

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 基本操作と例題

令和3年7月22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¥man¥フォルダ

Scratchとは

米国MIT(マサチューセッツ工科大学)メディアラボが開発した、子供向けのプログラム開発環境(IT用語で「Scratch」とは、プログラムを一から作成すること)。
公立小学校では2020年から(新型コロナ感染拡大より2021年より本格化)、中学校では2021年からプログラミング教育が必修化されることとなり、巷で子供向けプログラミング教室も開設されている。これらの学校で「Scratch」の採用が増えている。

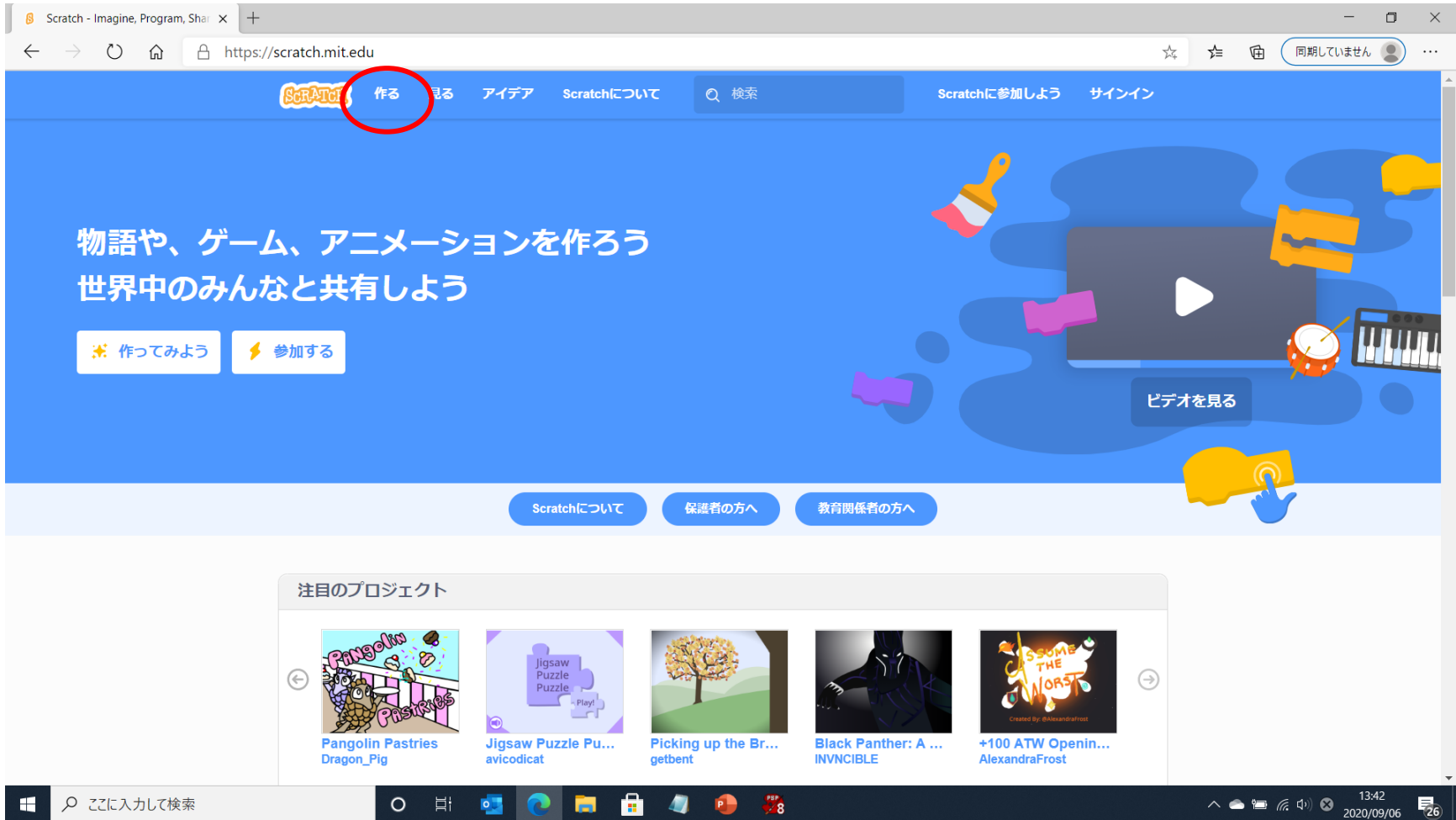
[特徴]

- (1) Webブラウザが動く各種パソコン(Windows/Mac)、タブレット(iPad/Android)を用いて、インストールせずにオンライン環境で無償で利用できる。
(インストールしてオフライン使用も可)。
- (2) ブロックとよばれるソフトウェア部品を画面上のドラッグ操作により、組合わせながらプログラミングできる(一部、数値・文字のキーボード入力が必要)。
- (3) 変数名などプログラム内で使用する言語は英語に限らず、日本語など数十カ国の言語を使用できる。
- (4) SNS機能を備えていて、作成したプログラムを友達と共有できる。
- (5) プログラミング言語で代表的なC言語を意識したソフトウェア部品が多数用意されており、C言語の関数(組込み関数)に相当する独自ブロックを定義することにより、子供向けでありながら高度なアプリケーションを開発できる。
→変数・関数の概念を子供に理解させる必要があり、教育は容易ではない。

1. Web版「Scratch」ソフトウェアの起動方法

Web版Scratch起動(クリック)

Webブラウザ(下記はEdgeの例)でアドレスバーに、URL” <https://scratch.mit.edu/>”を入力。上部メニューで「作る」をクリック(サインインは必須ではない)(参考:インストール版を使用する場合は、”<https://scratch.mit.edu/download/>”よりダウンロード)



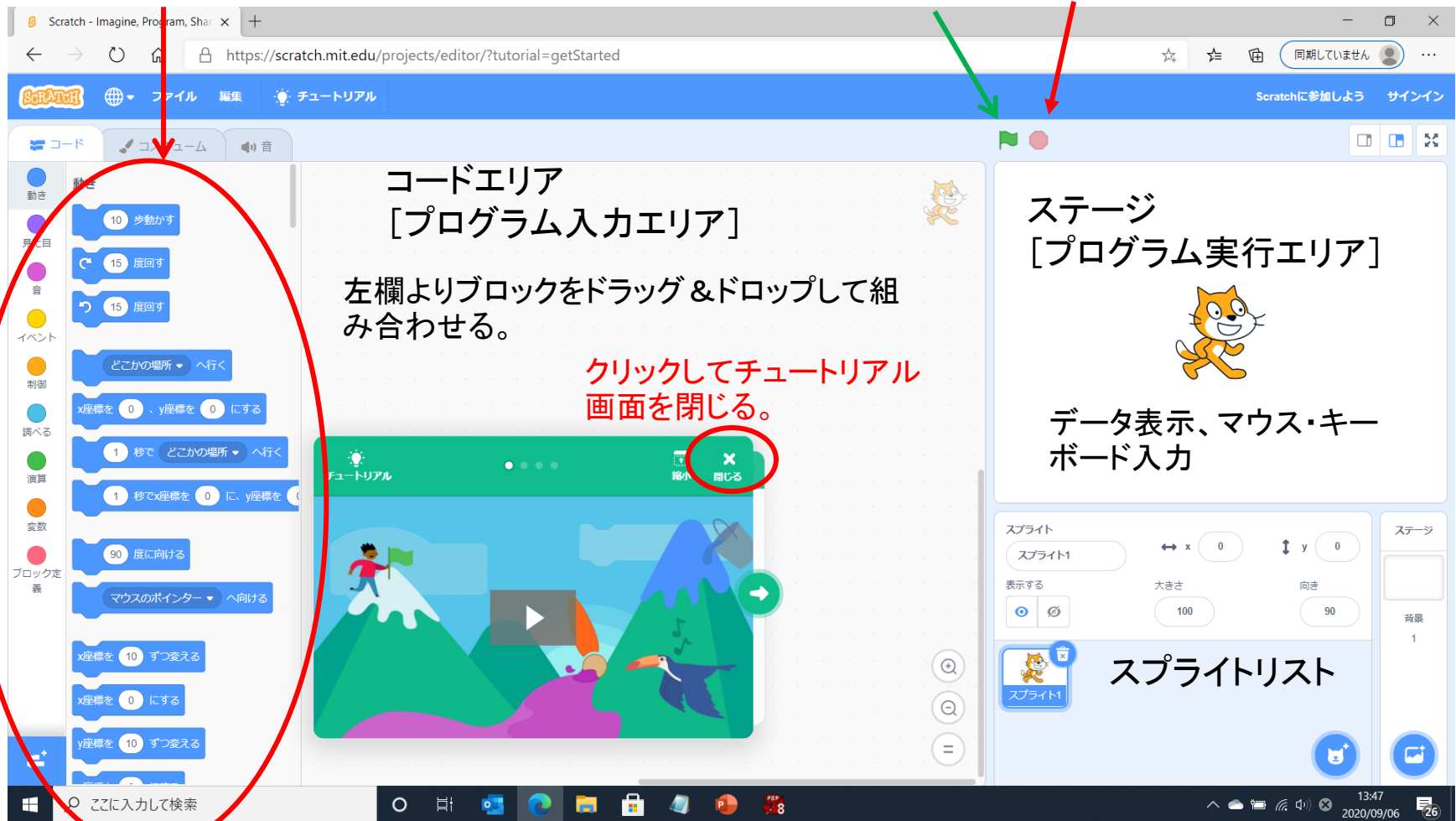
2. プログラム開発環境の概要(1): 画面構成

自動起動されるチュートリアル画面を閉じて、プログラム入力画面を準備します。
以下、分割画面の機能を把握しておいてください。

ブロックパレット
[プログラミング用のブロック・部品のメニュー]

プログラムの
起動

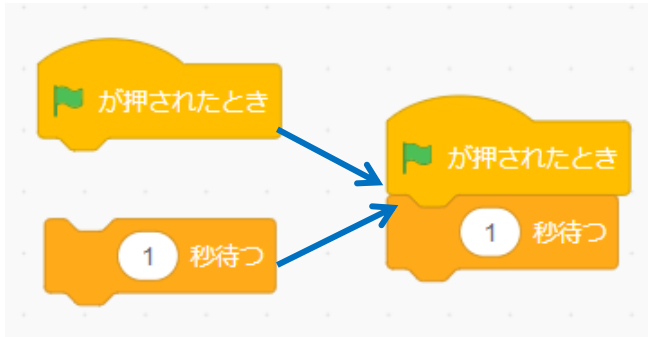
プログラムの
停止



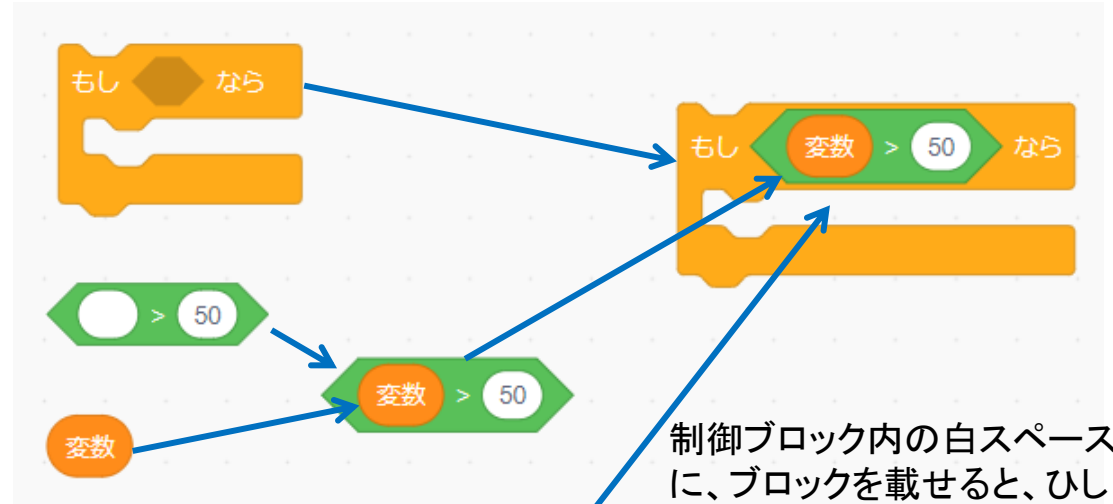
2. プログラム開発環境の概要(2):ブロック・部品の連結規則

プログラミング用ブロック・部品の形状が同じであれば、ドラッグ操作により連結・合成することができます。以下、典型的な形状例を示します。

[ブロックの縦連結]

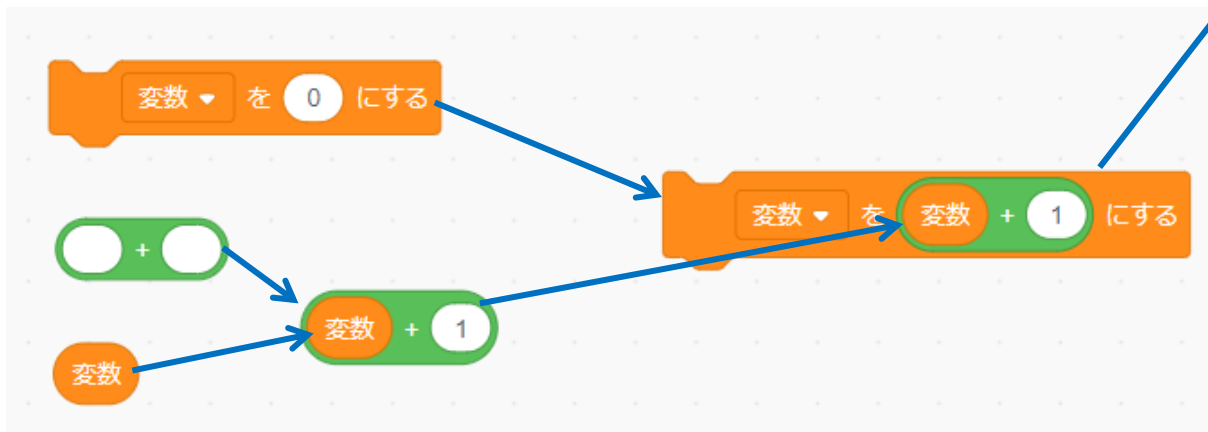


[ひし形ブロック(条件)の連結]



制御ブロック内の白スペースに、ブロックを載せると、ひし形条件が満たされた場合に実行される。

[楕円ブロック(変数、演算)の連結]



白楕円には、数値・文字を直接入力したり、変数・演算子のブロックを載せることができる。

3. プログラミング(1): キャラクターに「こんにちは」と喋らせる(1)

はじめに、左欄のコードのメニューより「イベント」を選択し、「(旗)が押されたとき」を中央のプログラム入力エリアにドラッグ & ドロップして配置します。プログラムの第1ブロックは原則として、「(旗)が押されたとき」ブロックで始めます。

The screenshot shows the Scratch web interface. On the left, the 'Code' menu is selected, and the 'Events' category is highlighted. A red circle is drawn around the 'When green flag clicked' block in the 'Events' menu. A red arrow points from this block to the 'When green flag clicked' block in the workspace. Another red circle is drawn around the 'Events' category icon in the left sidebar. A text box with the Japanese text 'マウスの左ボタンを押しながら移動させる。' (Move while pressing the left mouse button.) is overlaid on the workspace. The workspace contains the 'When green flag clicked' block. The right panel shows the 'Sprites' area with the Scratch cat sprite selected. The bottom status bar shows the date and time: 2020/09/06 14:06.

