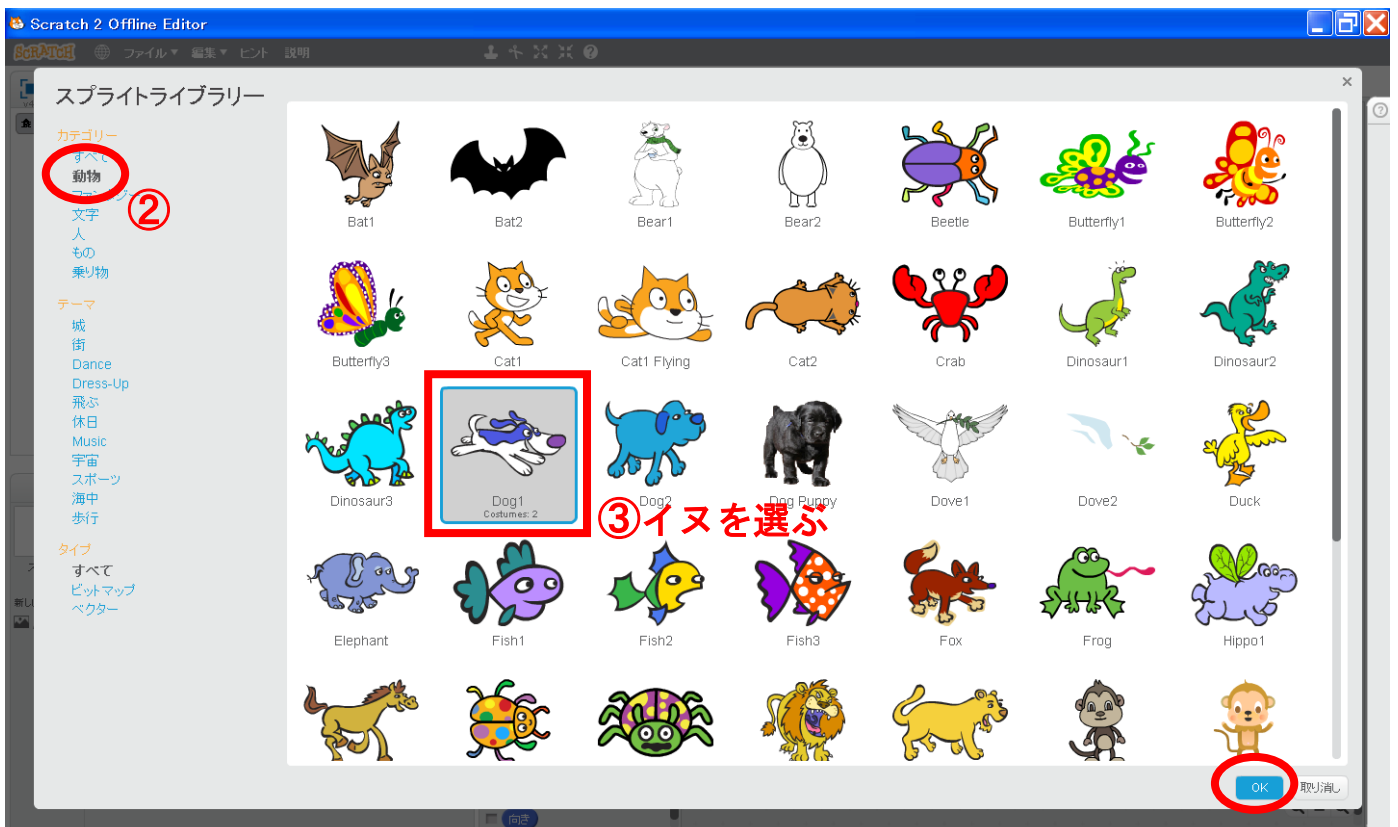
