

Scratch とプログラミング思考

(Scratch)

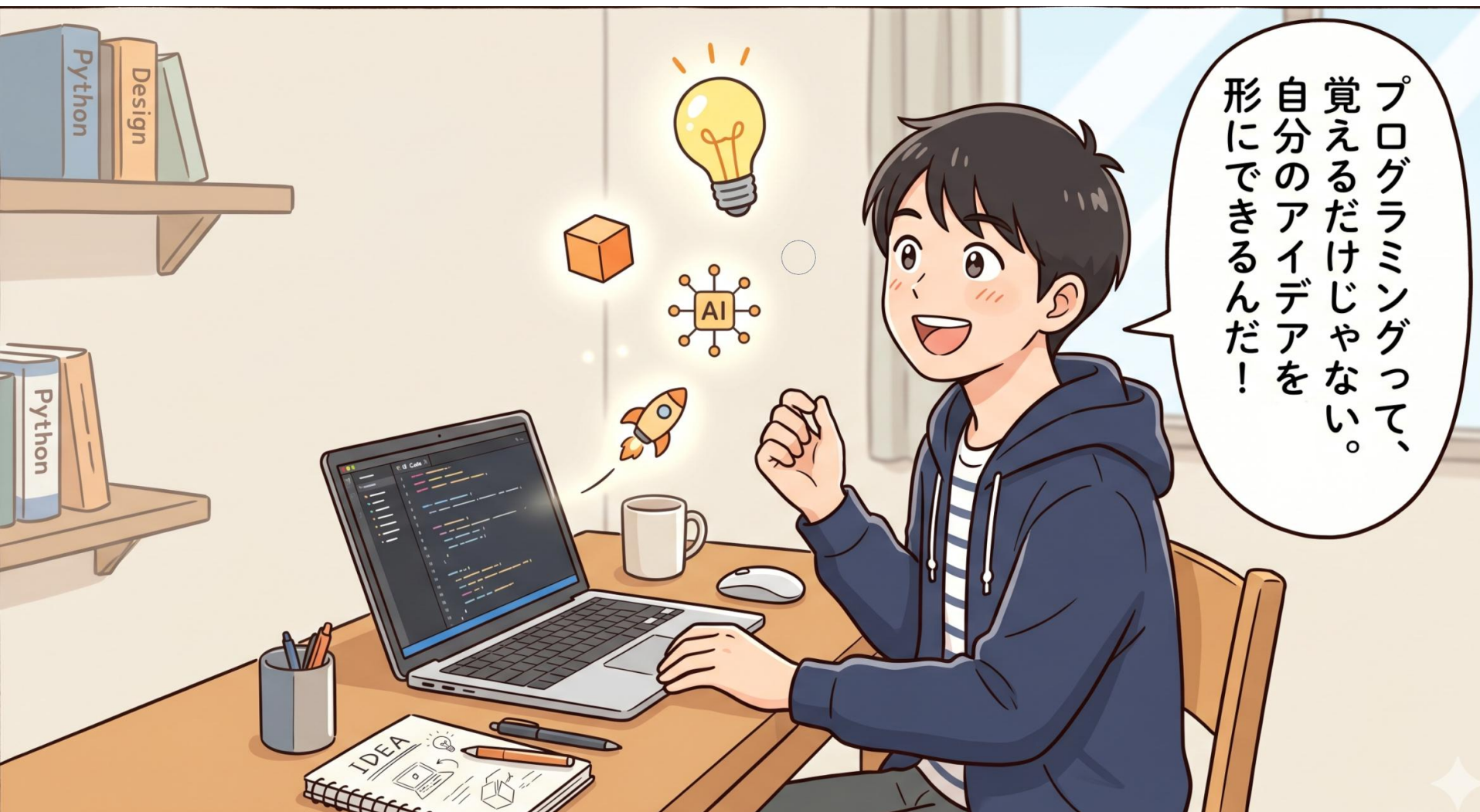
URL: <https://www.kkaneko.jp/pro/man/index.html>

金子邦彦





1. プログラミング ー 人間の力を増幅する



プログラミングは創造的 — 授業の狙い

×

覚える、
問題を解くだけ

捉え直す

○

創造的な作業

プログラミング = 人間の力を増幅する営み

ゴール: 応用へ進むための土台をつくる

説明パート

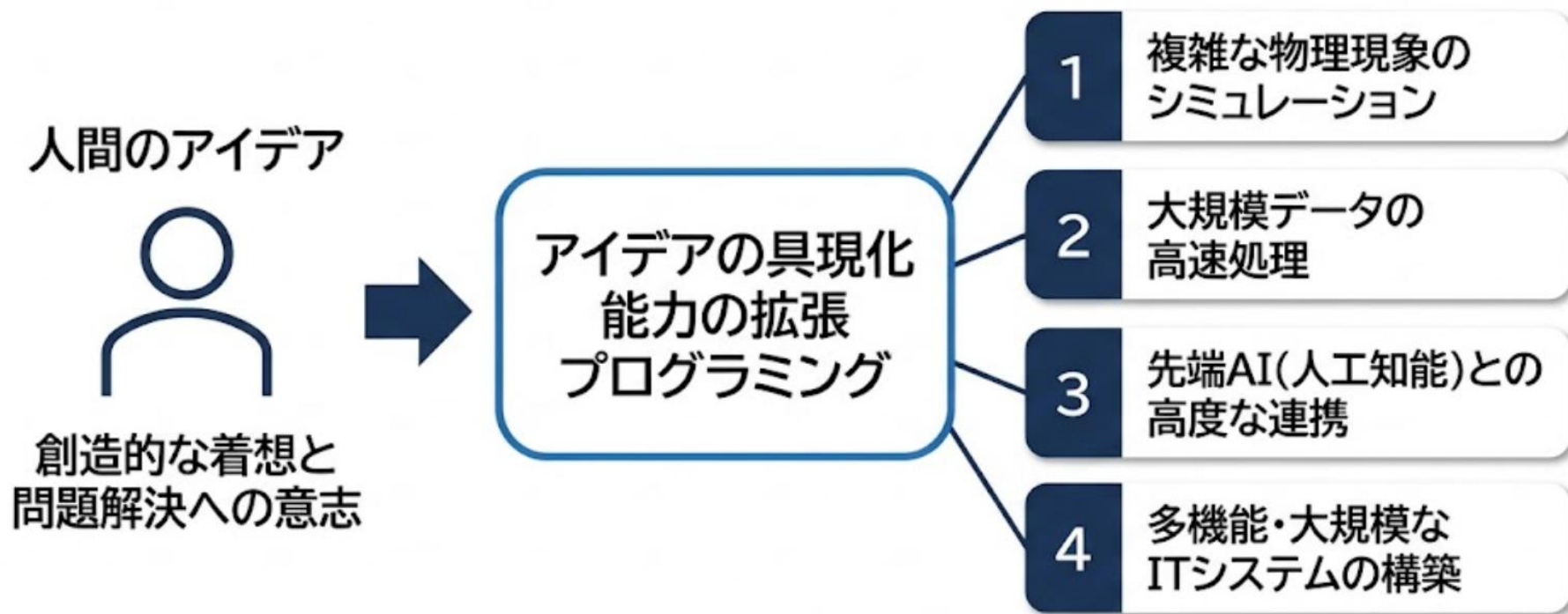
プログラミング的思考の枠組みを学ぶ

+

演習パート

CodeCombat で楽しく体得する

プログラミングで何ができるのか



誤解: 単なる『覚えて、問題を解くだけの勉強』

正解: 『自分のアイデアを形にする』創造的な作業

プログラムとは — 命令を書いた手順の集まり



プログラムは、命令を書いた手順の集まり

指示を与える

自動で処理

結果を得る



合計40歩進んだ!

だから複雑な作業も**自動化**・**効率化**できる

プログラミングを学ぶことで何が身につくか



将来へ

Step1

使いこなす

コンピュータを自分の
思い通りに活用できる

Step2

増幅する

コンピュータの活用
で、自分自身の能力を
増幅できる

Step3

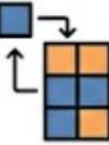
成長する

創造力・発想力・
デザイン力・行動力・
チャレンジ精神・
論理的思考力

Step4

広げる

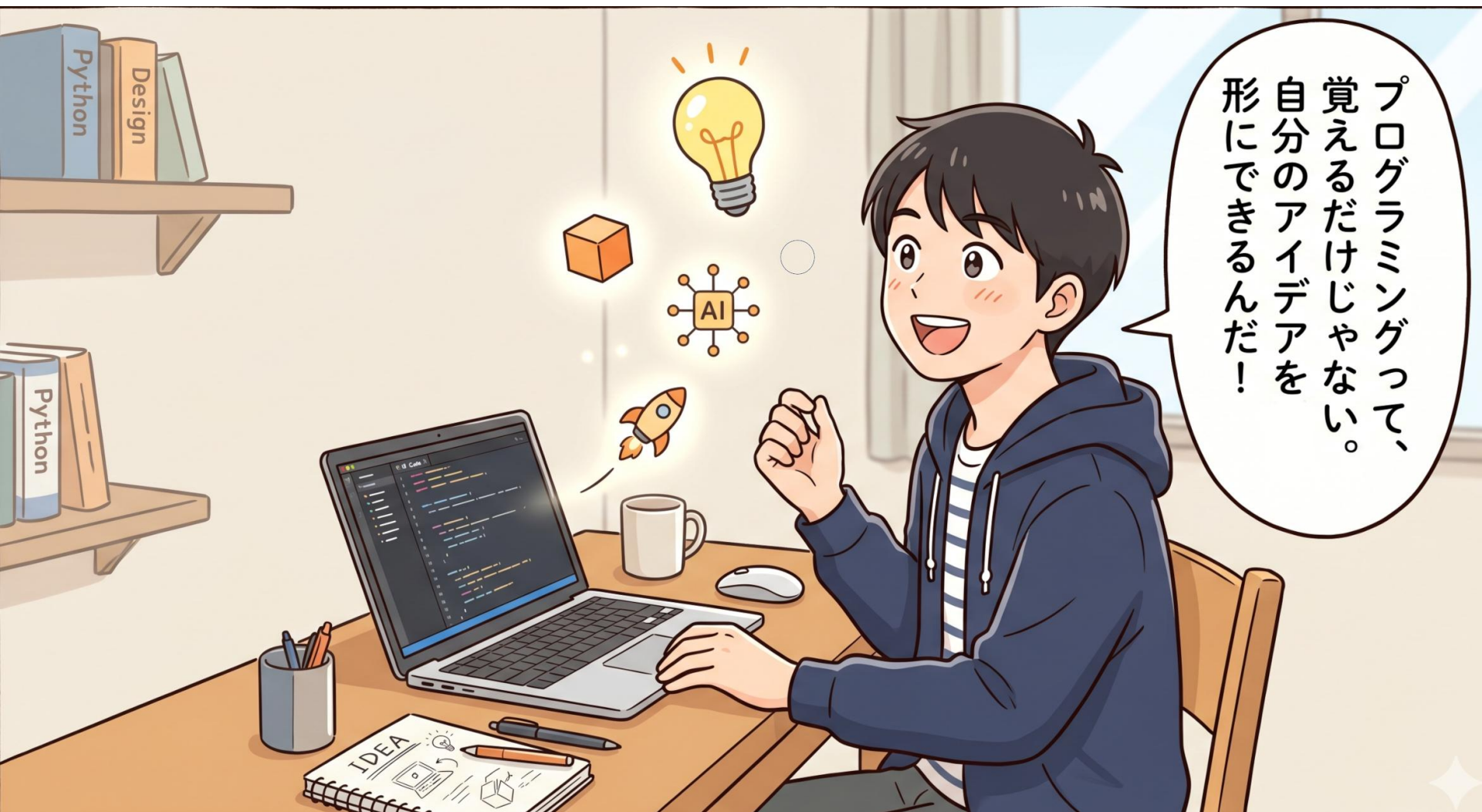
可能性を広げる。
プログラミングは
『部品』を組み立て
て作品を作ることに
似ている
→ エンジニアの素養が
身につく