Scratch - Imagine, Program, Share

https://scratch.mit.edu/projects/editor/?tutorial=getStarted

Scratch

ファイル 編集 チュートリアル

Scratchに参加しよう サインイン

コード コスチューム 音

イベント

動き

見た目

音

イベント

制御

調べる

演算

変数

ブロック定義

このスプライトが押されたとき

メッセージ1を受け取ったとき

メッセージ1を送る

メッセージ1を送って待つ

1秒待つ

10回繰り返す

マウスの左ボタンを押しながら移動させる。

スプライト

スプライト1

表示する

大きさ 100

向き 90

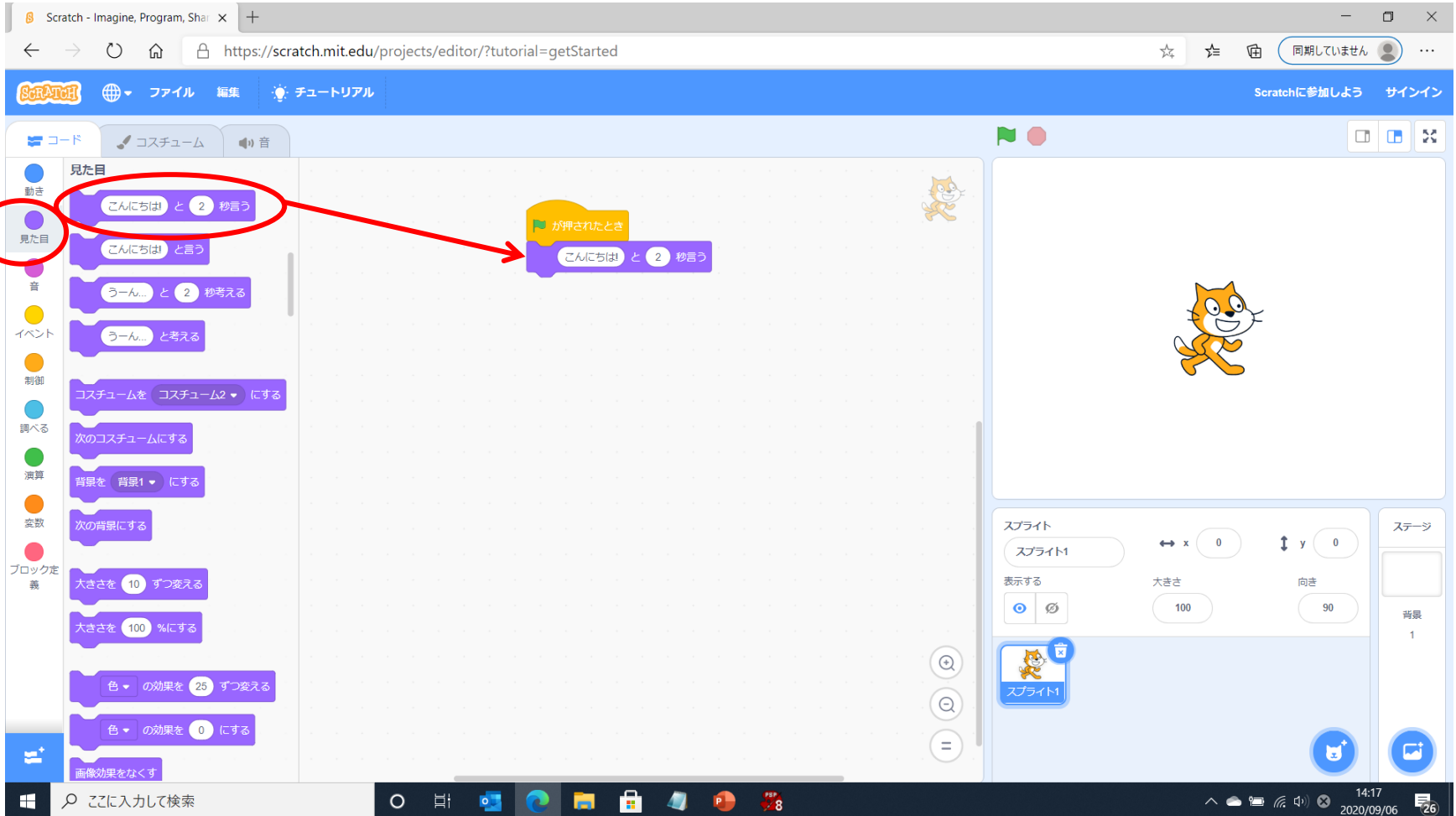
ステージ

背景 1

14:06 2020/09/06

3. プログラミング(1): キャラクターに「こんにちは」と喋らせる(2)

左欄のコードのメニューより「見た目」を選択し、「(こんにちは!)と(2)秒言う」を中央のプログラム入力エリアに配置されている「(旗)が押されたとき」ブロックの直下にドラッグ&ドロップして配置し、上のブロックと結合させます(凹凸部を重ねると結合します)。



3. プログラミング(1):キャラクターに「こんにちは」と喋らせる(3):実行

プログラムを実行させる場合は、プログラム実行エリアの左上の「緑色の旗」をクリックします。実行中のプログラムを停止させる場合は、右横の「赤丸」をクリックします。

プログラム
の起動

プログラムの
停止

[プログラム実行エリア]

こんにちは!

実行結果

Scratch - Imagine, Program, Share

https://scratch.mit.edu/projects/editor/?tutorial=getStarted

Scratch

ファイル 編集 チュートリアル

Scratchに参加しよう サインイン

コード コスチューム 音

見た目

動き

見た目

音

イベント

制御

調べる

演算

変数

ブロック定義

見た目

こんにちは! と 2 秒言う

こんにちは! と言う

うーん... と 2 秒考える

うーん... と考える

コスチュームを コスチューム2 にする

次のコスチュームにする

背景を 背景1 にする

次の背景にする

大きさを 10 ずつ変える

大きさを 100 %にする

色 の効果を 25 ずつ変える

色 の効果を 0 にする

画像効果をなくす

が押されたとき

こんにちは! と 2 秒言う

スプライト

スプライト1

表示する

大きさ

向き

ステージ

背景

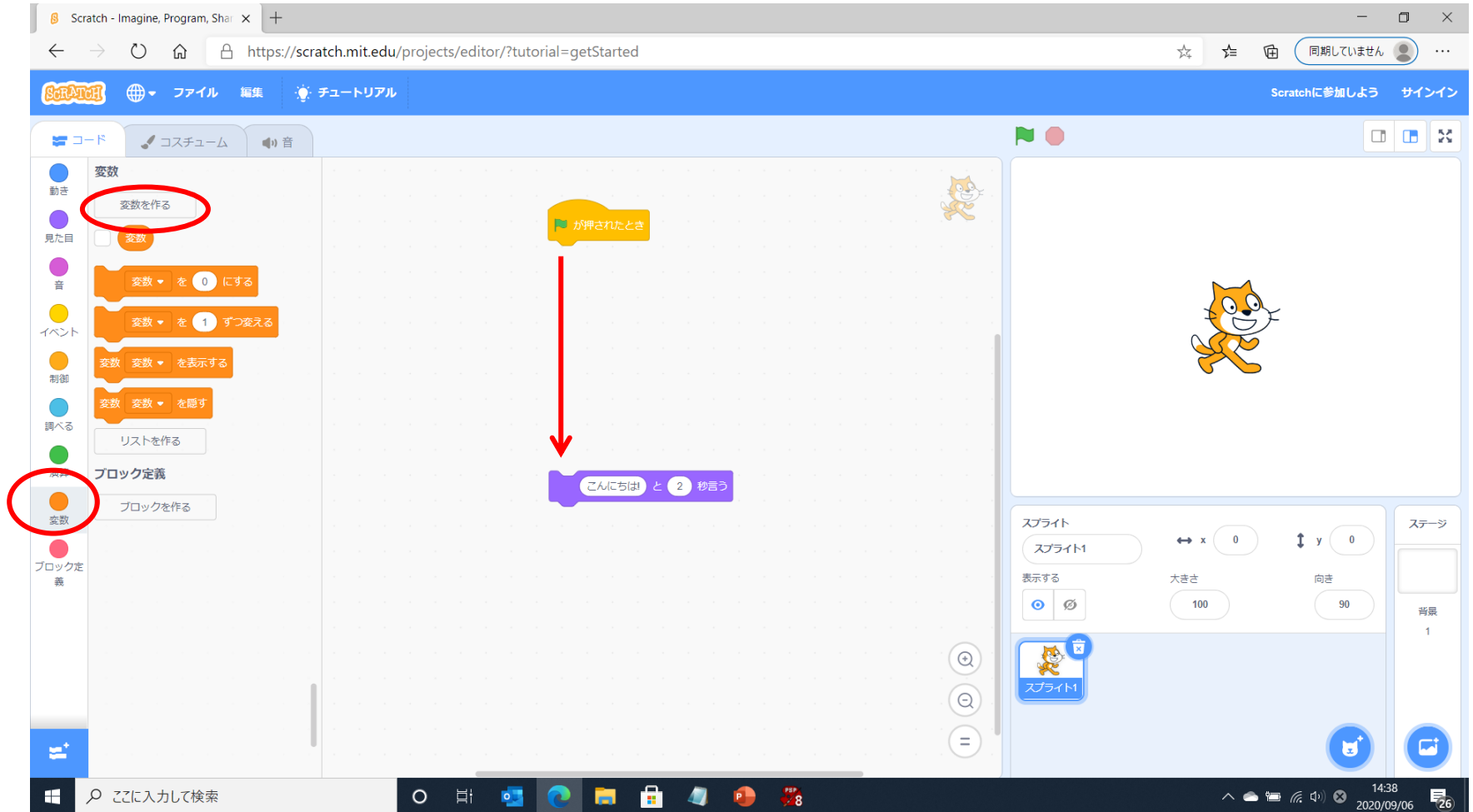
1

ここに入力して検索

14:24 2020/09/06

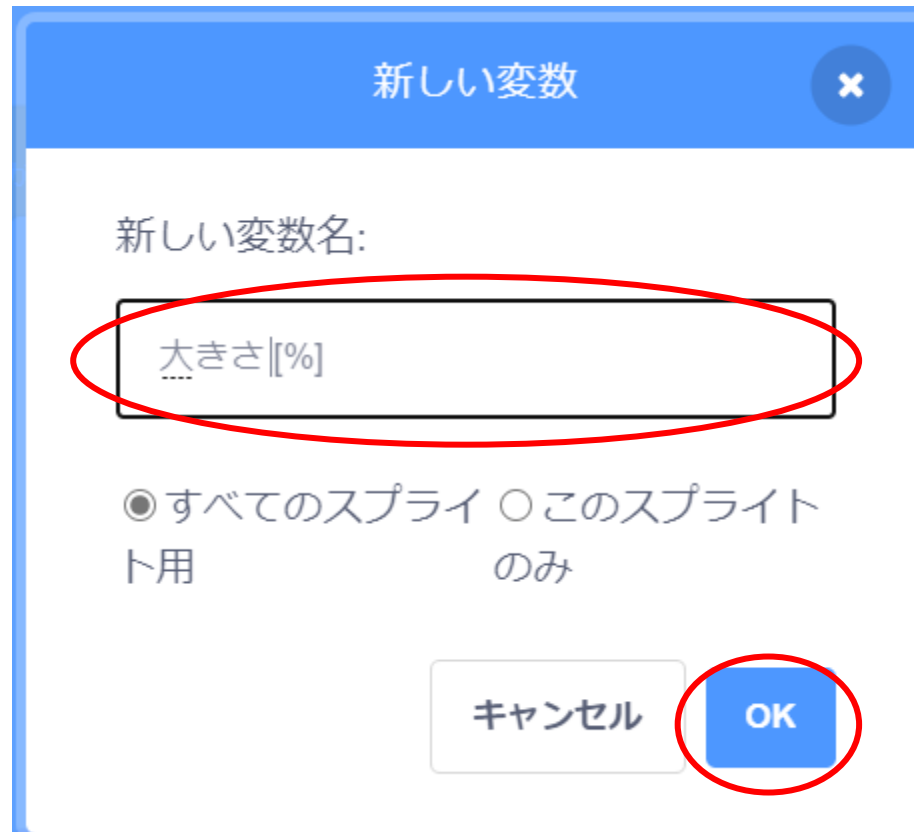
3. プログラミング(2): キャラクターの大きさを変更する(1)

まず、「(こんにちは!)と(2)秒言う」ブロックを下方にドラッグして切り離します。次に、キャラクターの大きさをキーボード入力する変数を作成します。左欄のコードのメニューより「変数」を選択し、「変数を作る」をクリックします。