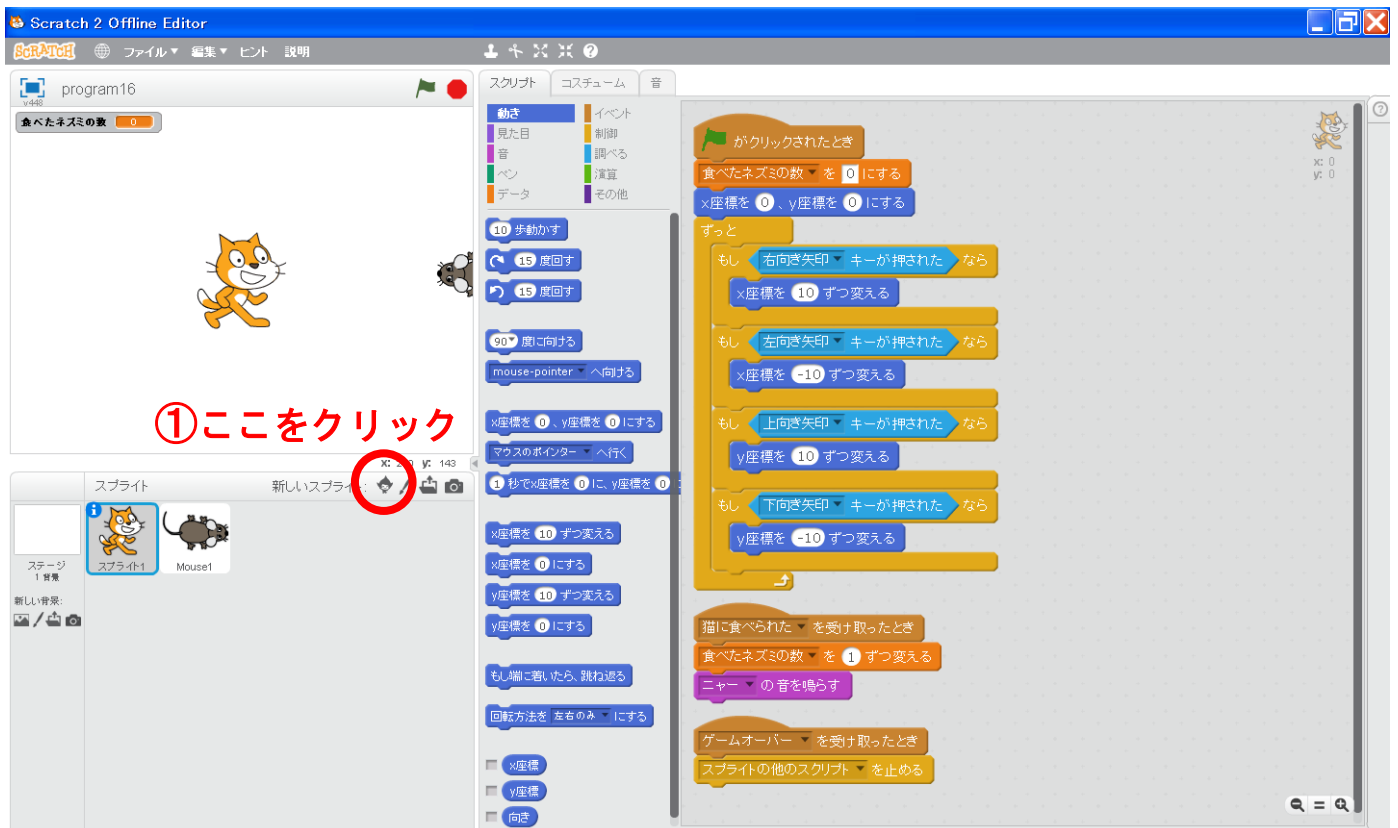
⑬に戻した後で、少しだけ変更します。