7-1. プログラミングとは何か

ー 人間の力を増幅する



情報工学科の学び

学ぶこと

コンピュータ



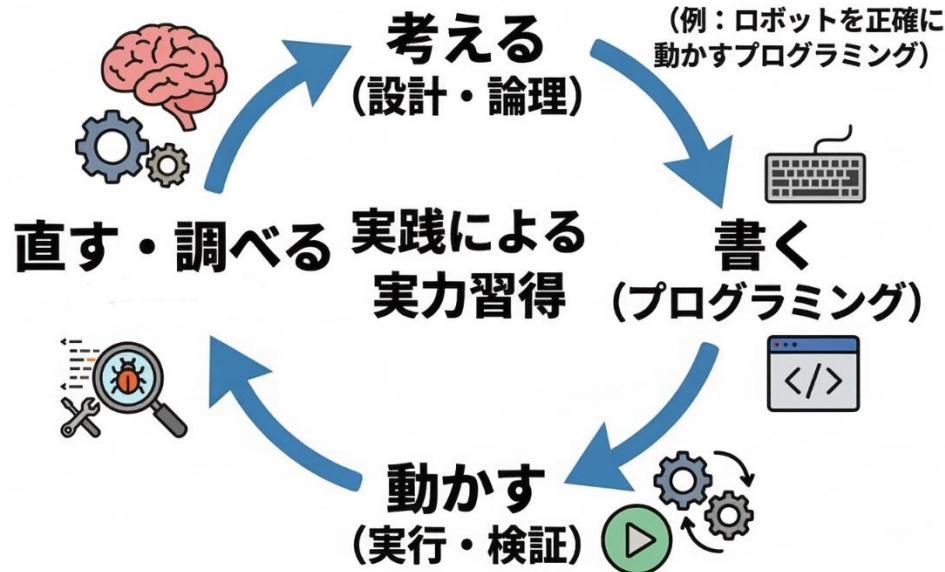
インターネット

デジタル活用



情報工学の
面白さ

実践的な学びのサイクル



学習スタイル



個人ワーク
(一人で取り組む)



グループワーク
(仲間と助け合う)

創造 
(新しいアイデア)

設計 
(システム全体)

具現化  
(形にする・実装)

プログラミングは創造的 — 授業の狙い

×

覚える、
問題を解くだけ

捉え直す

○

創造的な作業

プログラミング = 人間の力を増幅する営み

ゴール: 応用へ進むための土台をつくる

説明パート

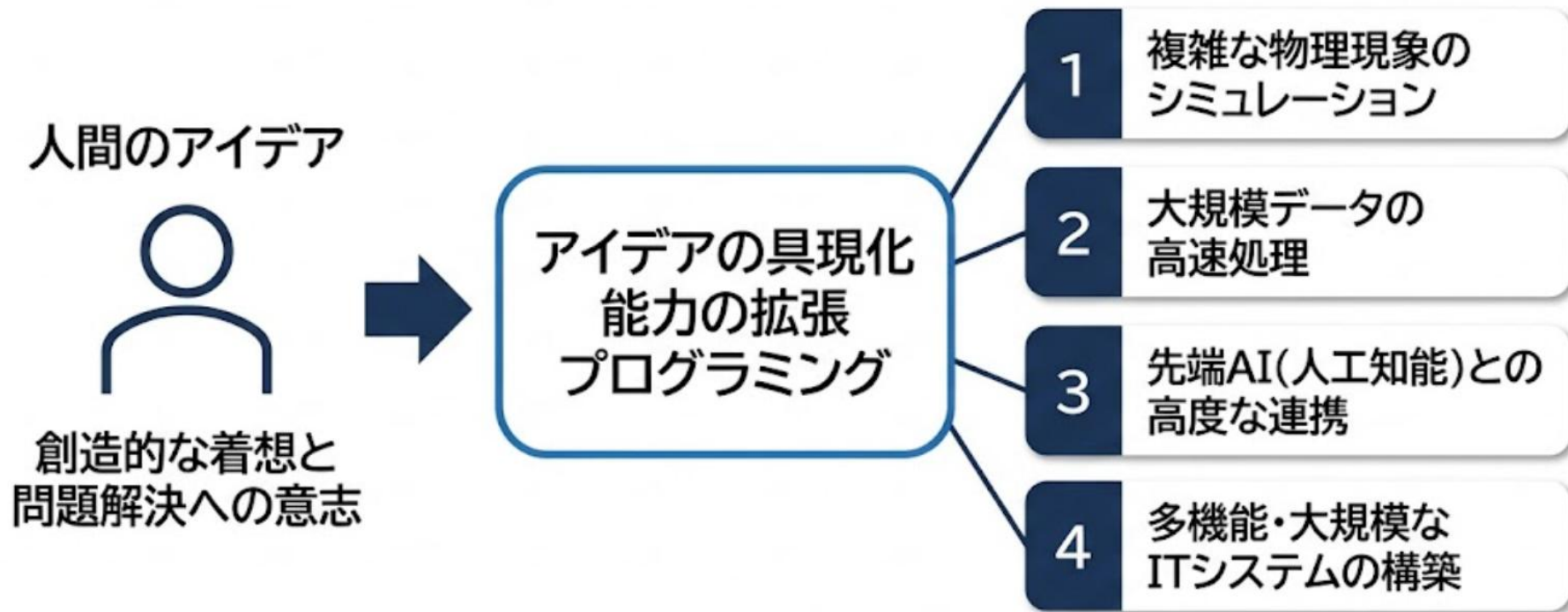
プログラミング的思考の枠組みを学ぶ

+

演習パート

CodeCombat で楽しく体得する

プログラミングで何ができるのか



誤解: 単なる『覚えて、問題を解くだけの勉強』

正解: 『自分のアイデアを形にする』創造的な作業

プログラムとは — 命令を書いた手順の集まり



プログラムは、命令を書いた手順の集まり

指示を与える

自動で処理

結果を得る



合計40歩進んだ!

だから複雑な作業も**自動化**・**効率化**できる

プログラミングを学ぶことで何が身につくか



将来へ

Step1

使いこなす

コンピュータを自分の
思い通りに活用できる

Step2

増幅する

コンピュータの活用
で、自分自身の能力を
増幅できる

Step3

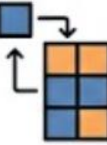
成長する

創造力・発想力・
デザイン力・行動力・
チャレンジ精神・
論理的思考力

Step4

広げる

可能性を広げる。
プログラミングは
『部品』を組み立て
て作品を作ることに
似ている
→ エンジニアの素養が
身につく





これからは"AIに指示してプログラミングする"側に立つ — LLMは使いこなす道具

プログラミング学習における LLM の活用

LLM とは

- 大規模言語モデル (Large Language Model)
- 自然言語でやり取りし、コードの生成・説明・修正を支援する AI

代表例：ChatGPT / Claude / Google Colab の AI アシスタント

特性を
理解

学習パートナー としての二面性

強み

強力な学習パートナーになる (コード生成・説明・修正を即座に支援)

注意点

誤ったコードを自信ありげに提示することがある

正しく
使う

望ましい使い方

1. 生成されたコードは必ず自分で実行して確かめる
2. なぜそう書くのかを LLM に説明させる
3. まずは LLM に頼らず基本概念を体得する

LLM活用の具体例

大前提：生成されたコードはそのまま信用せず、自分で実行して結果を確かめること

活用例1：コードの生成を頼む

依頼の例

Pythonで [1, 2, 3] と [4, 5, 5] の折れ線グラフを描くコードを書いて



LLMの応答

plt.plot(...) を使ったコードが返ってくる



学習者の行動

Colabなどで実行し、
思いどおりのグラフが出るかを確認する

活用例2：コードの意味を説明させる

依頼の例

この1行のどこがオブジェクトで、どこがメソッドで、どこが引数か説明して



LLMの応答

各部分の役割を解説してくれる



学習者の行動

『なぜそう書くのか』を
説明させることで、自分の理解を固める

LLM活用の具体例 – やるべき使い方とやってはいけない使い方

エラーの原因を尋ねる

推奨される使い方

実行してエラーが出る

↓
エラーメッセージをそのまま貼り付ける

↓
「このエラーの意味と直し方を教えて」と質問

↓
示された直し方を理解し、自分で修正して再実行

書き換え（リファクタリング）を頼む

推奨される使い方

• 「このコードを変数を使って書き直して」

• 「score = score + 1 のような再代入の例に書き換えて」

学んだ概念に沿った書き換えを依頼することで、概念と実コードの対応が見えやすくなる

やってはいけない使い方

避けるべき使い方

- 生成されたコードをコピーして提出するだけ
- 自分では一度も実行しない
- 意味を確認しない

強力な学習パートナーではなく、誤りを増幅する道具になってしまう

LLMは「自分で実行し、意味を確認する」使い方で初めて学習パートナーになる



7-2. プログラミング的思考と実践



ねこが進むほどスコアが増える — この小さな世界から、プログラミングが始まる

ねこの行動と得点（スコア）

ねこが10歩進むたびにスコアが1増え、10回くりかえして10になる



10歩を10回くりかえすと、スコアは10になる

対象・構造・条件の3点で世界を捉える

対象 (世界にあるモノ)

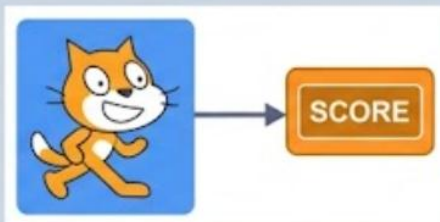


→ ねこ：
この世界で
動く 主体



→ スコア：
ねこの成果
果を測る量

構造 (モノどうしの関係)



→ ねこが前に進むこと
と、スコアが増えるこ
とが結びついている

前進と得点が連動しているのが、
この世界のしくみ

条件 (世界の決まり)

ねこは1回に
10歩ずつ進む

ねこが進むと、
スコアが1増える

ねこの行動とスコアを
支配するルール

プログラムと『3つの道具』

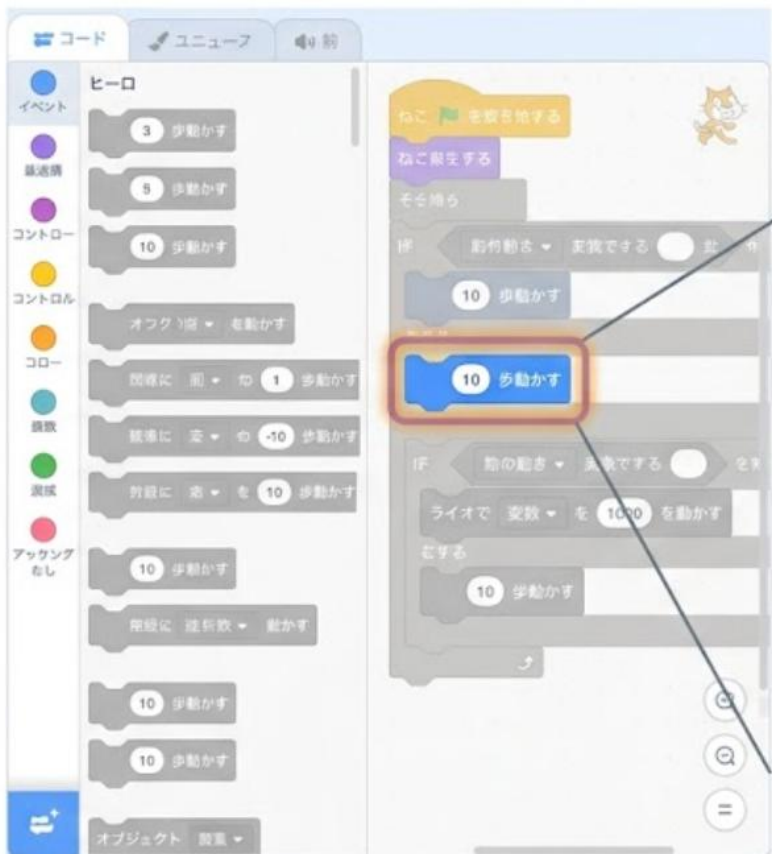


このプログラムが『3つの道具』でできている



- ① オブジェクト・メソッド・引数
- ② 変数と代入
- ③ アルゴリズムの感覚

① オブジェクト・メソッド・引数



オブジェクト=動かす対象 (ねこ)



メソッド=対象ができる動作

引数=動作に渡す数値

② 変数と代入



Scratch script showing variable initialization and incrementation:

- Flag clicked: Set score to 11
- New block: Set score to 0
- Set score to 10
- Increment score by 1
- Repeat loop: Set score to 0, then increment score by 1

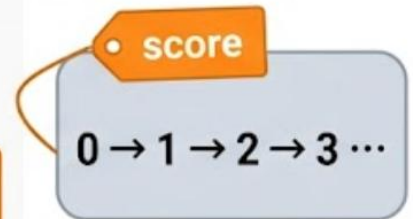
「score を 0 にする」
→ 代入：名札 score に値0を入れる（はじめの得点）



score を 0 にする



score を 1 ずつ変える



「score を 1 ずつ変える」
→ 再代入：今の score に1を足して入れ直す

③ アルゴリズムの感覚

手順1 →  旗が押されたとき (始まり1の合図)

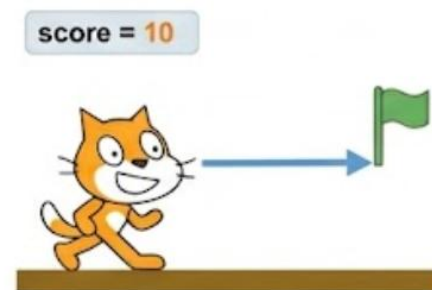
手順2 →  score を 0 にする (準備)

手順3 → 10回繰り返す  10 回繰り返す (くりかえし)

手順4 → 10歩動かす+
scoreを1つ変える
(くりかえす中身)