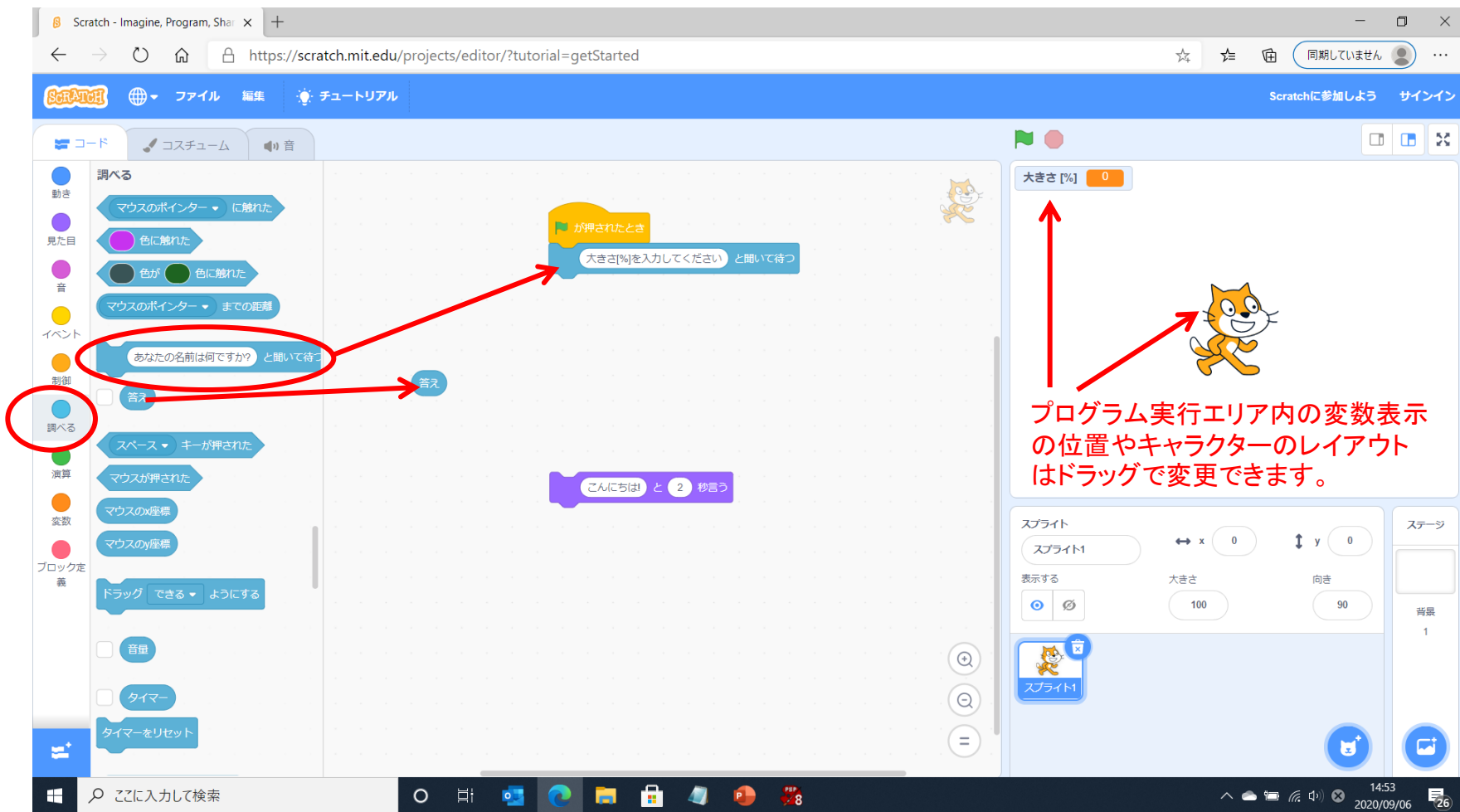
3. プログラミング(2): キャラクターの大きさを変更する(2)

「新しい変数」ダイアログで「新しい変数名:」のテキストボックス内に、例えば、「大きさ[%]」を入力し、「OK」ボタンをクリックして閉じます。



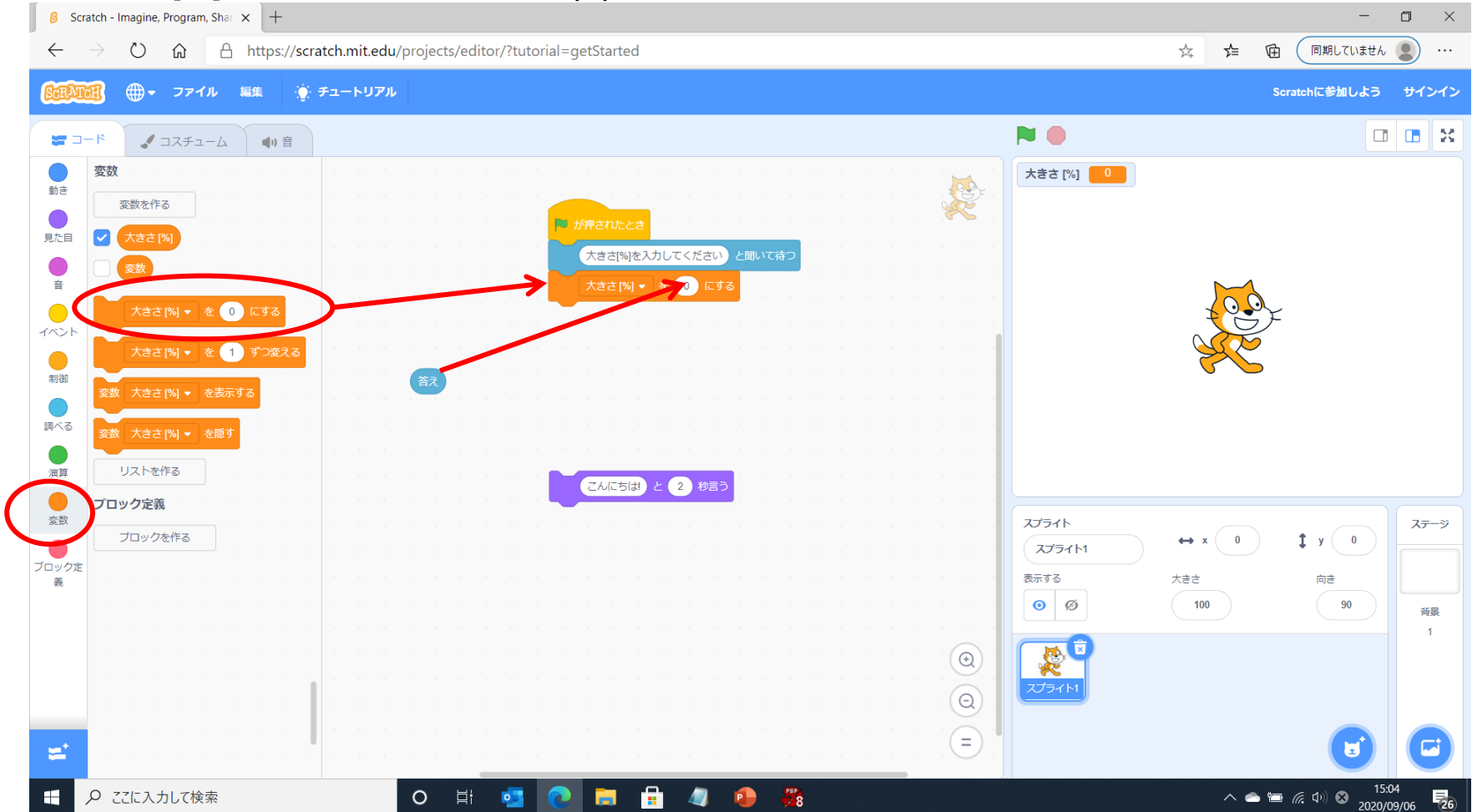
3. プログラミング(2):キャラクターの大きさを変更する(3)

左欄のコードのメニューより「調べる」を選択し、「(あなたの名前は何ですか?)と聞いて待つ」を「(旗)が押されたとき」ブロックの直下にドラッグ & ドロップして配置し、上のブロックと結合させます。質問文の(あなたの名前は何ですか?)を(大きさ[%])を入力してください)に変更します。また、上記メニューの下「答え」ブロックもプログラム入力エリア内の適当な空きスペースにドラッグ & ドロップしておきます。



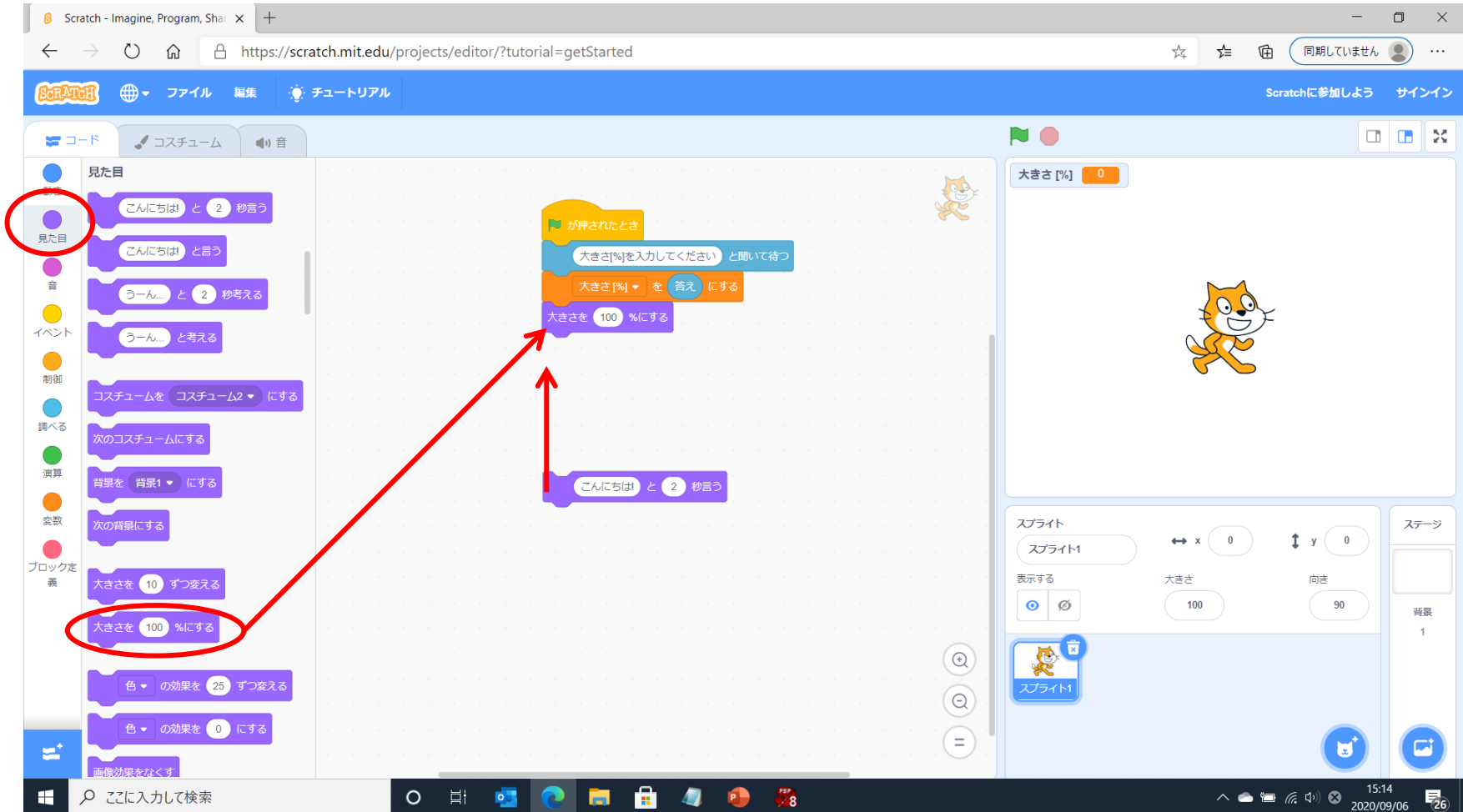
3. プログラミング(2): キャクターの大きさを変更する(4)

左欄のコードのメニューより「変数」を選択し、「大きさ[%]を(0)にする」を「(大きさ[%]を入力してください)と聞いて待つ」ブロックの直下にドラッグ & ドロップして配置し、上のブロックと結合させます。また、プログラム入力エリアに置いていた「答え」ブロックを「大きさ[%]を(0)にする」の白丸(0)の上にドラッグして載せます。



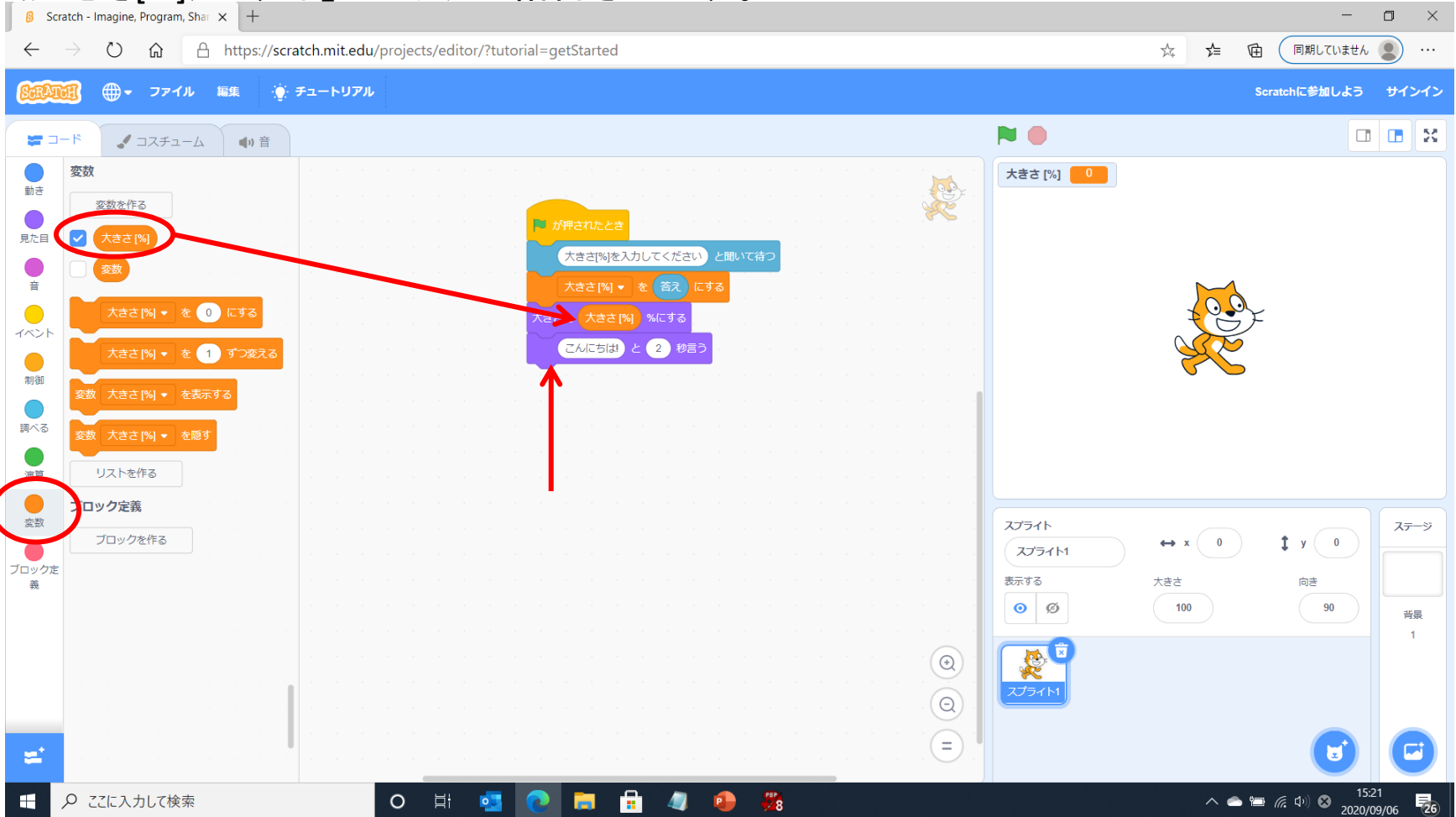
3. プログラミング(2): キャクターの大きさを変更する(5)

左欄のコードのメニューより「見た目」を選択し、「大きさを(100%)にする」を「(大きさ [%]を(答え)にする」ブロックの直下にドラッグ&ドロップして配置し、上のブロックと結合させます。