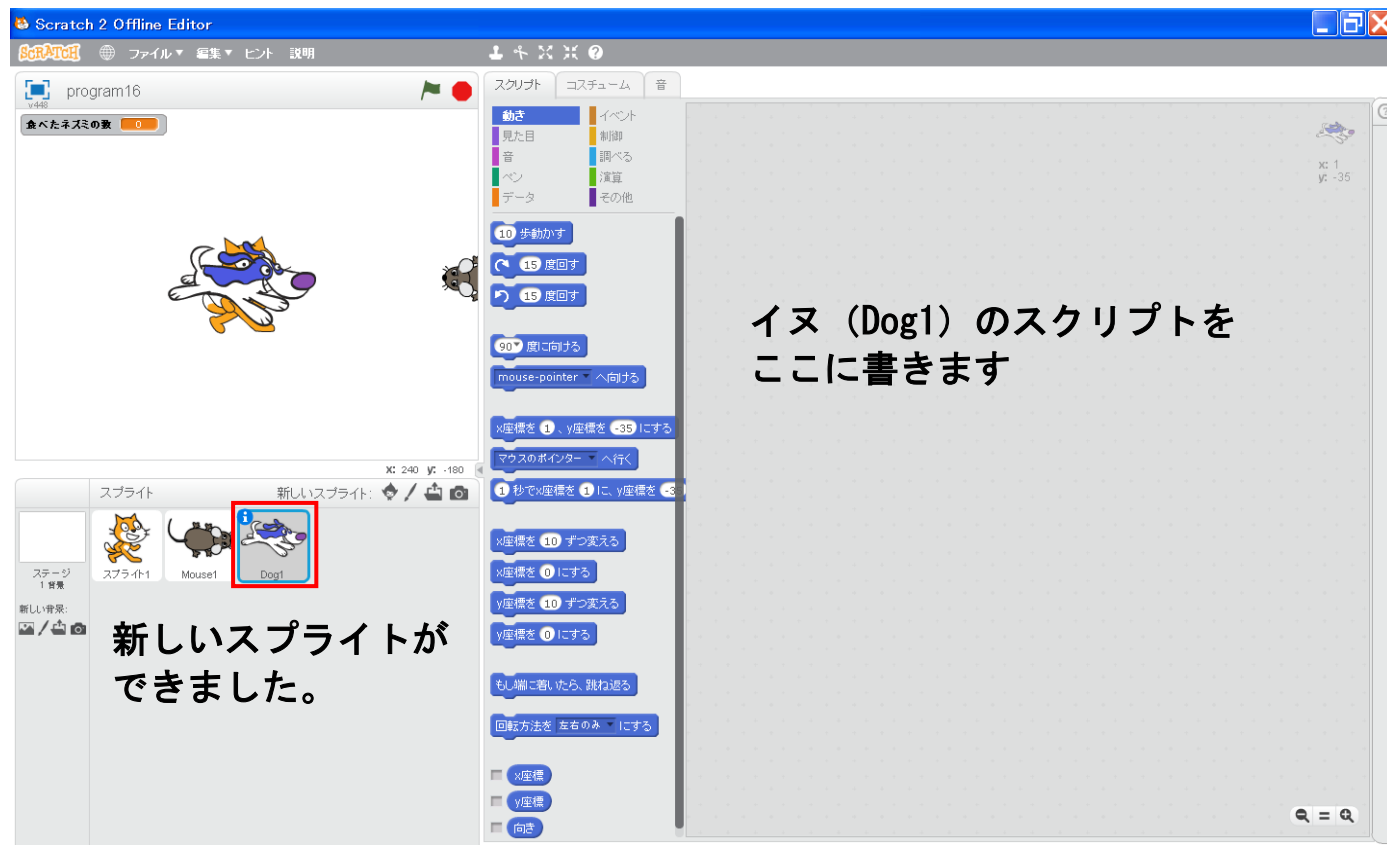


イヌ (Dog 1)