 10 歩動かす

 score を 1 ずつ変える



どんな順番でブロックを並べればゴールに着くかを考える力

プログラムで使用される『3つの道具』

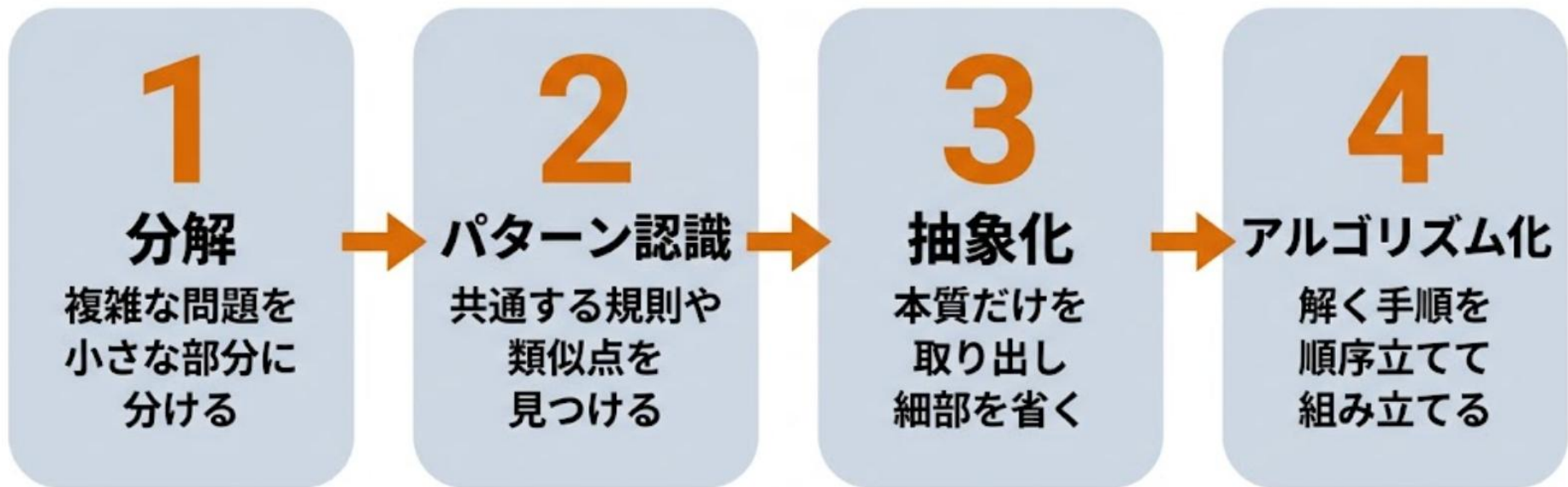


① オブジェクト・メソッド・引数
ねこ・動かす・10のまとまり

② 変数と代入
scoreに値を入れ、書き換える

③ アルゴリズムの感覚
ブロックを正しい順に積み重ねる

複雑な問題を解くための四つの考え方



現代の知的作業を支える基礎能力

『ねこが進むとスコアが増える』世界を、 分解・パターン認識・抽象化・アルゴリズム化

1 分解

10歩動かす



世界のできごとを最小の動作に分ける：
『10歩動かす』と
『scoreを1ずつ変える』の2つ

2 パターン認識

10歩動かす

→ scoreを1ずつ変える

くりかえし現れる
同じ形を見つける：
『10歩動かす →
scoreを1ずつ変える』
の組が何度も現れる

3 抽象化

N歩は N点

例
5歩なら？

M歩は M点

例
3点なら？

変わりうる数値を見
抜く：歩数や得点はル
ールが変われば変わる。
『N歩進むとM点増え
る』と一般化し、10や
1はその一例と捉える

4 アルゴリズム化

scoreを0にする（準備）

決めた回数くりかえす

N歩動かす+

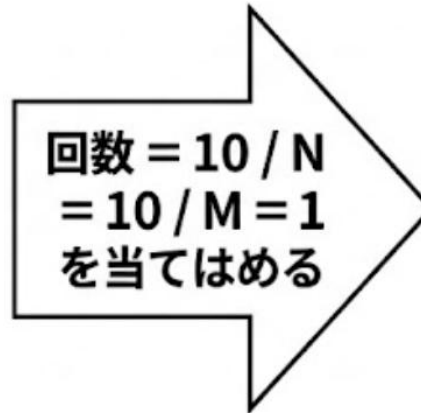
scoreをM点ずつ変える

N=10, M=1

一般化した手順を順
序立てる：準備して
てから、決めた回数
くりかえす形に並べ、
今回はN=10・M=1を
当てはめる

アルゴリズム化の結果に、10, 10, 1の具体的な値を設定

一般の形 (N・M・回数は変わらうる)



Scratchのブロック (数値を当てはめた形)



3. 変数に関する演習



ブロックを組み合わせるだけで、キャラクターを自由に操れる

Scratch : ブロックで動かすプログラミング



Scratch は scratch.mit.edu 上の Web ブラウザで動かす
(インストール不要)

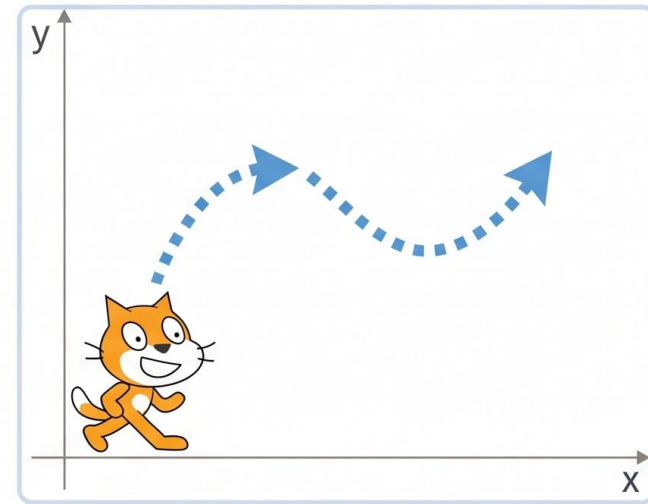
① プログラムを作る

② 命令の通りに

③ キャラクタが動く

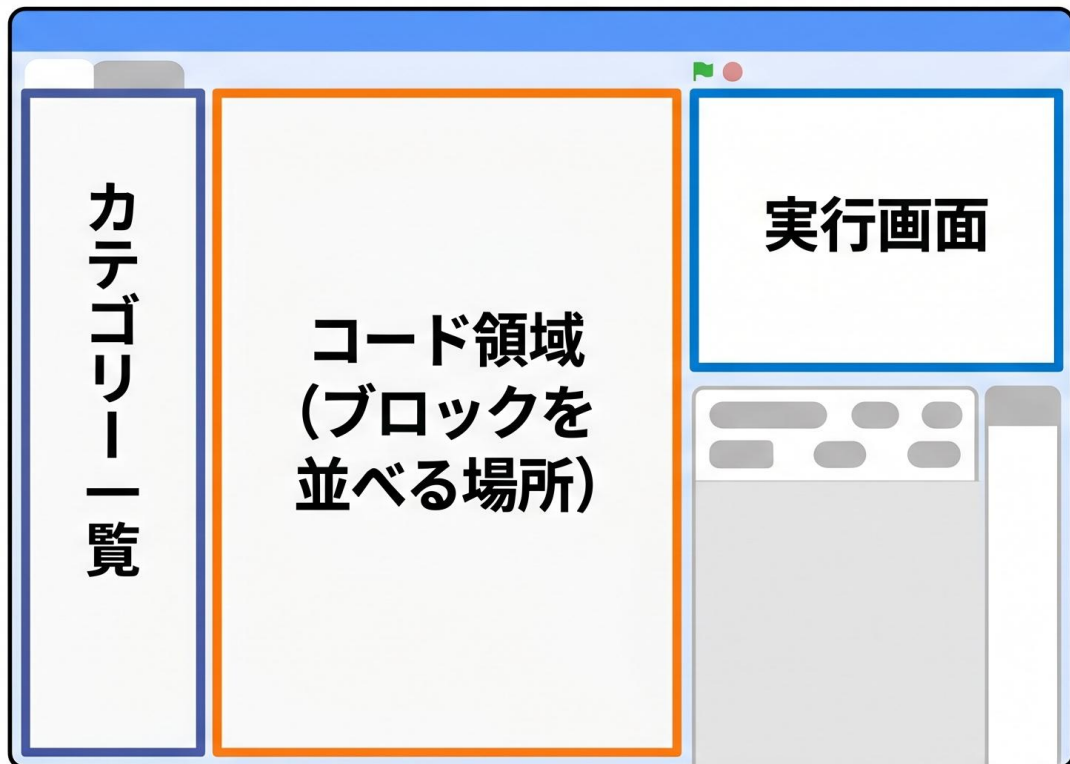


実行



ブロックを組み合わせるだけで、キャラクタを自在に操れる

画面構成



ブロック操作の基本

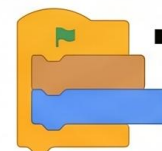
① ブロックを置く



② ブロックを組み合わせる

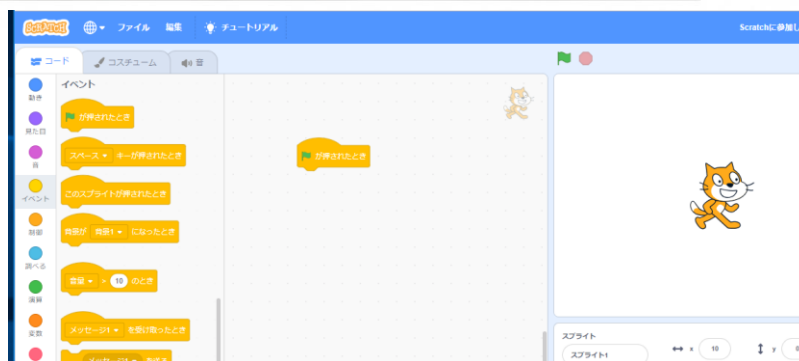


③ プログラムの起動



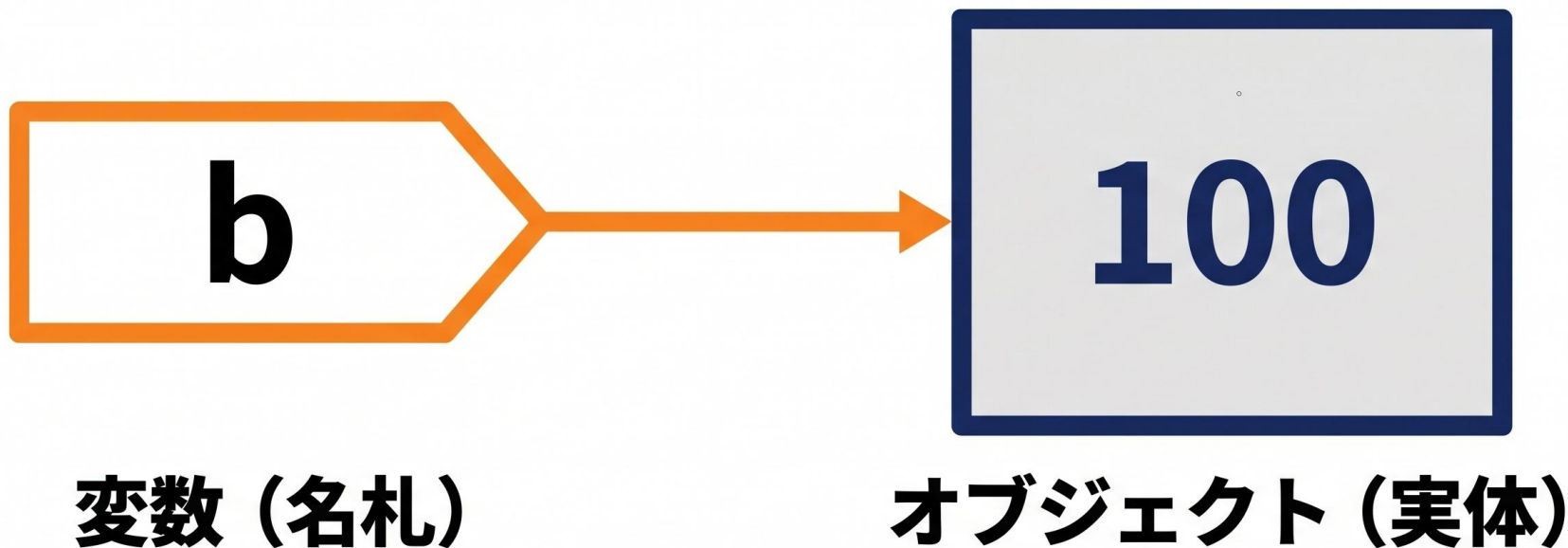
ブロックの動作が
順に実行される

実際の画面





変数は『名札』、オブジェクトは『実体』。名札が実体を指し示す。



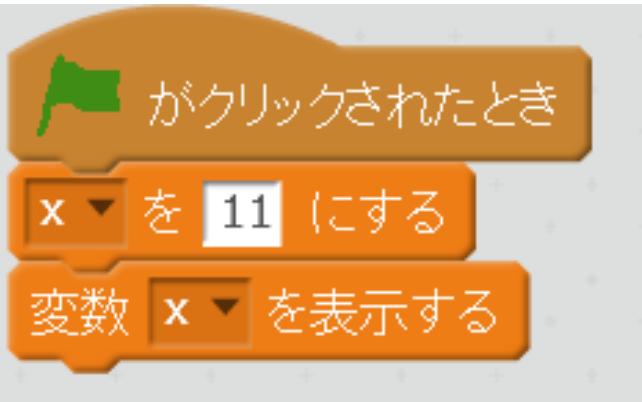
① データに名前を付けて保管する

② 変数 (名札) は実体を指し示すだけ

③ 必要に応じて書き換えられる



Scratch での変数



変数 x を使うような
プログラム



変数 x の値の表示結果

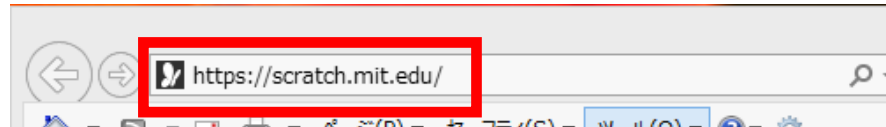
演習



- パソコンにログインする
- Webブラウザを起動する
- Webブラウザで、次のURLを開く

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

Webブラウザの起動



- 「やってみよう」をクリック

※ 次ページに続く

Scratch 作る 見る ヒント Scratchについて 検索 Scratchに参加しよう サインイン

物語やゲーム、アニメーションを作って
世界中の人と共有しましょう

Scratchに参加する
無料です!