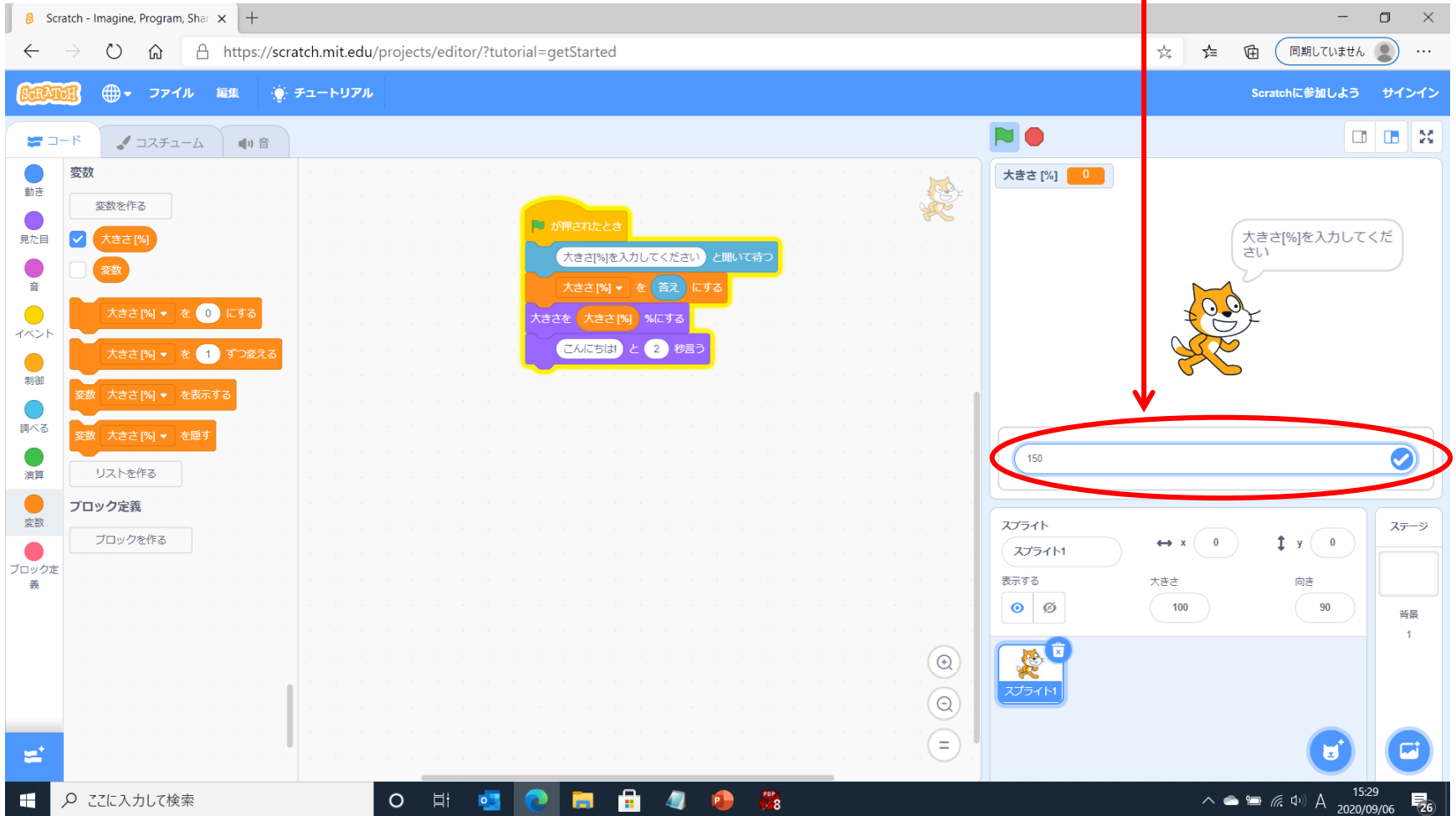
3. プログラミング(2): キャクターの大きさを変更する(5)

左欄のコードのメニューより「変数」を選択し、変数「大きさ[%]」を「大きさを(100%)にする」ブロックの白丸(100%)の上にドラッグ & ドロップして載せます。そして、下方に切り離しておいた「(こんにちは!)と(2)秒言う」ブロックを上方にドラッグして、「大きさを(大きさ[%])にする」ブロックに結合させます。



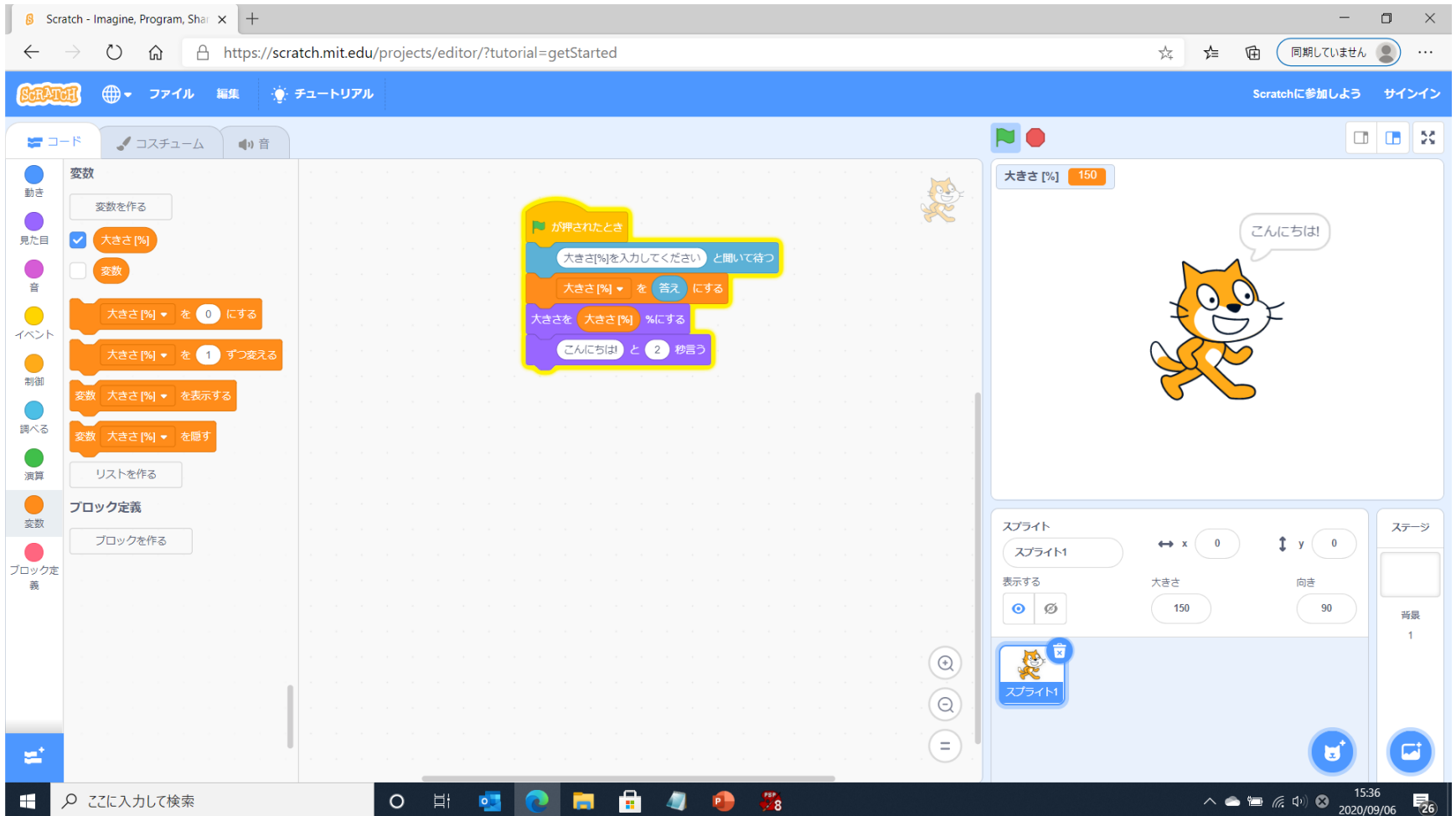
3. プログラミング(2):キャラクターの大きさを変更する(6):実行1

プログラム実行エリアの左上の「緑色の旗」をクリックして実行させますと、プログラム実行エリアのキャラクターの下にテキストボックスが表示されます。例えば、「150」と入力し、エンターキーまたは右端のチェックマークをクリックします。



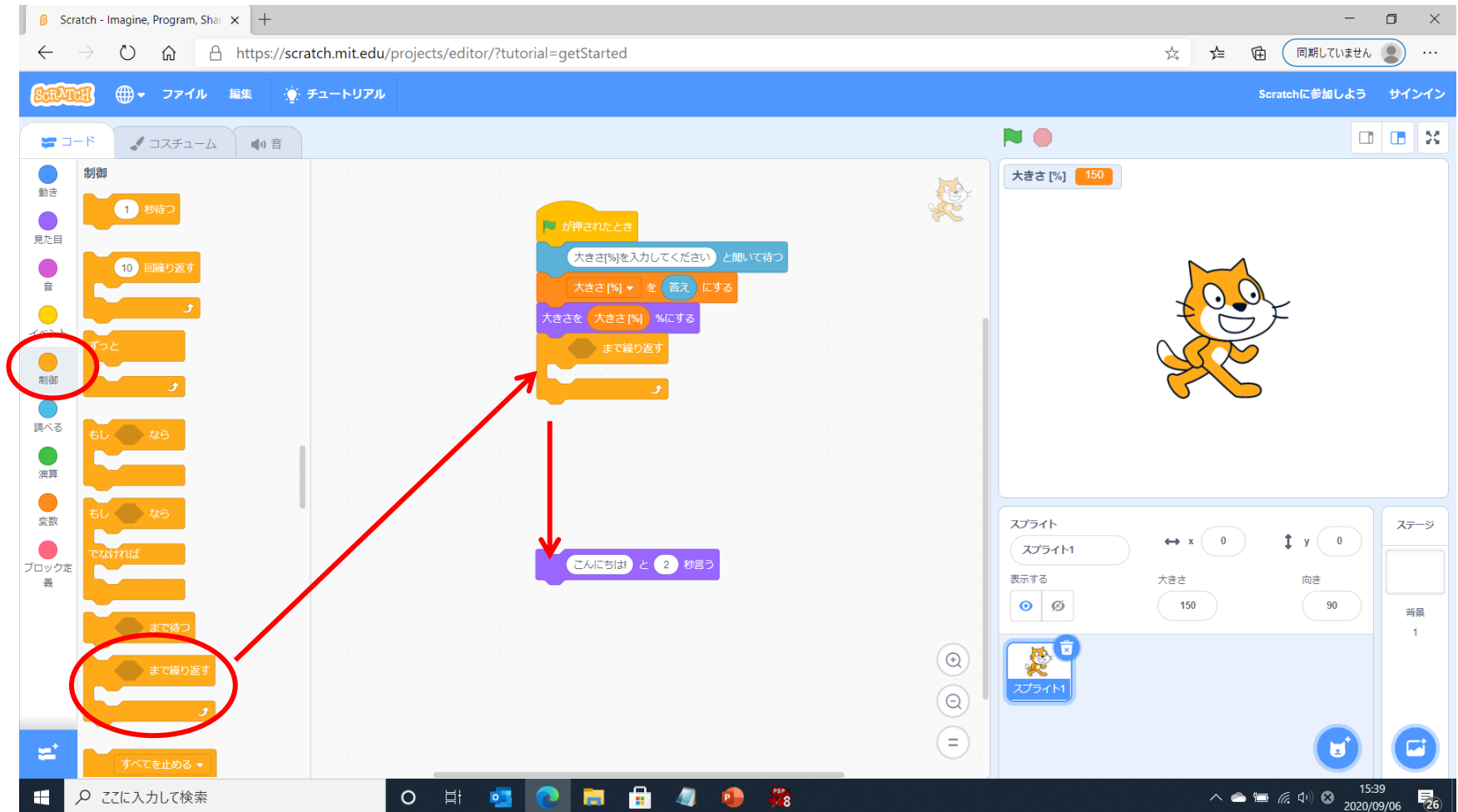
3. プログラミング(2):キャラクターの大きさを変更する(7):実行2

以下のようにキャラクターの大きさが変更されます。



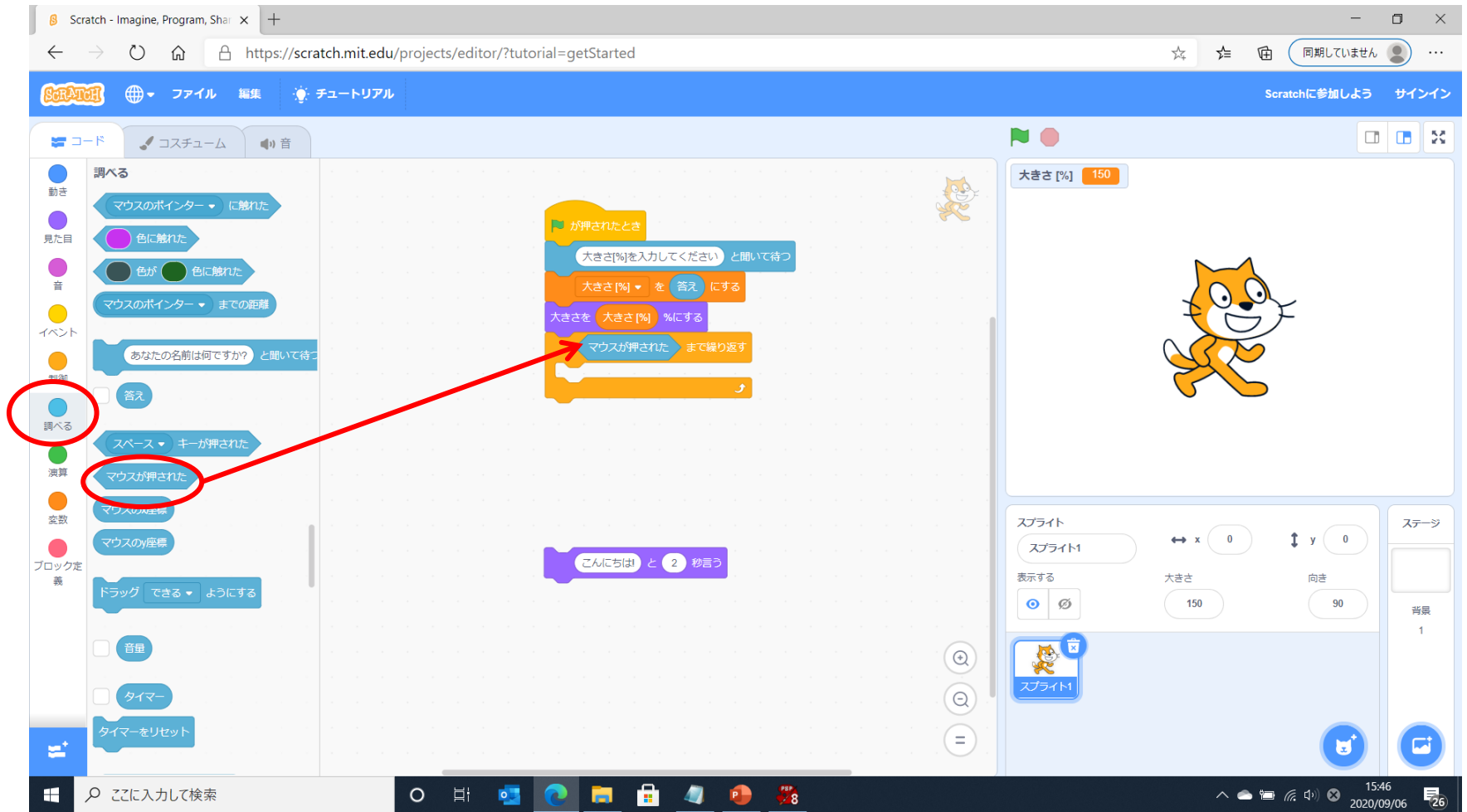
3. プログラミング(3): キャクターの位置をマウスに連動させる(1)