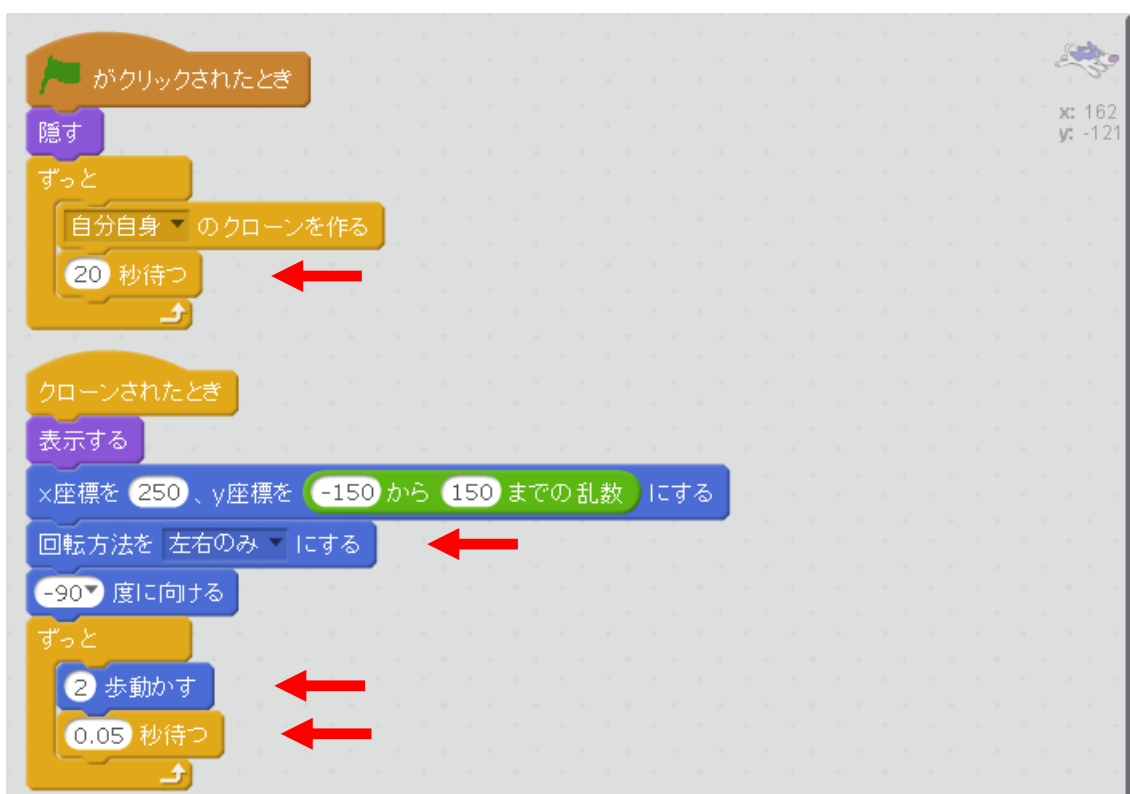
イヌを登場させます。登場人物（スプライト）を追加し、イヌのスクリプトを書きます。