30,608,728 プロジェクトが共有されているクリエイティブ・ラーニング・コミュニティ

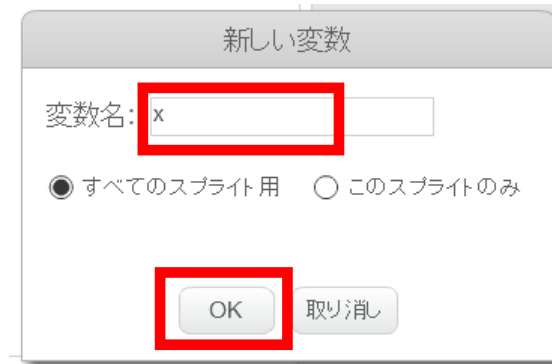
演習



5. 「データ」をクリックし、「変数を作る」をクリック



6. 変数名を「x」。(半角の x です)
「OK」をクリック。



※ 次ページに続く

演習



7. 「x を 0 にする」と「変数 x を表示する」を置き、この2つを組み合わせる



2つを
組み合わせる

※ 次ページに続く

8. 「イベント」をクリック。「 がクリックされたとき」を組み合わせる



The image shows the Scratch interface. On the left, the 'Events' (イベント) menu is highlighted with a red box. It contains several options: a green flag icon for 'when clicked' (がクリックされたとき), a space key for 'when space key is pressed' (スペース キーが押されたとき), 'when this sprite is clicked' (このスプライトがクリックされたとき), and 'when backdrop becomes backdrop1' (背景が backdrop1 になったとき). On the right, the script area shows a sequence of three blocks: 'when clicked' (green flag icon), 'set x to 0' (x を 0 にする), and 'show variable x' (変数 x を表示する). A red arrow points from the 'when clicked' block in the menu to the 'when clicked' block in the script area. The entire script area is enclosed in a red box.

組み合わせる

※ 次ページに続く

演習

9. 実行ボタンをクリックすると、「x 0」のように表示されるので確認する



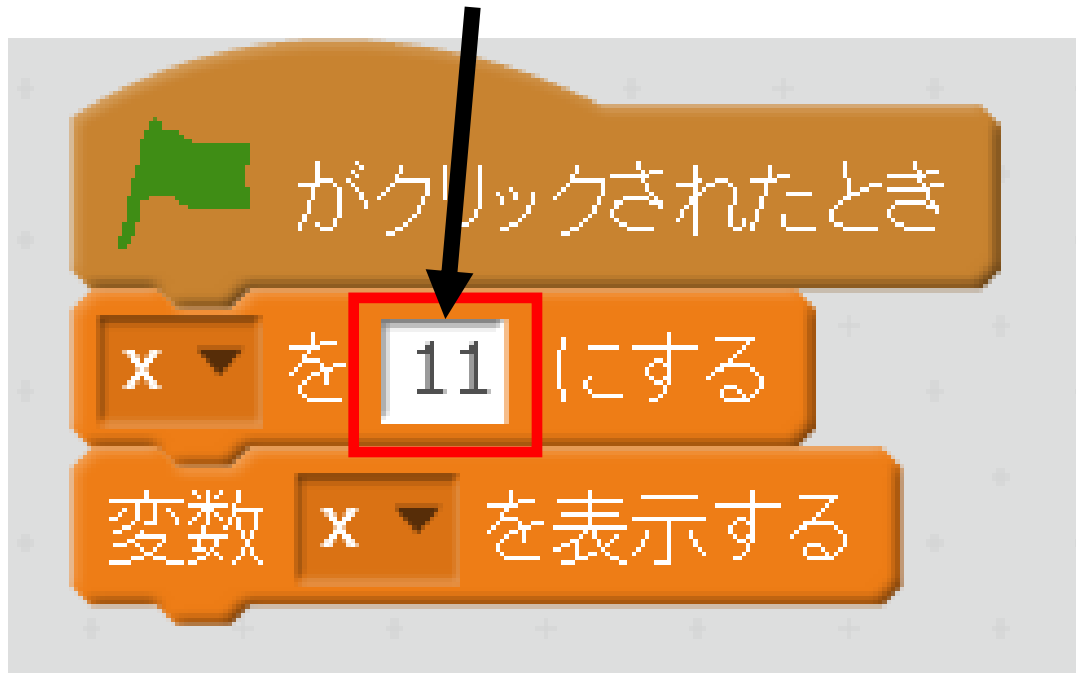
※ 次ページに続く

演習



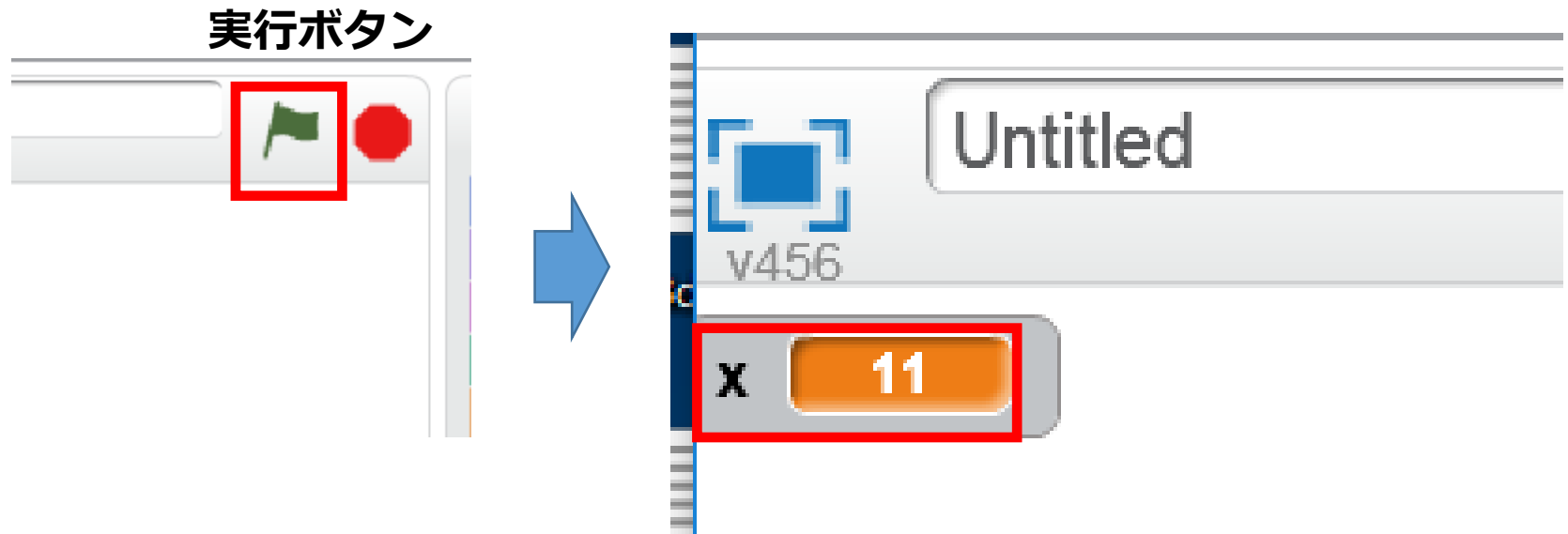
10. 「x を 0 にする」を「x を 11 にする」に書き換えてみる

※ 半角の「11」（全角の1 1では動かない）



※ 次ページに続く

11. 実行ボタンをクリックすると、「x 11」のよう
に表示されるので確認する



※ 次ページに続く

演習

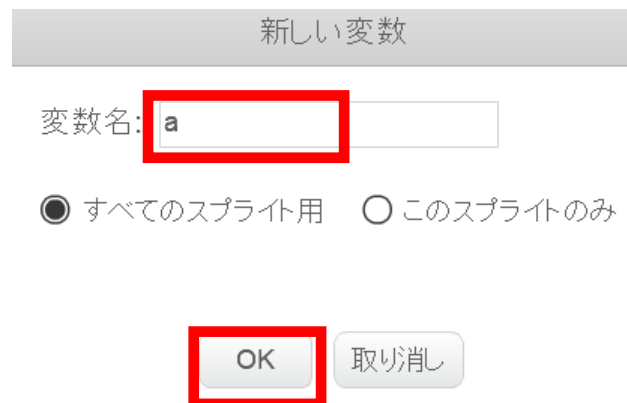


新しい変数 a を追加し，値を 100 に設定したい。

12. 「データ」をクリックし，「変数を作る」をクリック



13. 今度は，変数名を「a」．（半角の a）
「OK」をクリック。



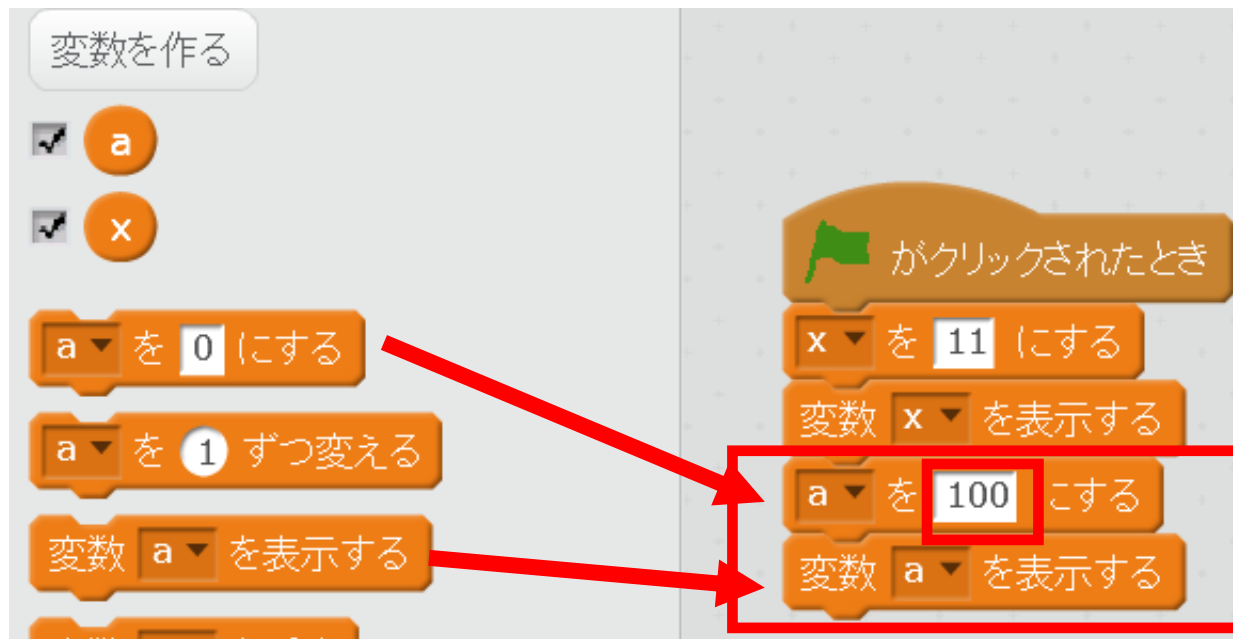
※ 次ページに続く

演習



14. 「aを0にする」と「変数aを表示する」を組み合わせる.