まず、「(こんにちは!)と(2)秒言う」ブロックを下方にドラッグして切り離します。次に、左欄のコードのメニューより「制御」を選択し、「(ひし形)まで繰り返す」を、「大きさを(大きさ[%])にする」ブロックの直下にドラッグ&ドロップして配置し、上のブロックと結合させます。



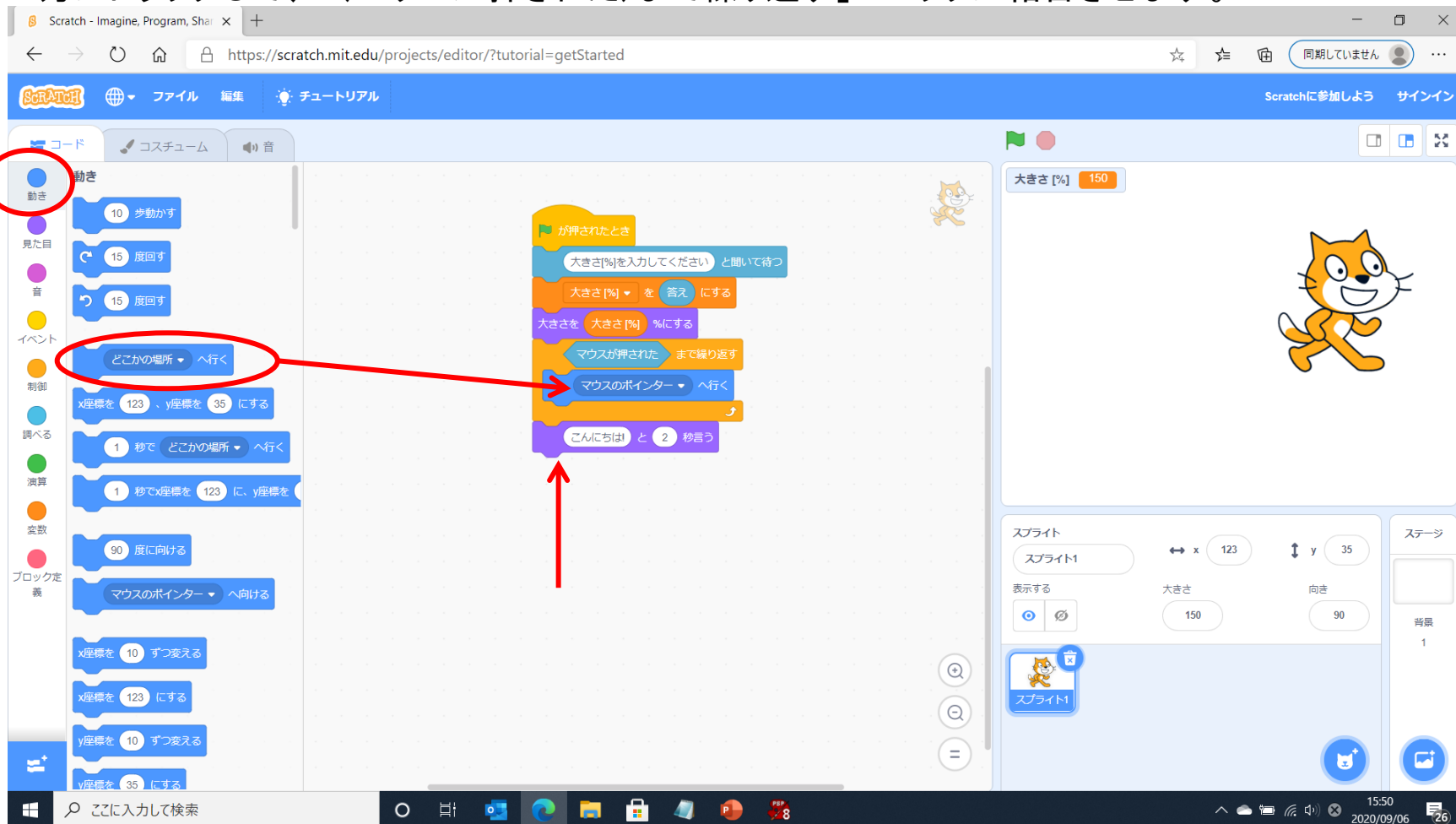
3. プログラミング(3):キャラクターの位置をマウスに連動させる(2)

左欄のコードのメニューより「調べる」を選択し、「マウスが押された」を「(ひし形)まで繰り返す」ブロックの(ひし形)マークの上にドラッグ&ドロップして載せます。



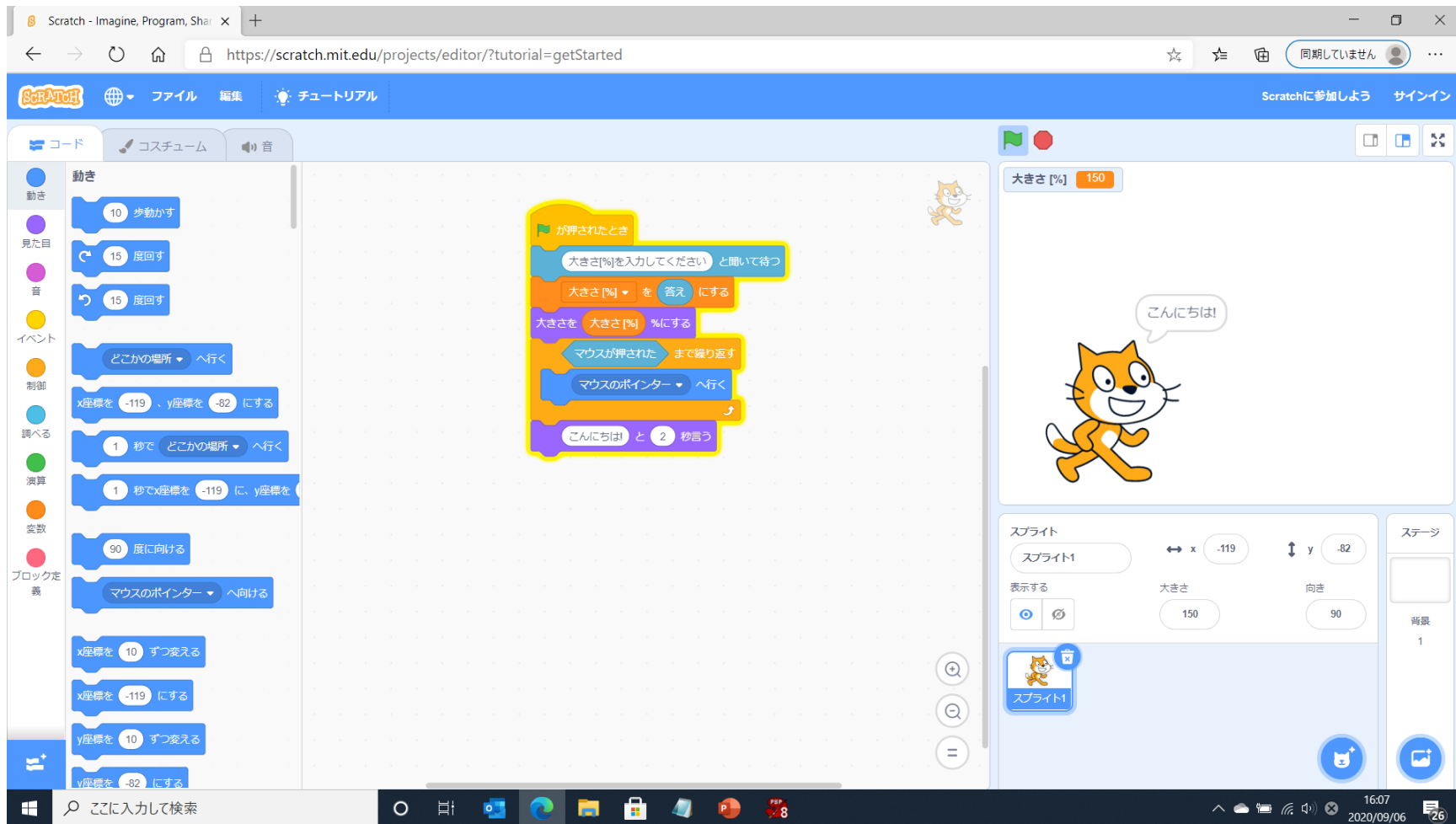
3. プログラミング(3): キャクターの位置をマウスに連動させる(3)

左欄のコードのメニューより「動き」を選択し、変数「(どこかの場所)へ行く」を「(マウスが押された)まで繰り返す」ブロックの中央のスペース上にドラッグ & ドロップして挿入させます。挿入したブロック内の(どこかの場所)の上をクリックしてプルダウンメニューを表示させ、(マウスのポインター)に変更します。そして、下方に切り離しておいた「(こんにちは!)と(2)秒言う」ブロックを上方にドラッグして、「(マウスが押された)まで繰り返す」ブロックに結合させます。



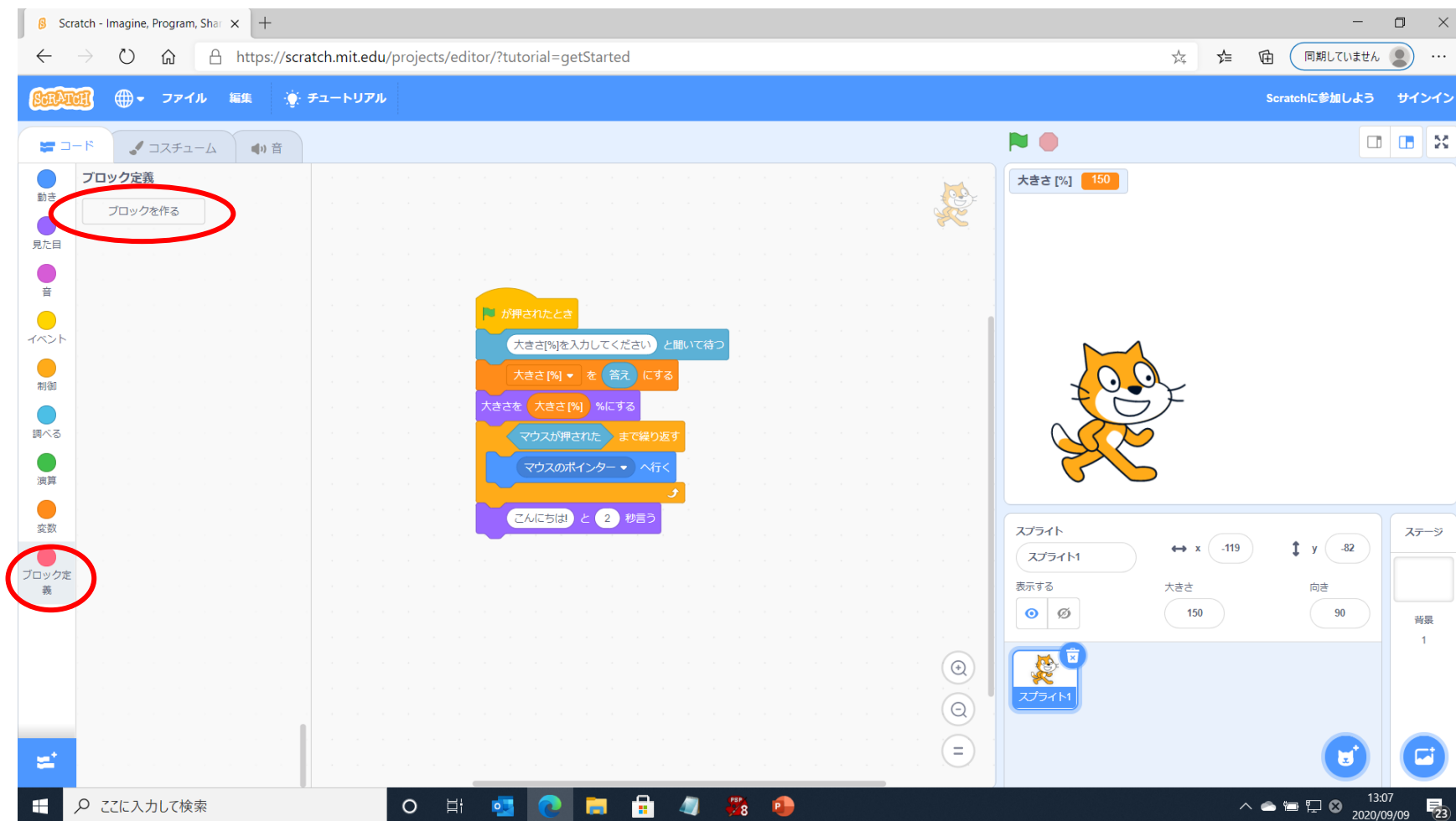
3. プログラミング(3):キャラクターの位置をマウスに連動させる(4):実行

プログラム実行エリアの左上の「緑色の旗」をクリックして実行させますと、15ページと同様に、プログラム実行エリアのキャラクターの下のテキストボックスが表示されますので、大きさを%で入力します。そうしますと、マウスに連動してキャラクターが移動するようになります。マウスの左ボタンを押しますと、「こんにちは」と喋り終了します。



3. プログラミング(4): キャラクターの大きさ設定部をブロック化する(1)

プログラムが複雑になると縦長で見にくくなります。そこで、繰り返し出てくる一連の処理やまとまった処理を、ユーザ定義ブロック(赤色、C言語の関数に相当)で表現することができます。まず、左欄のコードのメニューより「ブロック定義」を選択し、「ブロックを作る」をクリックします。



3. プログラミング(4): キャラクターの大きさ設定部をブロック化する(2)