登場人物（スプライト）を追加します。

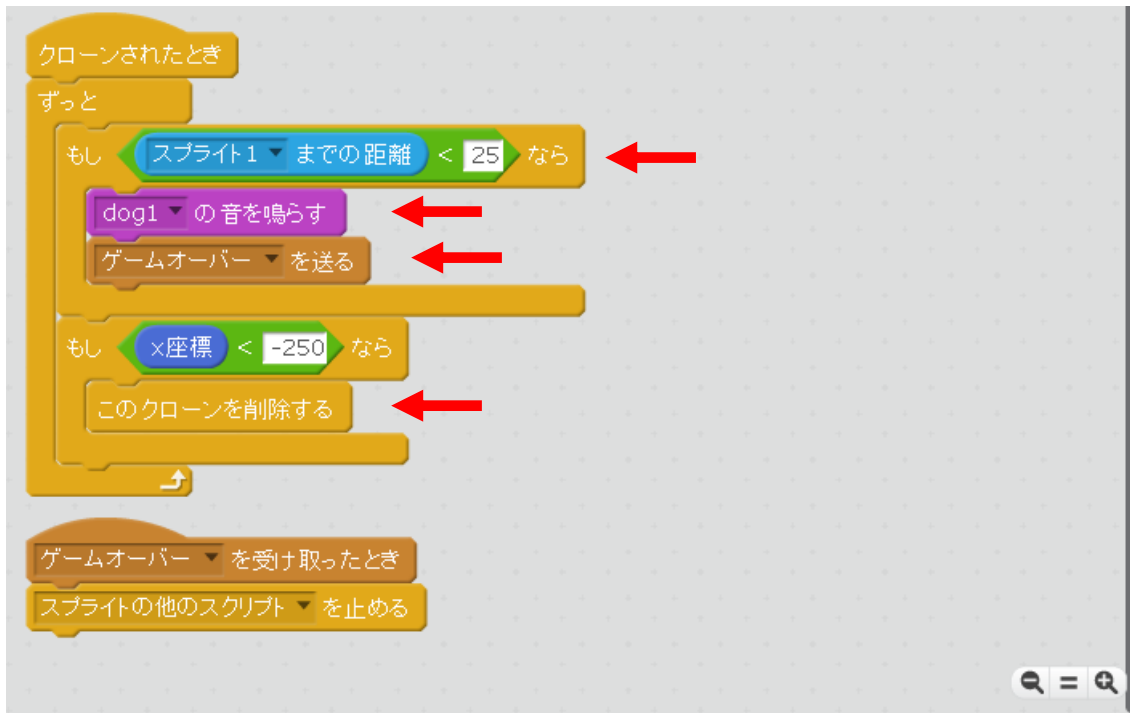




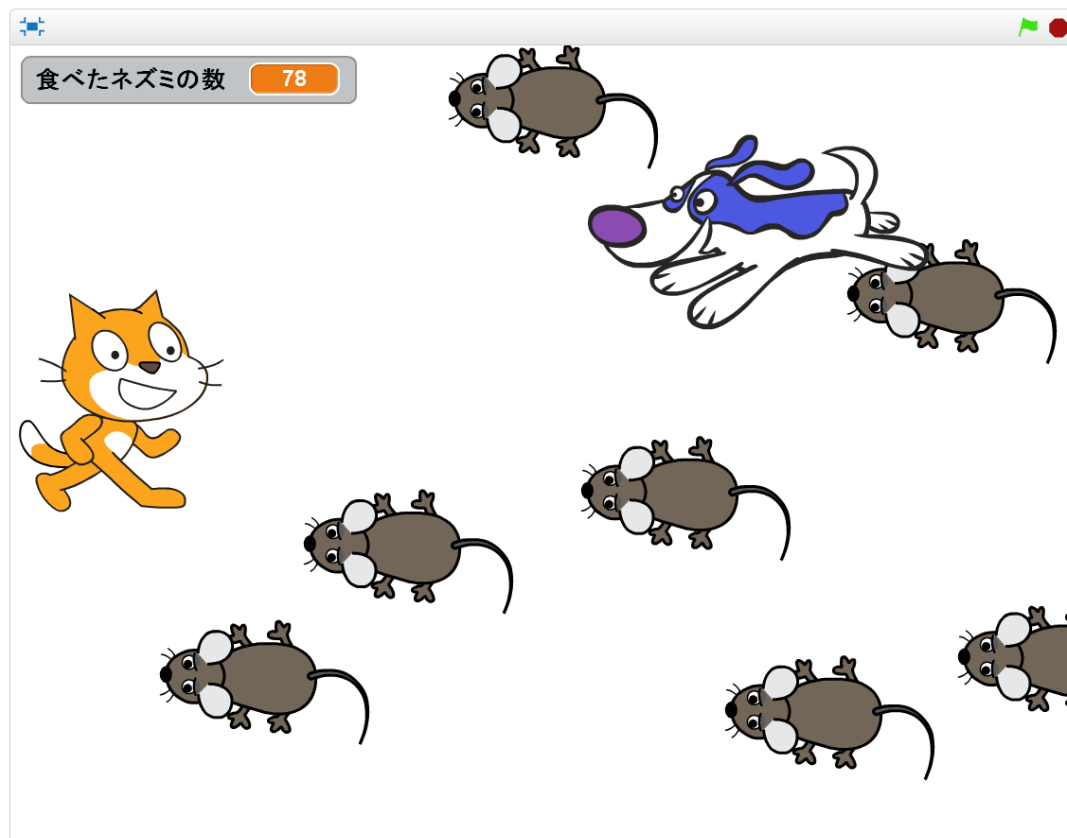
スクリプトの内容は、ネズミと基本的には同じですが、**赤矢印**で示した場所だけ変更します。



次のページに続く



実行例



いかがですか。少し工夫するだけで高度で面白いゲームになりましたね。自分で手を加えてさらにおもしろくしてもよいし、新しいゲームを考えてもよいでしょう。

参考までに、この講座で作ってもらったものとは少し違ったゲームの作成例も配布しておきます。ゲーム作りやプログラミングを楽しんでください。