そして、「0」のところを「100」に書き換える（半角の「100」です）

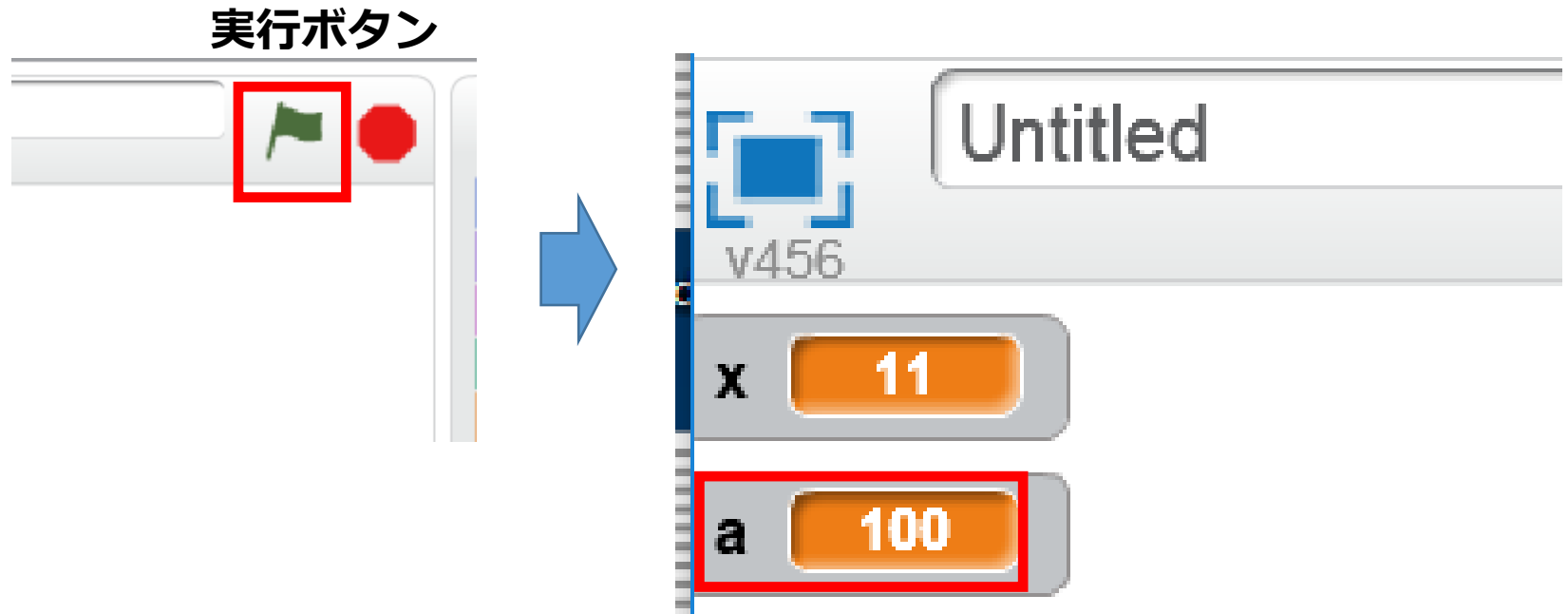


※ 次ページに続く

演習



15. 実行ボタンをクリックすると, 「x 11」, 「a 100」のように表示されるので確認する



※ 次ページに続く

演習問題



- ◆新しい変数 b を追加し，値を 200 に設定しなさい
- ◆新しい変数 r を追加し，値を 1.08 に設定しなさい
- ◆新しい変数 z を追加し，値を -10 （マイナス10）に設定しなさい
- ◆下の図、左のように組み立てなさい

がクリックされたとき

- x を 11 にする
- 変数 x を表示する
- a を 0 にする
- 変数 a を表示する
- b を 200 にする
- 変数 b を表示する
- r を 1.08 にする
- 変数 r を表示する
- z を -10 にする
- 変数 z を表示する

x	11
a	100
b	200
r	1.08
z	-10

実行結果を確認
しなさい



4. 式に関する演習

変数が役に立つのは



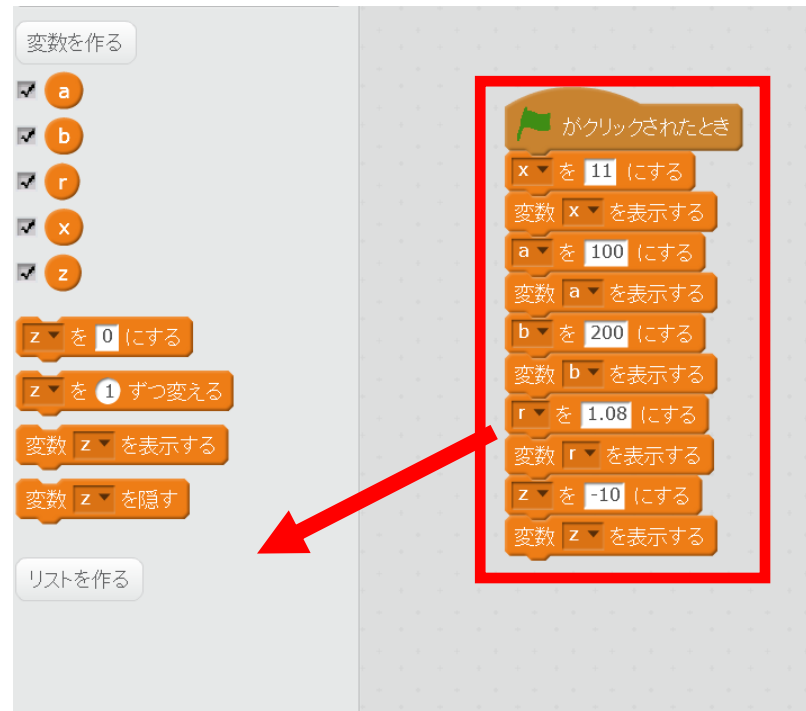
- ◆ データを記憶したいとき
- ◆ 同じような計算などを，値を変えながら何度も行いたいとき

演習



1. さきほど作成したブロックは不要なので、ブロックを

マウスの右ボタンを押しながら、中央エリアにドラッグする。



ドラッグすると
消える

※ 次ページに続く

演習



2. 「イベント」をクリック。下の図のようにブロックを置く。



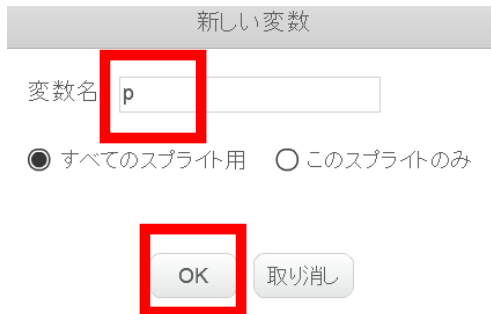
※ 次ページに続く

演習



3. 「データ」をクリック。
新しい変数 p を作る

4. 下の図のようにブロックを組み合わせる。



※ 次ページに続く

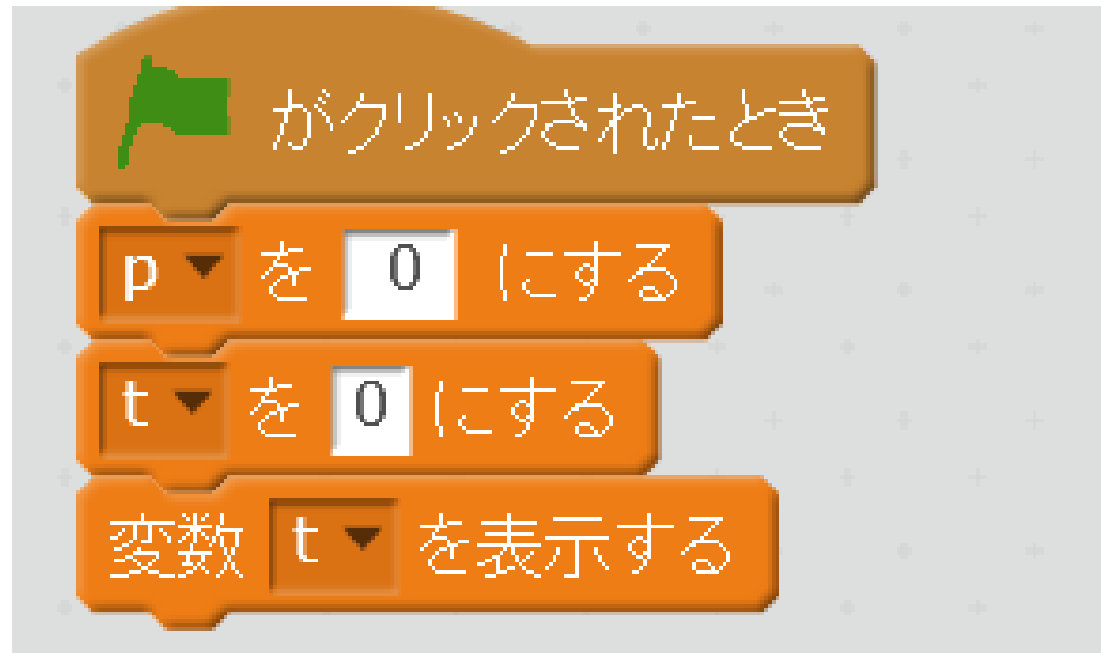
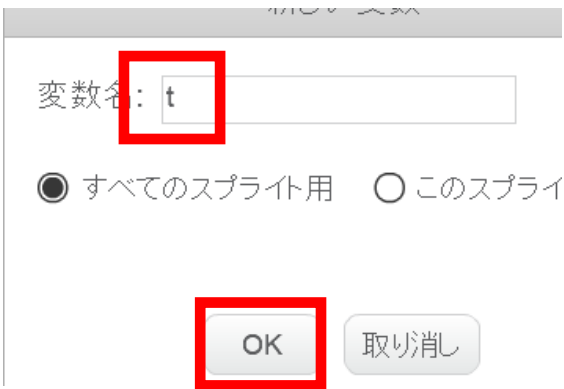
演習



5. 「データ」をクリック.

新しい変数 t を作る

6. 下の図のようにブロックを組み合わせる.