以下の「ブロックを作る」ダイアログで「ブロック名」を例えば「大きさ設定」に変更し、「OK」ボタンをクリックして閉じます。



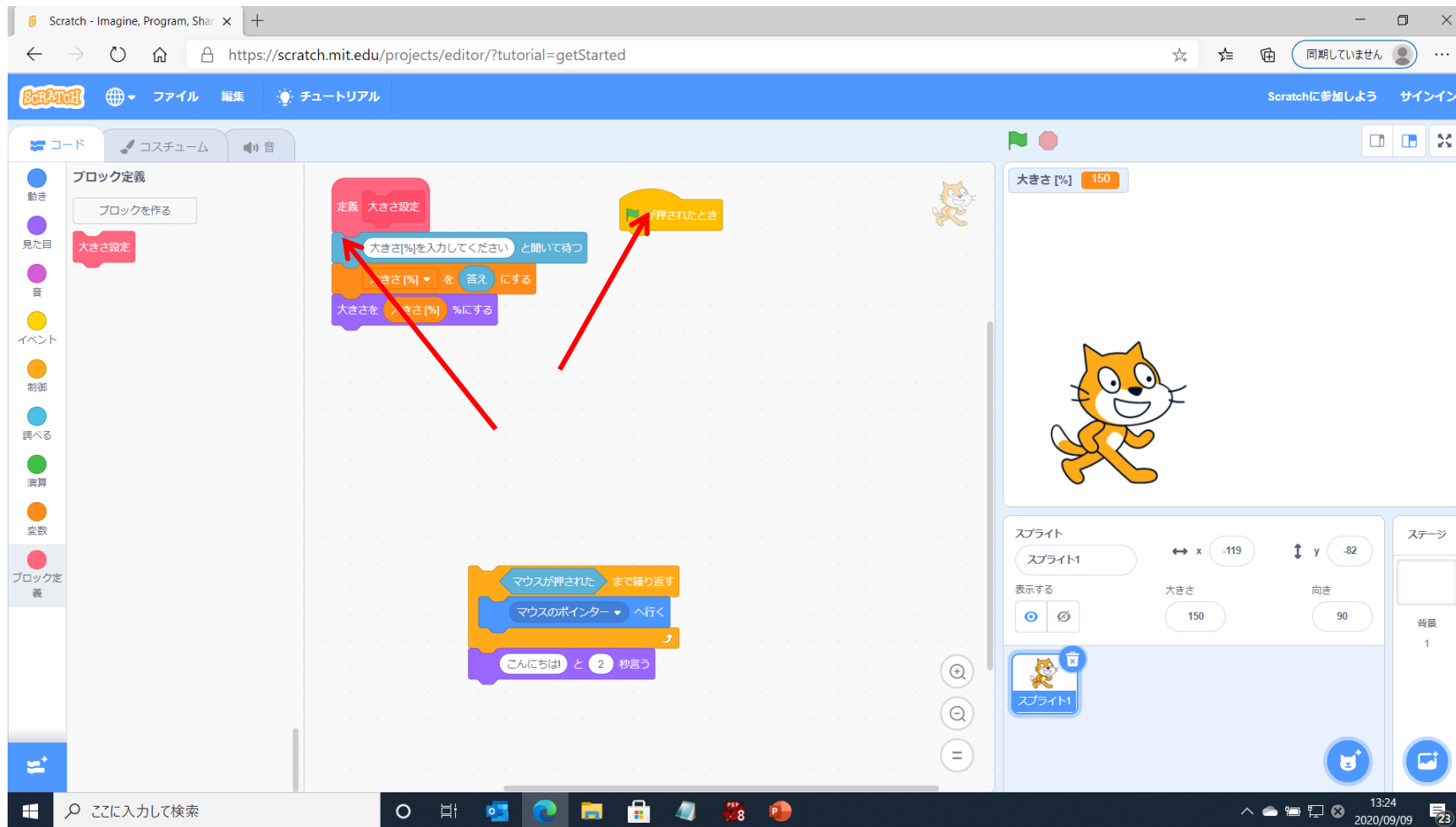
3. プログラミング(4): キャラクターの大きさ設定部をブロック化する(3)

作成していたプログラムの「(大きさ[%])を入力してください」と聞いて待つ」を下方にドラッグして、「旗が押されたとき」より切り離します。更に、「<マウスが押された>まで繰り返す」を下方にドラッグして、「大きさを(大きさ[%])%にする」より切り離します。

The screenshot shows the Scratch programming environment. On the left, the 'Blocks' palette is open, showing a custom block definition named '大きさ設定' (Set Size). The definition is triggered by 'when green flag clicked' and contains a 'repeat until mouse clicked' loop. Inside the loop, the steps are: 'ask for input (size %)', 'set size to input', and 'set size to (size %) %'. A red arrow indicates the flow of the program. On the right, the 'Sprites' panel shows 'Sprite1' with a size of 150%.

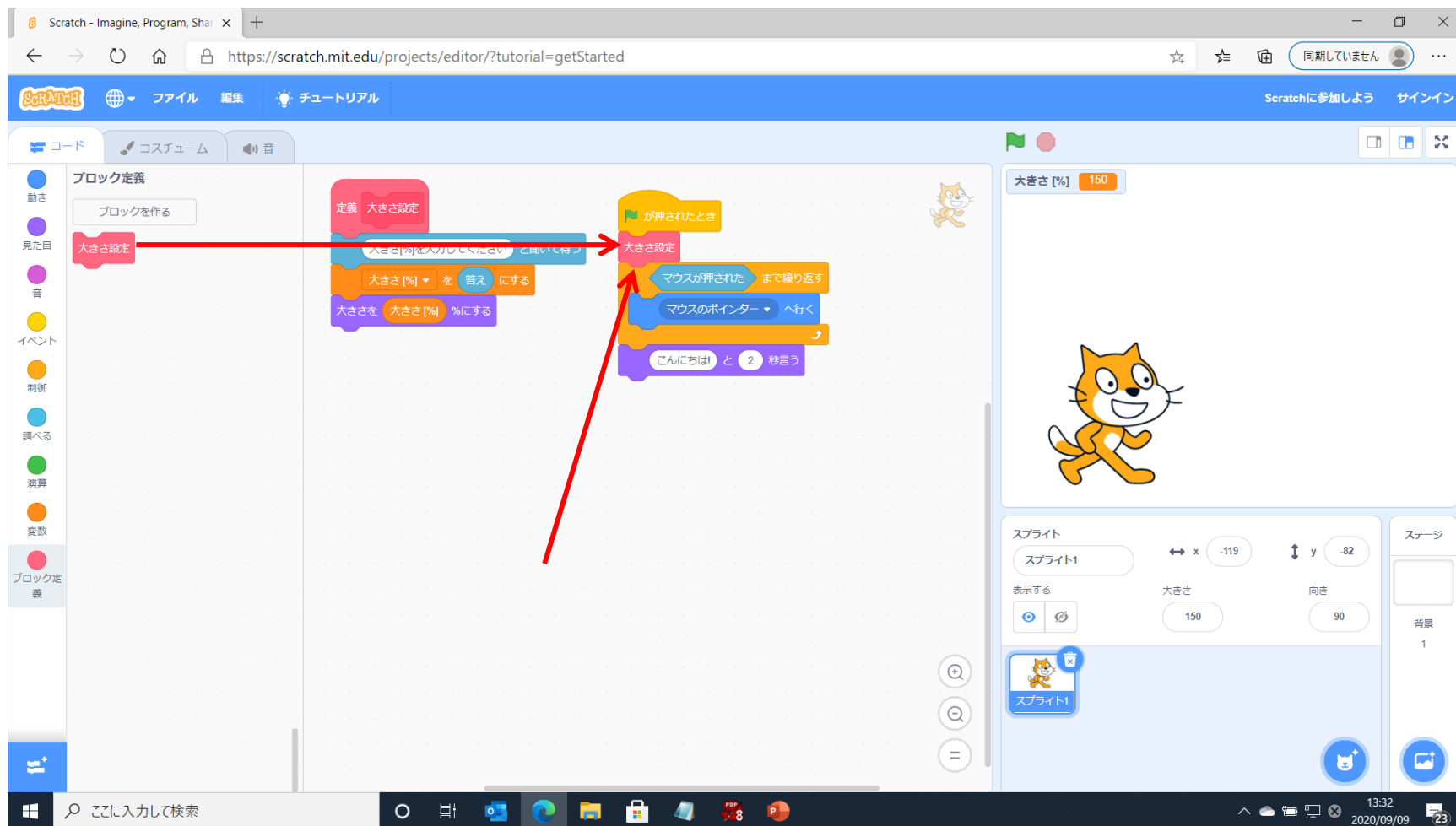
3. プログラミング(4): キャラクターの大きさ設定部をブロック化する(4)

切り離した「(大きさ[%]を入力してください)と聞いて待つ」を先頭とする3ブロックを、左上に新規に生成されたユーザ定義ブロック「定義(大きさ設定)」の下方にドラッグして連結させます。「旗が押されたとき」ブロックは右上の空きスペースに移動させます。



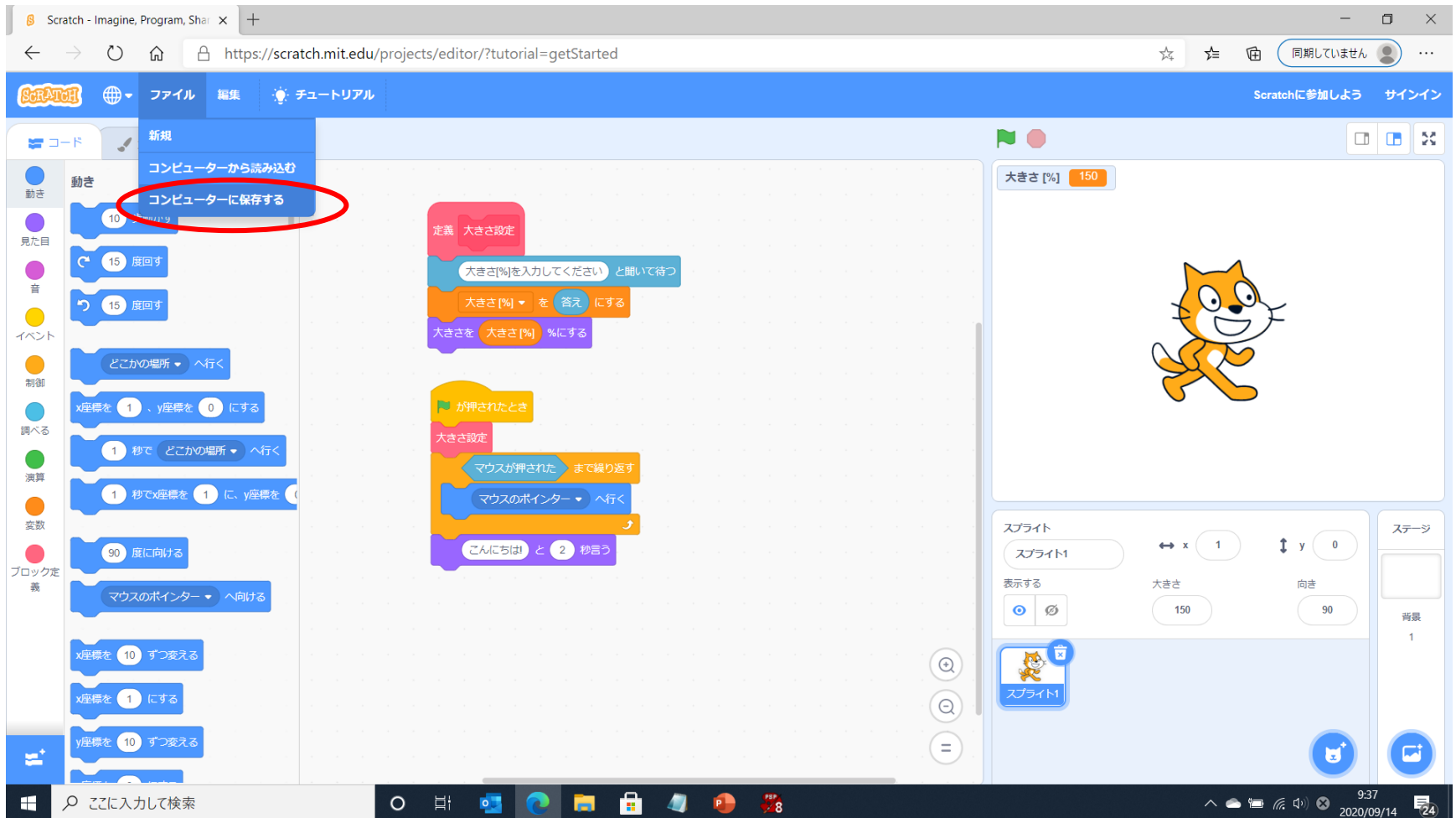
3. プログラミング(4): キャラクターの大きさ設定部をブロック化する(5)

左欄のブロック定義メニューに新規に生成されたユーザ定義ブロック「大きさ設定」を「旗が押されたとき」ブロックの下方にドラッグ & ドロップさせて連結させます。続いて、「大きさ設定」ブロックの下方に「<マウスが押された>まで繰り返す」を先頭とする3ブロックをドラッグして連結させます。これで20ページと同様に実行できます。



4. プログラムの保存: 作成したプログラム(ブロック配置)を保存する

作成したプログラムは左上の「ファイル」メニューの「コンピュータに保存する」をクリックすることにより、“Scratchのプロジェクト.sb3”という名称でローカルPCのダウンロードフォルダに保存できます(ファイル名を変更して、適当なフォルダに保存して下さい)。逆に「コンピュータから読み込む」でローカルPCに保存されているプログラムのファイルを再読み込みできます。



5. 例題(1): 1から100までの整数の総和を求める

count変数とsum変数を定義し、C言語のforループに相当する機能を使用します。"Scratch3_sum.sb3" (メニュー「ファイル」→「コンピュータから読み込む」で本ファイルを指示すれば、本プログラムを読み込みます。)

The screenshot displays the Scratch3 interface with a script area on the left and a stage area on the right. The script area contains the following blocks:

- When green flag clicked (yellow block)
- Set count to 1 (orange block)
- Set sum to 0 (orange block)
- Loop: Repeat until count is greater than 100 (orange block with a green arrow)
- Inside the loop:
 - Set sum to sum + count (orange block)
 - Set count to count + 1 (orange block)
- Speak sum for 5 seconds (purple block)

The stage area shows the Scratch cat sprite with a speech bubble containing the number 5050. The top right corner of the stage displays the current values of the variables: count is 101 and sum is 5050. The bottom right corner shows the sprite's properties: Sprite 1, visible, size 100, and direction 90.

5. 例題(2):リアルタイム時計

年・月・日・時・分・秒の6つの変数を定義し、現在の日時を取得して6つの各変数の値を更新する処理を、1秒間隔に繰り返し、秒の更新に合わせてキャラクターを回転させます。”Scratch3_time.sb3”

The image displays the Scratch3 editor interface for a project titled "Scratch3_time.sb3". The script area on the left contains a loop that starts with a "when green flag is clicked" event, followed by a "repeat until mouse clicked" loop. Inside this loop, the script updates the year, month, day, hour, minute, and second variables to their current values. After updating the seconds, it rotates the cat character by $\text{seconds} \times 6 + 90$ degrees and then waits for 1 second before repeating the loop.