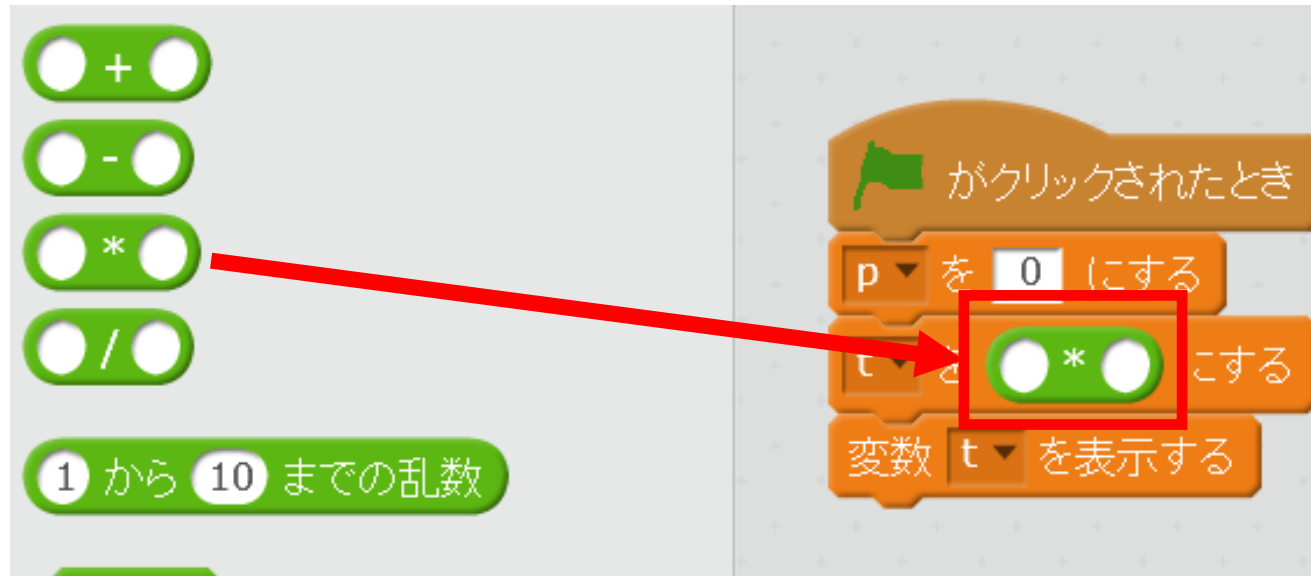
※ 次ページに続く

演習



7. 「演算」をクリック。

8. 「○*○」のブロックを、
下の図のように組み合わせる。

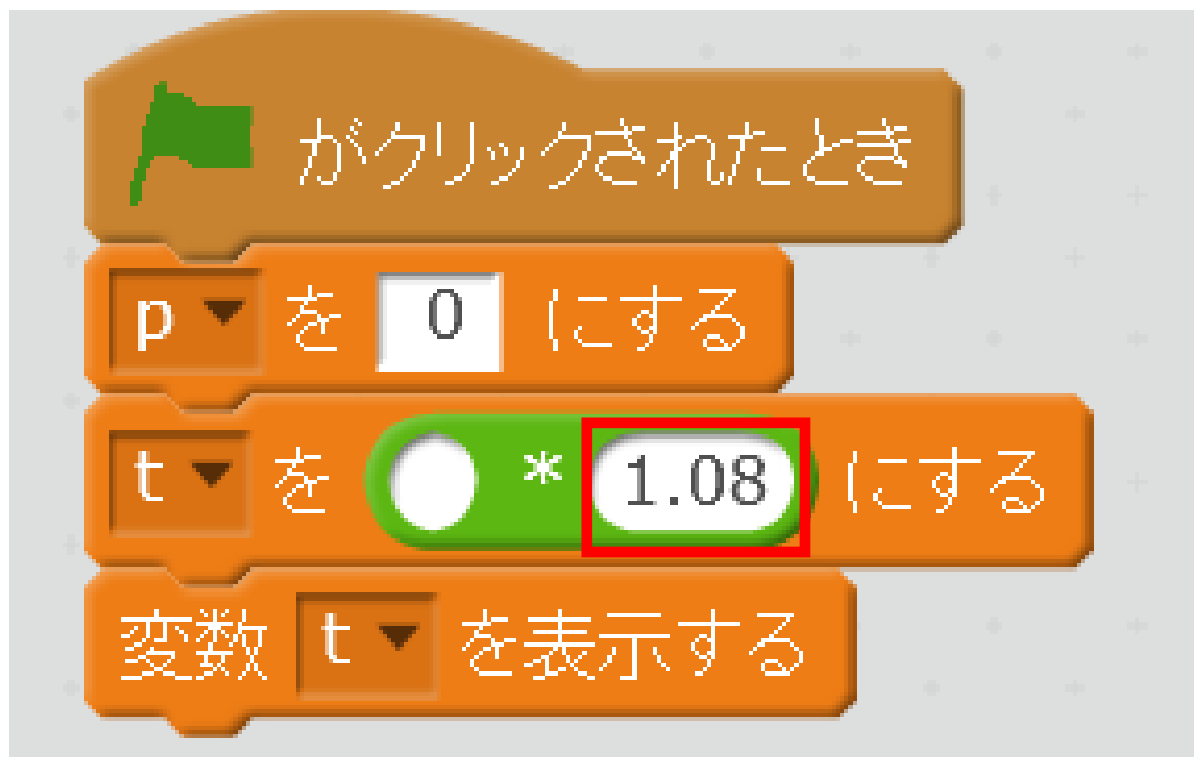


※ 次ページに続く

演習



9. 「1.08」のように書き換える. 「1.08」も半角



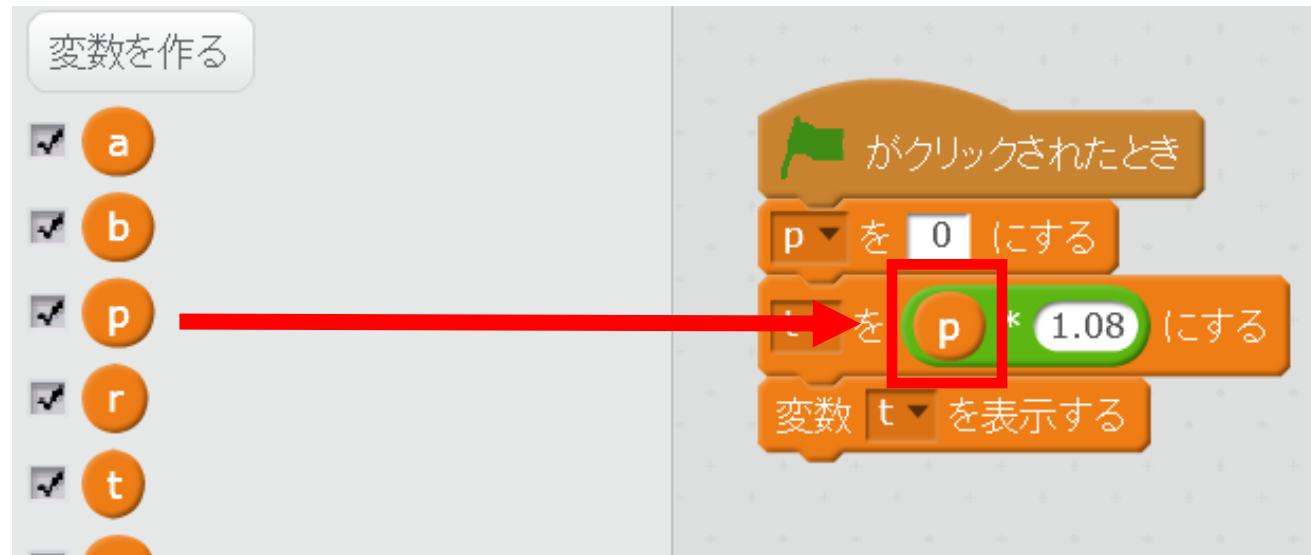
※ 次ページに続く

演習



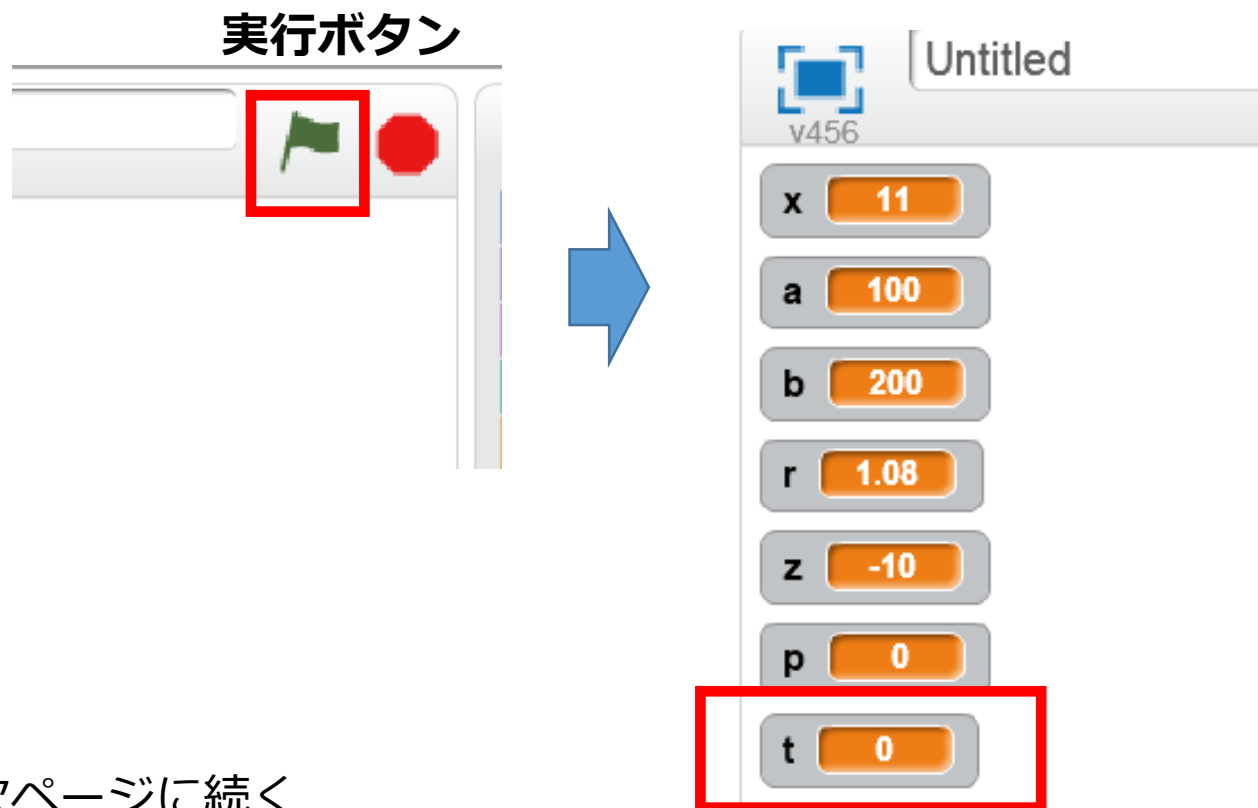
10. 「データ」をクリック.

11. 下の図のように, 変数 p のオレンジ色のブロックをはめ込む



※ 次ページに続く

12. 実行ボタンをクリックすると、「t 0」のように表示されるので確認する



※ 次ページに続く

演習



13. 変数 p の値を 100 に変えて, $100 * 1.08$ の値を
求めてみる



※ 次ページに続く

演習



14. 実行ボタンをクリックすると、「t 108」のように表示されるので確認する

実行ボタン



x	11
a	100
b	200
r	1.08
z	-10
p	100
t	108

※ 次ページに続く

演習課題



変数の値を変えて、同じ計算を再実行

◆ 変数 p の値を 1000 に変えて、 $1000 * 1.08$ の値を求めなさい

◆ 変数 p の値を 2000 に変えて、 $2000 * 1.08$ の値を求めなさい

◆ 変数 p の値を 4000 に変えて、 $4000 * 1.08$ の値を求めなさい

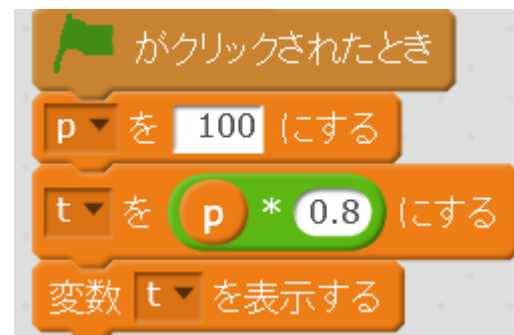
※ 次ページに続く

演習課題



2 割引きの価格を求めたい

◆ 右のようにプログラムを作りなさい



◆ 変数 p の値を 100 に変えて, 変数 t の値を確認



◆ 変数 p の値を 4500 に変えて, 変数 t の値を確認

◆ 変数 p の値を 8000 に変えて, 変数 t の値を確認

演習課題



正方形の面積を求めたい

◆ 下のようプログラムを作りなさい

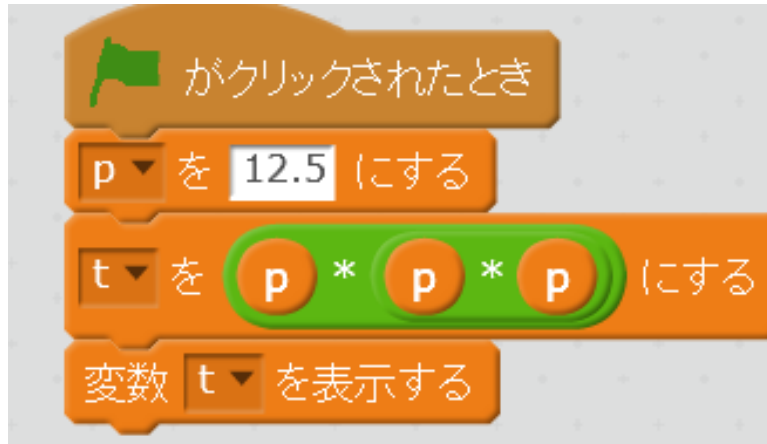


- ◆ 変数 p の値を 10 に変えて、変数 t の値を確認
- ◆ 変数 p の値を 25 に変えて、変数 t の値を確認
- ◆ 変数 p の値を 12.5 に変えて、変数 t の値を確認