The right side of the interface shows the stage with a cat character. The top right displays the current date and time: 年 2020, 月 9, 日 9, 時 9, 分 28, 秒 38. The bottom right shows the sprite properties for "Sprite1", including its position (x: 8, y: -14), size (100), and rotation (-42 degrees).

5. 例題(2):リアルタイム時計(2):目覚まし時計

リアルタイム時計に対してアラーム時刻(時・分)を入力させると、指定時刻になると、キャラクターが「ニャー」と5回鳴きながら「時間ですよ」としゃべります。

Scratch3_alarm.sb3

年 2021 月 3 日 2

時 17 分 22 秒 0 A時 17 A分 22

時間ですよ!

スプライト

スプライト1

表示する 向き 90

大きさ 100

ステージ

背景 1

が押されたとき

アラーム時刻(時)を入力して下さい と聞いて待つ

A時 を 答え にする

アラーム時刻(分)を入力して下さい と聞いて待つ

A分 を 答え にする

時 = A時 かつ 分 = A分 まで繰り返す

年 を 現在の 年 にする

月 を 現在の 月 にする

日 を 現在の 日 にする

時 を 現在の 時 にする

分 を 現在の 分 にする

秒 を 現在の 秒 にする

秒 * 60 + 90 度に向ける

1 秒待つ

5 回繰り返す

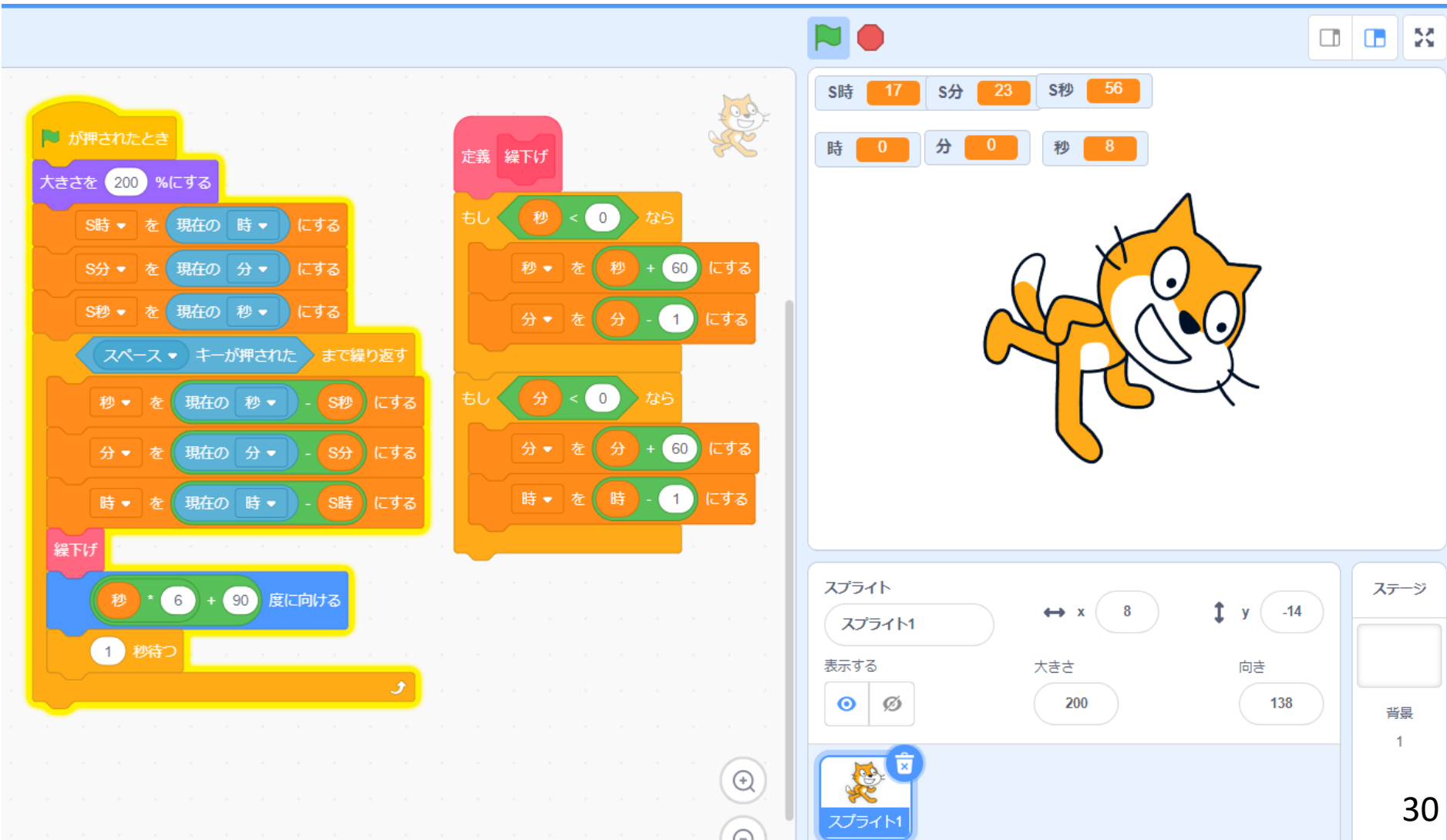
ニャー の音を鳴らす

時間ですよ! と 1 秒言う

5. 例題(2):リアルタイム時計(3):ストップウォッチ

リアルタイム時計に対して、起動時の時刻(時・分・秒)を記録し、起動時の時刻からの経過時間(時・分・秒)を表示させるようにします(年月日の表示は削除)。

"Scratch3_swatch.sb3"



The image shows the Scratch3 Swatch interface for a stopwatch. The main workspace contains two scripts and a definition block. The first script, triggered by a green flag click, sets the display size to 200% and initializes the time variables (S時, S分, S秒) to the current system time. It then enters a loop that repeats until the space key is pressed. Inside the loop, it calculates the elapsed time by subtracting the start time from the current time and updates the display. The second script, triggered by a '線下げ' (line down) button, handles time overflow: if seconds reach 60, it increments minutes and resets seconds; if minutes reach 60, it increments hours and resets minutes. The '線下げ' block also rotates the cat sprite 90 degrees and waits 1 second. The right panel shows the current time as 17:23:56 and the elapsed time as 0:0:8. The cat sprite is positioned at (8, -14) with a size of 200 and a rotation of 138 degrees.

が押されたとき

大きさを 200 %にする

S時 を 現在の 時 にする

S分 を 現在の 分 にする

S秒 を 現在の 秒 にする

スペース キーが押された まで繰り返す

秒 を 現在の 秒 - S秒 にする

分 を 現在の 分 - S分 にする

時 を 現在の 時 - S時 にする

線下げ

秒 * 60 + 90 度に向ける

1 秒待つ

定義 線下げ

もし 秒 < 0 なら

秒 を 秒 + 60 にする

分 を 分 - 1 にする

もし 分 < 0 なら

分 を 分 + 60 にする

時 を 時 - 1 にする

S時 17 S分 23 S秒 56

時 0 分 0 秒 8

スプライト

スプライト1

表示する

大きさ 200

向き 138

ステージ

背景 1

スプライト

5. 例題(3): 四則演算

2つの数値A・Bをキーボード入力し、四則演算と剰余算(AをBで割った余りを計算)を行った結果を表示します。
"Scratch3_calc.sb3"

The image shows a Scratch3 project titled "Scratch3_calc.sb3". The script is triggered by a "when green flag is clicked" event. It prompts the user to input two numbers, A and B. Number A is 300 and Number B is 35. The script then performs five calculations: Addition (300 + 35 = 335), Subtraction (300 - 35 = 265), Multiplication (300 * 35 = 10500), Division (300 / 35 = 8.571429), and Modulo (300 % 35 = 20). The results are displayed on the stage. The Scratch cat sprite is visible on the stage.

Script:

- が押されたとき
- 数値Aを入力してください と聞いて待つ
- 数値A を 答え にする
- 変数 数値A を表示する
- 数値Bを入力してください と聞いて待つ
- 数値B を 答え にする
- 変数 数値B を表示する
- 加算 を 数値A + 数値B にする
- 減算 を 数値A - 数値B にする
- 乗算 を 数値A * 数値B にする
- 除算 を 数値A / 数値B にする
- 剰余算 を 数値A を 数値B で割った余り にする

Stage:

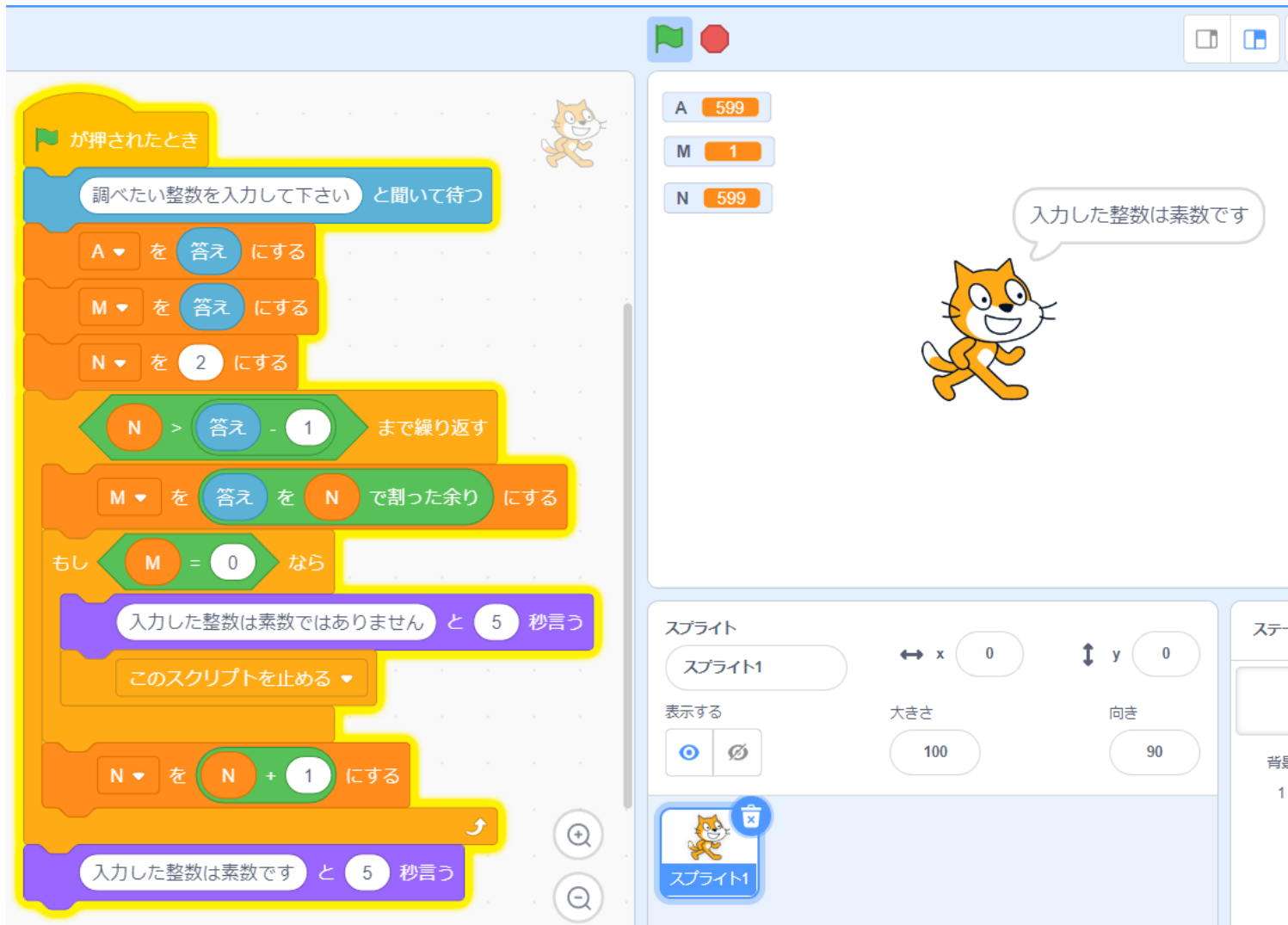
- 数値A: 300
- 数値B: 35
- 加算: 335
- 減算: 265
- 乗算: 10500
- 除算: 8.571429
- 剰余算: 20

Sprite: スプライト1 (Scratch cat)

Stage: スtage 1 (Background 1)

5. 例題(4): 入力された整数の素数判定

1またはそれ自身の値でしか割り切れない整数を素数と言います。入力した整数値を2以上の整数で順に割り算を行い、余りが0になる場合がなければ素数であると判定できます。プログラミングの基本3要素である順次処理、反復処理、条件分岐を全て使用します。”Scratch3_prime.sb3”



The image shows a Scratch3 script for prime number checking. The script is written in Japanese and uses a combination of blocks to handle input, loops, and conditional logic. The script is as follows:

```
when green flag clicked
  ask "Please enter the integer you want to check" and wait
  set A to answer
  set M to answer
  set N to 2
  loop until N > A - 1
    set M to remainder of A divided by N
    if M = 0 then
      say "The entered integer is not a prime number" for 5 sec
      stop this script
    set N to N + 1
  say "The entered integer is a prime number" for 5 sec
```

The script is shown in the Scratch3 editor. The script area on the left contains the following blocks:

- when green flag clicked
- ask "Please enter the integer you want to check" and wait
- set A to answer
- set M to answer
- set N to 2
- loop until N > A - 1
- set M to remainder of A divided by N
- if M = 0 then
- say "The entered integer is not a prime number" for 5 sec
- stop this script
- set N to N + 1
- say "The entered integer is a prime number" for 5 sec

The right side of the image shows the Scratch3 interface. The variables area on the top right shows three variables: A (599), M (1), and N (599). The stage area on the bottom right shows a speech bubble from the Scratch cat saying "The entered integer is a prime number".

5. 例題(5): 体格指数BMIの計算

身長と体重の値をキーボード入力し、BMIを計算し、体格判定します。体格判定は、ユーザ定義ブロック(赤色、C言語の関数に相当)で表現し、22が標準で、25以上を肥満、18.5未満を痩せとします。”Scratch3_bmi.sb3”



The image shows a Scratch3 script for calculating BMI. The script starts with a 'when green flag clicked' event, followed by prompts for height and weight, input fields, and variable displays. The BMI calculation is performed using the formula:
$$\text{BMI} = \frac{\text{体重[kg]} \times 10000}{\text{身長[cm]} \times \text{身長[cm]}}$$
 The script then uses a series of conditional blocks to determine the body type: if BMI > 24.9, it is '肥満' (Obese); if BMI < 18.5, it is '痩せ' (Thin); otherwise, it is '普通' (Normal). A speech bubble with the result is shown, and a 5-second timer is set. The right panel shows the current values: 身長[cm] 175, 体重[kg] 80, BMI 26.122449, and 判定 肥満.