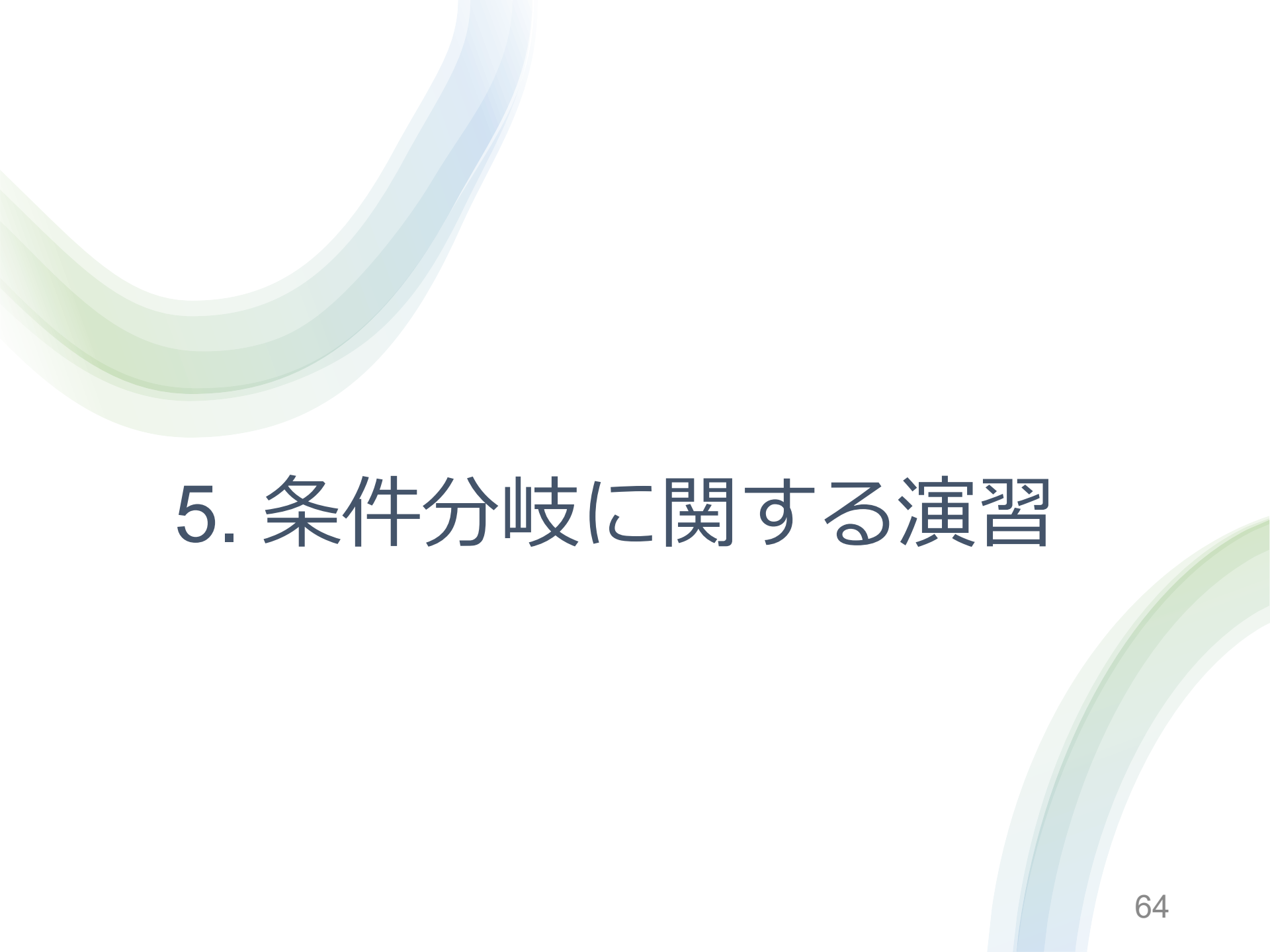
各自への演習課題

立方体の体積を求めたい

◆ 下のようプログラムを作りなさい



- ◆ 変数 p の値を 10 に変えて、変数 t の値を確認
- ◆ 変数 p の値を 25 に変えて、変数 t の値を確認
- ◆ 変数 p の値を 12.5 に変えて、変数 t の値を確認



5. 条件分岐に関する演習

条件分岐



変数や式の値によって、結果が変わる

例えば

age の値が 20未満 → 100 yen

20以上 → 200 yen

演習



- 今まで作成したブロックは不要なので、ブロックを
- マウスの右ボタンを押しながら、中央エリアにドラッグする。

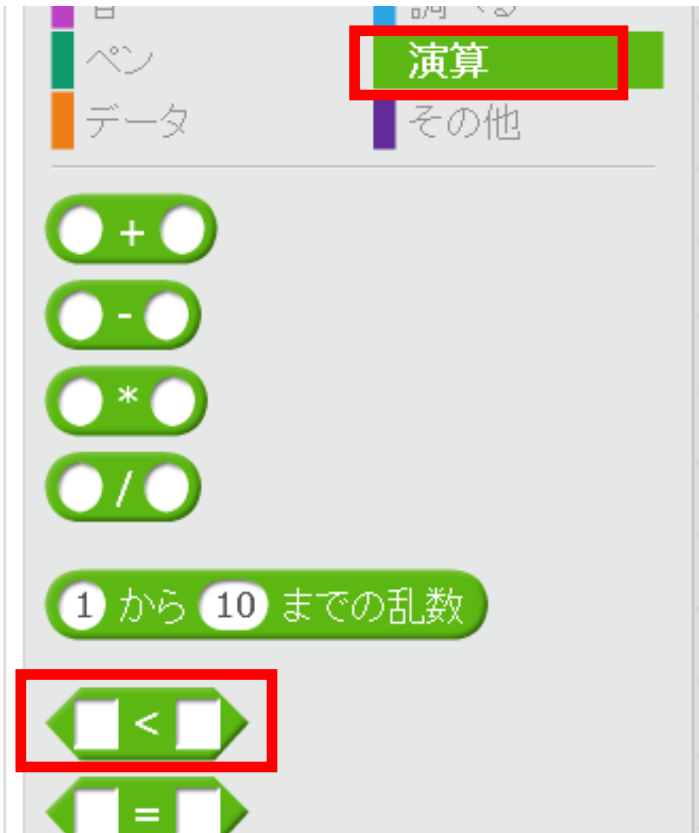


ドラッグすると
消える

演習



- 次のブロックを使う



演習



次の2つの変数を作る

- age
- price

演習



ブロックを下ののように組み立てる

```
whenClickedClicked
setAge0
ifAgeLessThan20
  setPrice100
  setPrice200
showPrice
```

演習



- 変数 age の値を 18 にして, 実行ボタンを押すと, price の値が 100 になることを確認

がクリックされたとき

age を 18 にする

もし age < 20 なら

price を 100 にする

でなければ

price を 200 にする

変数 price を表示する

実行ボタン



age 18

price 100

演習



- 変数 `age` の値を 30 にして, 実行ボタンを押すと, `price` の値が 200 になることを確認

Scratch script showing the logic for the exercise:

- がクリックされたとき
- age を 30 にする
- もし age < 20 ならば
- price を 100 にする
- でなければ
- price を 200 にする
- 変数 price を表示する