身長[cm] 175
体重[kg] 80
BMI 26.122449
判定 肥満

肥満

スプライト
スプライト1
表示する
向き 90
サイズ 100

ステージ
背景 1

定義 体格判定

もし BMI > 24.9 なら
判定 を 肥満 にする
でなければ
もし BMI < 18.5 なら
判定 を 痩せ にする
でなければ
判定 を 普通 にする
判定 と 5 秒言う

が押されたとき
身長[cm]を入力してください と聞いて待つ
身長[cm] を 答え にする
変数 身長[cm] を表示する
体重[kg]を入力してください と聞いて待つ
体重[kg] を 答え にする
変数 体重[kg] を表示する
BMI を 体重[kg] * 10000 / 身長[cm] / 身長[cm] にする
体格判定

BMI= $\frac{\text{体重[kg]} \times 10000}{\text{身長[cm]} \times \text{身長[cm]}}$

5. 例題(6):検査精度指数(感度と特異度)の計算(1)

新型コロナウイルスのPCR検査、抗原検査で話題になる検査精度(感度と特異度)の計算を行います。感度とは、真の感染者数(誤って陰性判定された見逃し感染者を含む)に対して陽性判定された割合[%]、特異度は真の健常者数(誤って陽性判定された健常者を含む)に対して検査で陰性判定された割合[%]です。

各値が高いほど精度が良いが、通常は一方の値が高いと、もう一方の値は低くなる傾向があります。健康診断(がん検診)では、感度>特異度の場合が多く(見逃しは少ないが要精密検査が多い)、PCR検査・抗原検査では感度<特異度になっています(健常者を誤って陽性判定して隔離することは少ないが、感染者の見逃しが多い)。

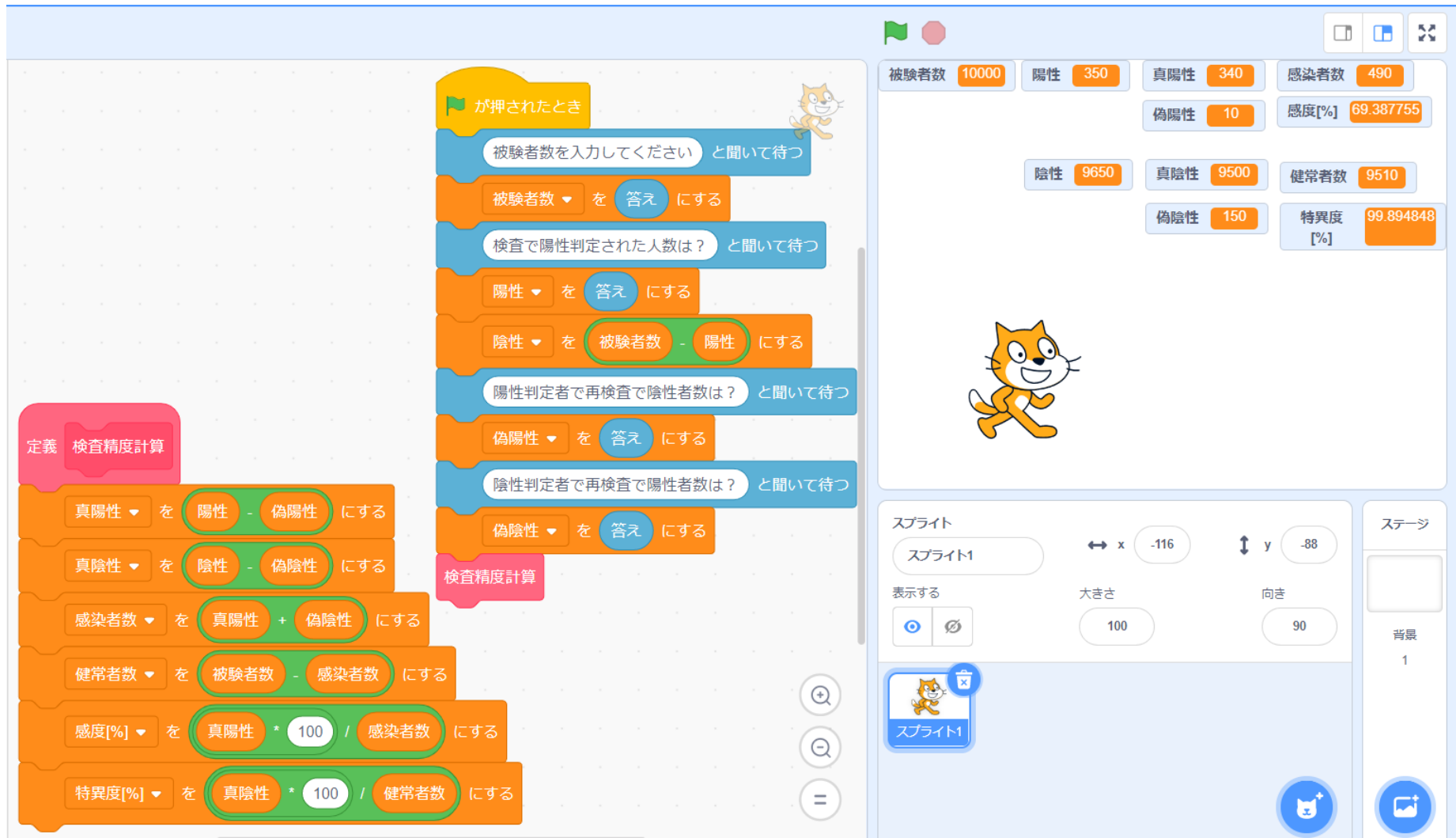
以下例は新型コロナウイルスのPCR検査の典型的な精度

"Scratch3_kensa.sb3"



5. 例題(6): 検査精度指数(感度と特異度)の計算(2)

被験者数、陽性判定数、偽陽性者数、偽陰性者数の4つの値をキーボード入力し、感染者数、健常者数、感度、特異度の各値を計算します。”Scratch3_kensa.sb3”



The image shows a Scratch3 project titled "Scratch3_kensa.sb3" designed for calculating medical test accuracy. The script is triggered by a "when green flag clicked" event and follows a sequence of input prompts and calculations. The interface includes a stage with a cat sprite and a sidebar with various input fields and calculated values.

Script Logic:

- When green flag clicked, ask "被験者数を入力してください" (Please enter the number of subjects) and wait.
- Set "被験者数" (Number of subjects) to the answer.
- Ask "検査で陽性判定された人数は?" (How many people were diagnosed as positive in the test?) and wait.
- Set "陽性" (Positive) to the answer.
- Set "陰性" (Negative) to "被験者数" minus "陽性".
- Ask "陽性判定者で再検査で陰性者数は?" (How many people diagnosed as positive in the re-test were negative?) and wait.
- Set "偽陽性" (False positive) to the answer.
- Ask "陰性判定者で再検査で陽性者数は?" (How many people diagnosed as negative in the re-test were positive?) and wait.
- Set "偽陰性" (False negative) to the answer.
- Define a function "検査精度計算" (Calculate test accuracy) with the following steps:
 - Set "真陽性" (True positive) to "陽性" minus "偽陽性".
 - Set "真陰性" (True negative) to "陰性" minus "偽陰性".
 - Set "感染者数" (Number of infected) to "真陽性" plus "偽陰性".
 - Set "健常者数" (Number of healthy) to "被験者数" minus "感染者数".
 - Set "感度[%]" (Sensitivity [%]) to "真陽性" multiplied by 100, divided by "感染者数".
 - Set "特異度[%]" (Specificity [%]) to "真陰性" multiplied by 100, divided by "健常者数".

Interface Data:

Category	Value
被験者数	10000
陽性	350
真陽性	340
偽陽性	10
感染者数	490
感度[%]	69.387755
陰性	9650
真陰性	9500
偽陰性	150
健常者数	9510
特異度[%]	99.894848

5. 例題(6): 検査精度指数(感度と特異度)の計算(3): プログラム部拡大

"Scratch3_kensa.sb3"

定義 検査精度計算

真陽性 ▼ を 陽性 - 偽陽性 にする

真陰性 ▼ を 陰性 - 偽陰性 にする

感染者数 ▼ を 真陽性 + 偽陰性 にする

健常者数 ▼ を 被験者数 - 感染者数 にする

感度[%] ▼ を $\frac{\text{真陽性} \times 100}{\text{感染者数}}$ にする

特異度[%] ▼ を $\frac{\text{真陰性} \times 100}{\text{健常者数}}$ にする

が押されたとき

被験者数を入力してください と聞いて待つ

被験者数 ▼ を 答え にする

検査で陽性判定された人数は? と聞いて待つ

陽性 ▼ を 答え にする

陰性 ▼ を $\text{被験者数} - \text{陽性}$ にする

陽性判定者で再検査で陰性者数は? と聞いて待つ

偽陽性 ▼ を 答え にする

陰性判定者で再検査で陽性者数は? と聞いて待つ

偽陰性 ▼ を 答え にする

検査精度計算

5. 例題(7):カレンダー曜日の計算(1)

年・月・日の3つの値をキーボード入力し、曜日を計算します。”Scratch3_yobi.sb3”

The screenshot displays the Scratch3 interface with a script for calculating the day of the week (Yobi) based on Year, Month, and Day inputs. The script is organized into three main sections: Yobi display, Yobi calculation, and Yobi display.

Yobi display (定義 曜日の表示):

- もし Yobi = 0 なら
曜日を 日(Sunday) にする
- でなければ
もし Yobi = 1 なら
曜日を 月(Monday) にする
- でなければ
もし Yobi = 2 なら
曜日を 火(Tuesday) にする
- でなければ
もし Yobi = 3 なら
曜日を 水(Wednesday) にする
- でなければ
もし Yobi = 4 なら
曜日を 木(Thursday) にする
- でなければ
もし Yobi = 5 なら
曜日を 金(Friday) にする
- でなければ
曜日を 土(Saturday) にする

Yobi calculation (定義 曜日の計算):

- もし Month < 3 なら
month2 を Month + 12 にする
year2 を Year - 1 にする
- sum を year2 + Day にする
dvar を year2 / 4 にする
- 小数点以下切捨て 1
dvar を year2 / 100 にする
- 小数点以下切捨て -1
dvar を year2 / 400 にする
- 小数点以下切捨て 1
dvar を month2 * 2.8 + 1.6 にする
- 小数点以下切捨て 1
Yobi を sum を 7 で割った余り にする

Yobi display (定義 小数点以下切捨て number or text):

- ivar を dvar を四捨五入 にする
- もし ivar > dvar なら
ivar を ivar - 1 にする
- もし number or text < 0 なら
sum を sum - ivar にする
- でなければ
sum を sum + ivar にする

が押されたとき:

- 年を西暦4桁で入力してください と聞いて待つ
- Year を 答え にする
- year2 を Year にする
- 月を入力してください と聞いて待つ
- Month を 答え にする
- month2 を Month にする
- 日を入力してください と聞いて待つ
- Day を 答え にする
- 曜日の計算
- 曜日の表示
- 曜日を 5 秒待つ

The right panel shows the input fields: Year (2021), Month (3), Day (3), and Yobi (3). The output field shows the calculated day: 水(Wednesday). The stage area displays the Scratch cat character with a speech bubble saying "水(Wednesday)".

5. 例題(7): カレンダー曜日の計算(2): プログラム部拡大

The image shows a Scratch script divided into three main sections: 曜日の表示 (Day of the Week Display), 曜日の計算 (Day of the Week Calculation), and 小数点以下切捨て (Truncate after decimal point).

曜日の表示 (Day of the Week Display):

- Define: 曜日の表示
- Logic: A series of nested if-then blocks that set the day of the week based on the value of Yobi (0 to 5).
 - もし Yobi = 0 なら: 曜日 を 日(Sunday) にする
 - でなければ: もし Yobi = 1 なら: 曜日 を 月(Monday) にする
 - でなければ: もし Yobi = 2 なら: 曜日 を 火(Tuesday) にする
 - でなければ: もし Yobi = 3 なら: 曜日 を 水(Wednesday) にする
 - でなければ: もし Yobi = 4 なら: 曜日 を 木(Thursday) にする
 - でなければ: もし Yobi = 5 なら: 曜日 を 金(Friday) にする
 - でなければ: 曜日 を 土(Saturday) にする

曜日の計算 (Day of the Week Calculation):

- Define: 曜日の計算
- Logic: A series of nested if-then blocks that calculate the day of the week based on the values of Month, Year, and Day.
 - もし Month < 3 なら: month2 を Month + 12 にする
 - year2 を Year - 1 にする
 - sum を year2 + Day にする
 - dvar を year2 / 4 にする
 - 小数点以下切捨て 1
 - dvar を year2 / 100 にする
 - 小数点以下切捨て -1
 - dvar を year2 / 400 にする
 - 小数点以下切捨て 1
 - dvar を month2 * 2.6 + 1.6 にする
 - 小数点以下切捨て 1
 - Yobi を sum を 7 で割った余り にする

小数点以下切捨て (Truncate after decimal point):

- Define: 小数点以下切捨て number or text
- Logic: A series of nested if-then blocks that truncate the decimal part of a number.
 - ivar を dvar を四捨五入 にする
 - もし ivar > dvar なら: ivar を ivar - 1 にする
 - もし number or text < 0 なら: sum を sum - ivar にする
 - でなければ: sum を sum + ivar にする

Main Script:

- が押されたとき: 年を西暦4桁で入力してください と聞いて待つ
- Year を 答え にする
- year2 を Year にする
- 月を入力してください と聞いて待つ
- Month を 答え にする
- month2 を Month にする
- 日を入力してください と聞いて待つ
- Day を 答え にする
- 曜日の計算
- 曜日の表示
- 曜日 と 5 秒言う

5. 例題(8): キャラクターのランダムウォーク(1)

キャラクターの位置(x,y)、向き、大きさの4つのパラメータを0.5秒間隔に乱数で変更し、キャラクターに生きているような動きを付けます。”Scratch3_life.sb3”

The image shows a Scratch3 script for a cat character's random walk. The script is organized into several sections:

- When clicked:** A loop that repeats until the mouse is clicked again. Inside the loop, it updates the X and Y coordinates, the angle, and the size of the character.
- Define X coordinate update:** A function that updates the X coordinate. It sets the X coordinate to a random value between X coordinate - 30 and X coordinate + 30. If the X coordinate is greater than 220, it sets it to -220. If the X coordinate is less than -220, it sets it to 220.
- Define Y coordinate update:** A function that updates the Y coordinate. It sets the Y coordinate to a random value between Y coordinate - 30 and Y coordinate + 30. If the Y coordinate is greater than 160, it sets it to -160. If the Y coordinate is less than -160, it sets it to 160.
- Define angle update:** A function that updates the angle. It sets the angle to a random value between angle - 30 and angle + 30. If the angle is greater than 180, it sets it to 0. If the angle is less than -180, it sets it to 0.
- Define size update:** A function that updates the size. It sets the size to a random value between size - 10 and size + 10. If the size is greater than 300, it sets it to 100. If the size is less than 10, it sets it to 100.

The character's current position is X: 50, Y: -16. The size is 114 and the angle is 17. The background is a solid blue color.

5. 例題(8): キャラクターのランダムウォーク(2): プログラム部拡大

"Scratch3_life.sb3"