実行ボタン



age 30

price 200

各自への演習問題

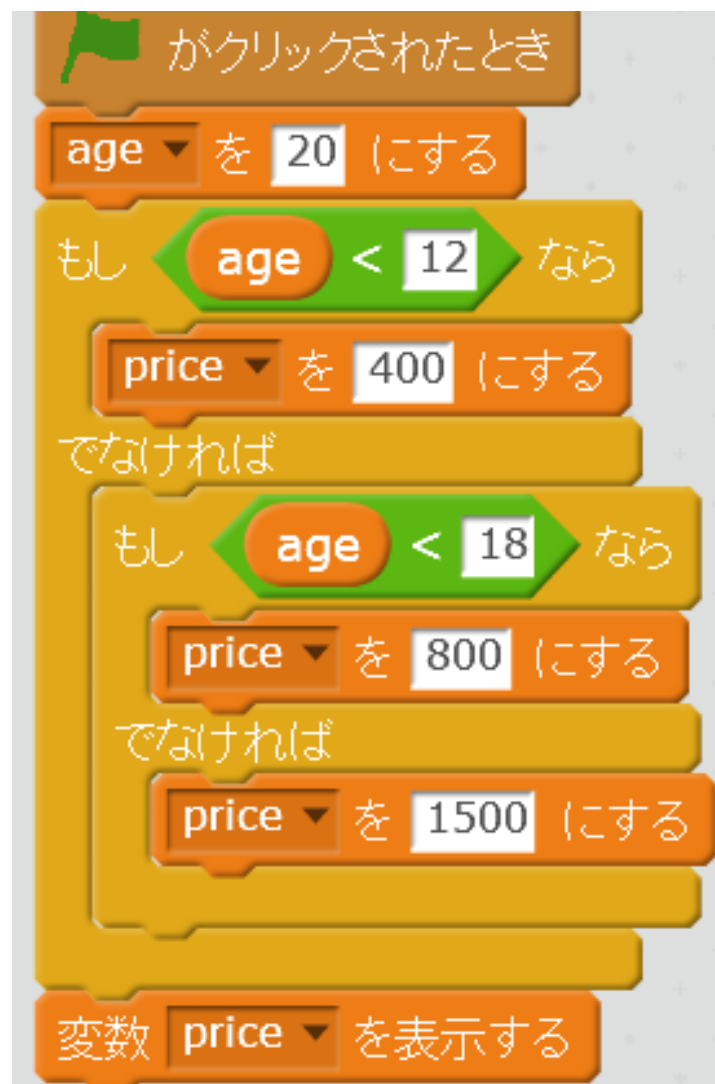


ある映画館は

- 12歳未満 400円
- 12歳以上18歳未満 800円
- 18歳以上 1500円

• 右のようにブロックを
組み立てなおしなさい

• そして、変数 age の値を
10, 15, 20 と変えて、実行



演習問題

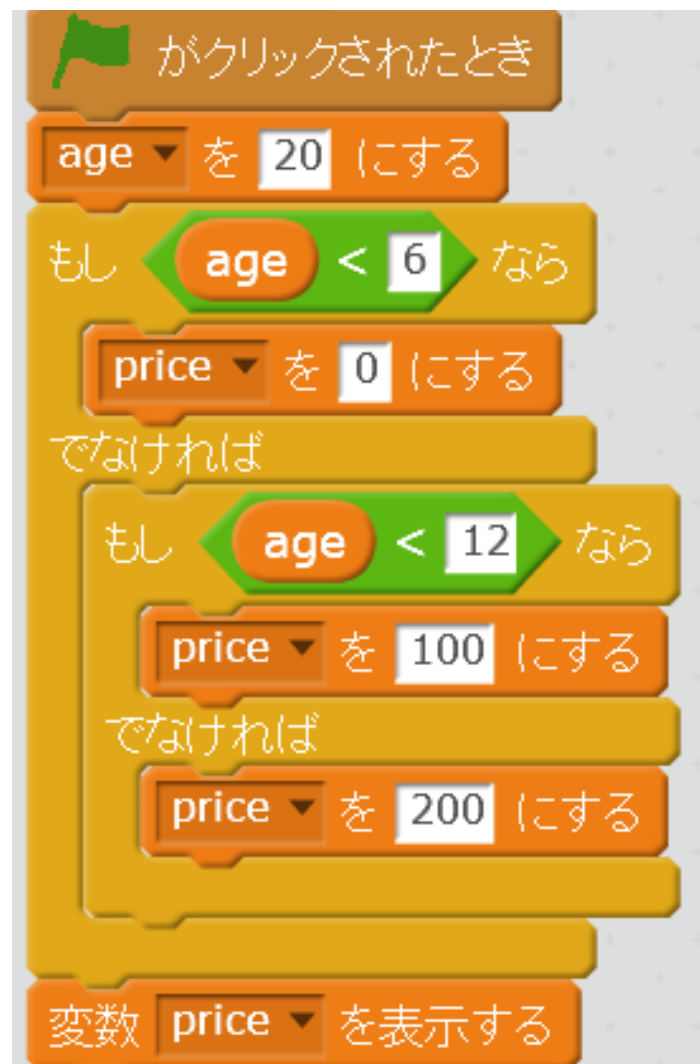



あるバスは

- 6歳未満 0円
- 12歳未満 100円
- 12歳以上 200円

• 自分で考えて、ブロックを組み立てなおしなさい

• そして、変数 age の値を 5, 10, 15, 20 と変えて、実行してみなさい





5. 繰り返しに関する演習

繰り返し



- 同じような処理を繰り返すこと.
- 変数の値を変えながら, 繰り返すのが定石

• 新しいブロックを使う



キャラクターを動かす

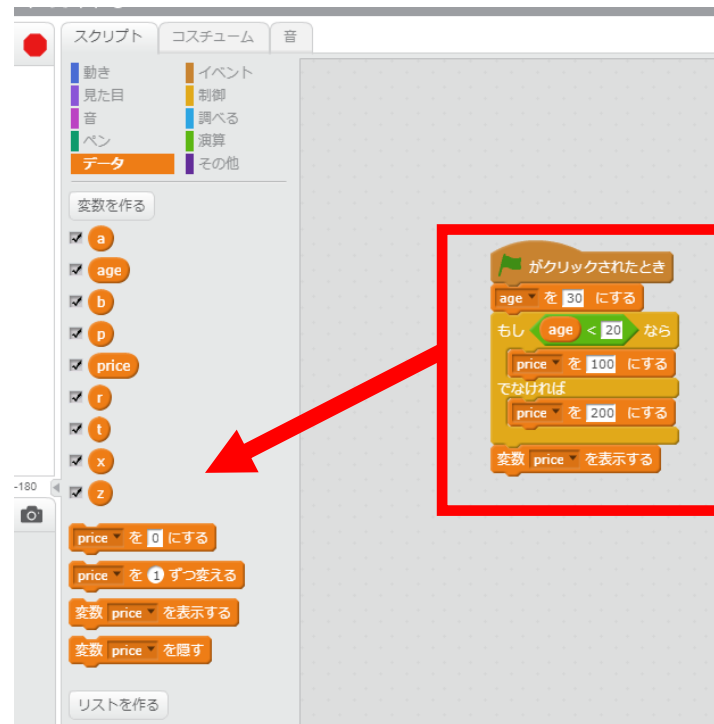


繰り返す

演習



- 今まで作成したブロックは不要なので、ブロックをマウスの右ボタンを押しながら、中央エリアにドラッグする。



ドラッグすると
消える

※ 次ページに続く

演習



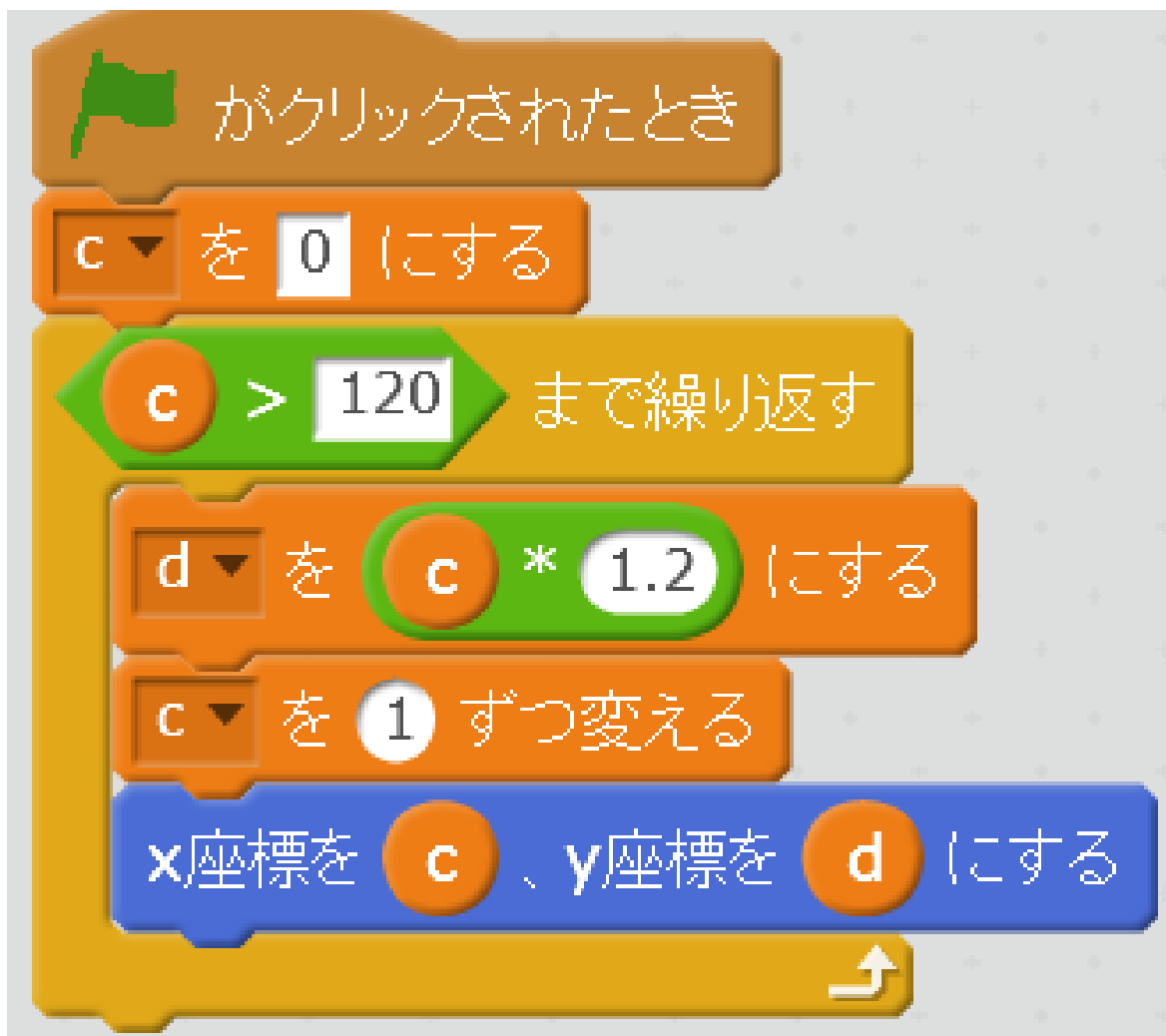
次の2つの変数を作る

- c
- d

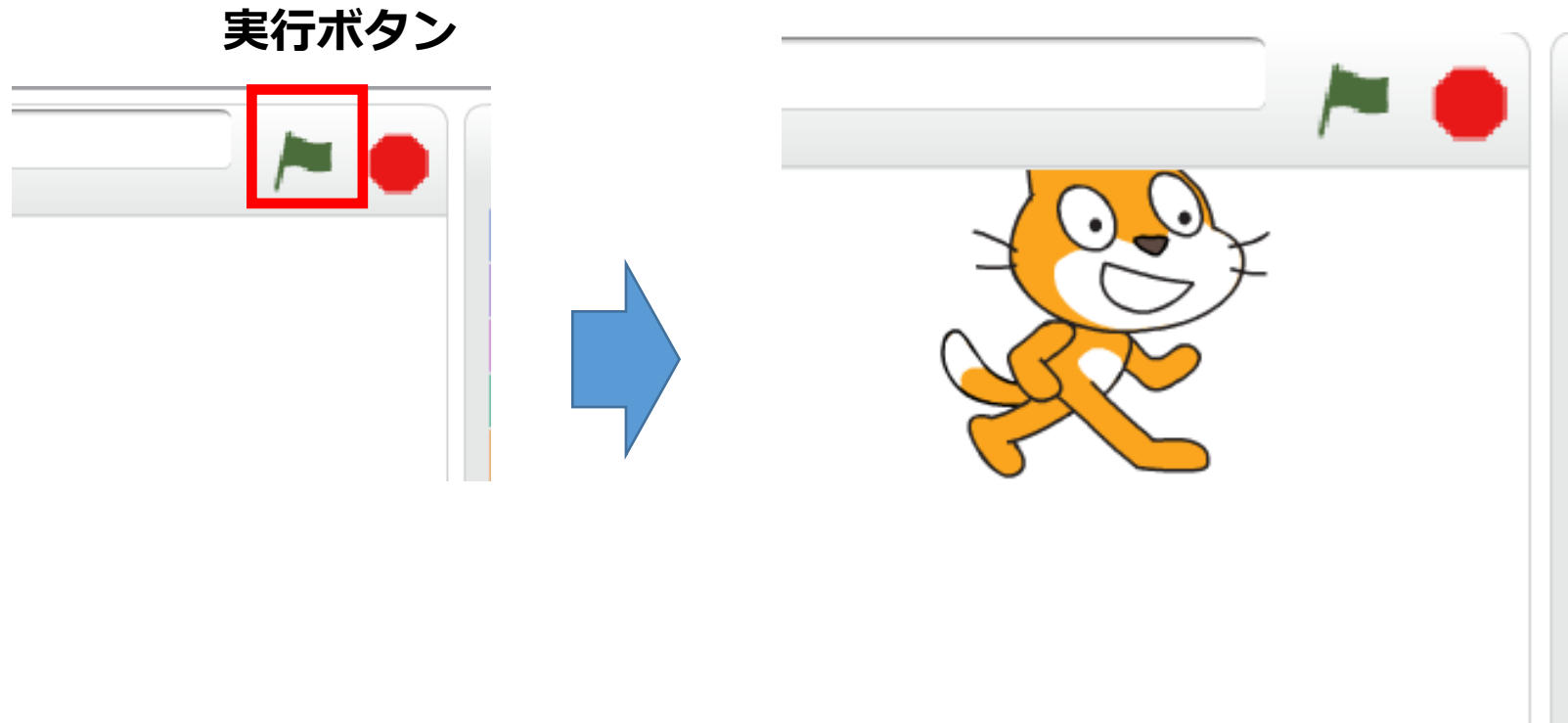
演習



- ブロックを下のように組み立てる



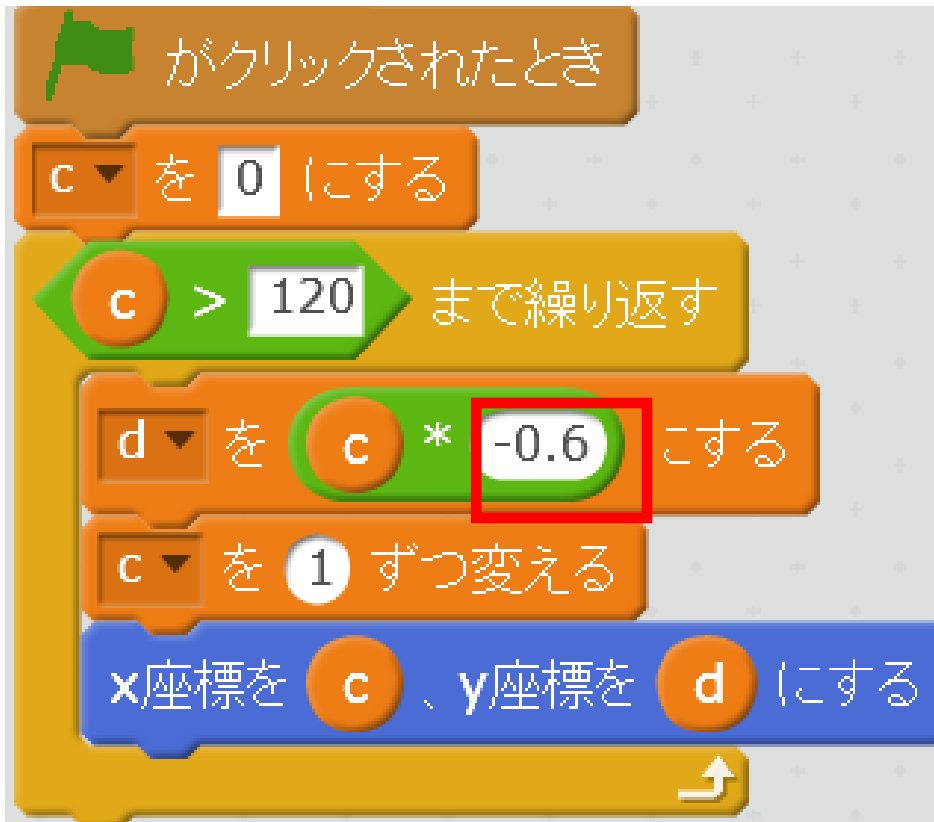
- 実行ボタンを押すとキャラクターが動くことを確認



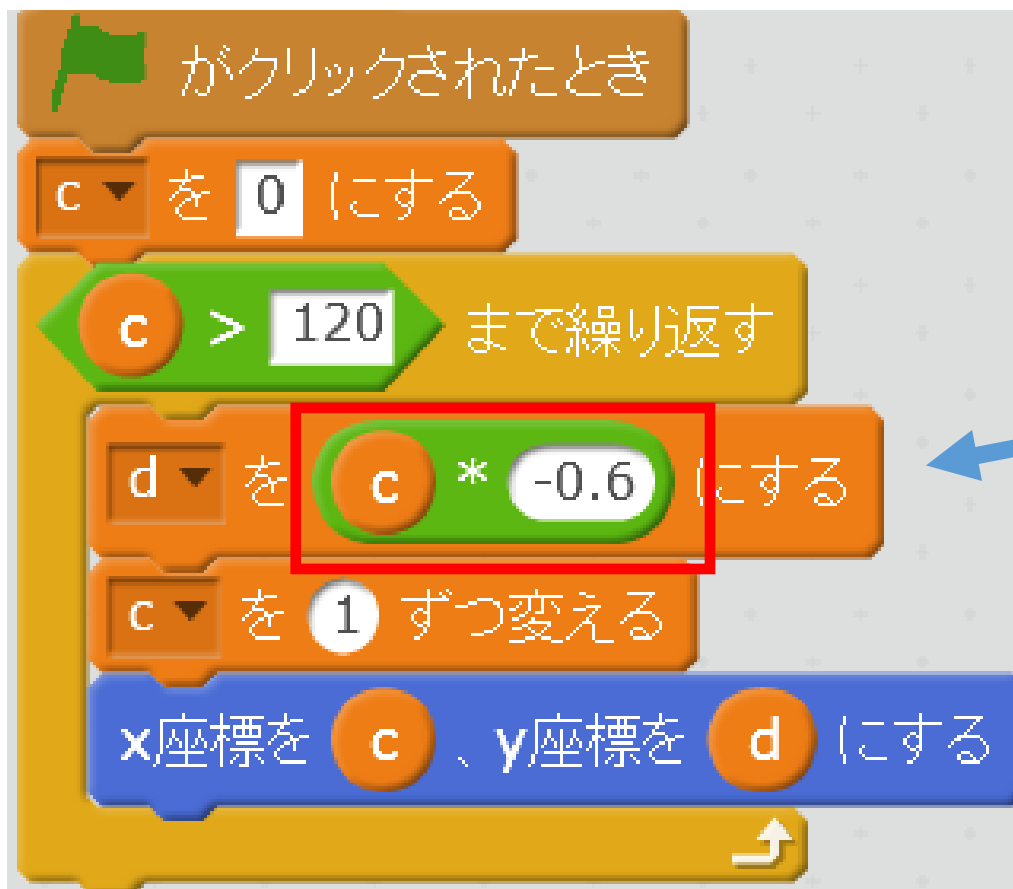
演習



- 「-0.6」のように書き換えて，もう1度実行してみなさい



まとめ



変数 c の値が変わっても、
いつも
 d の値は c の -0.6 倍にしたい

演習

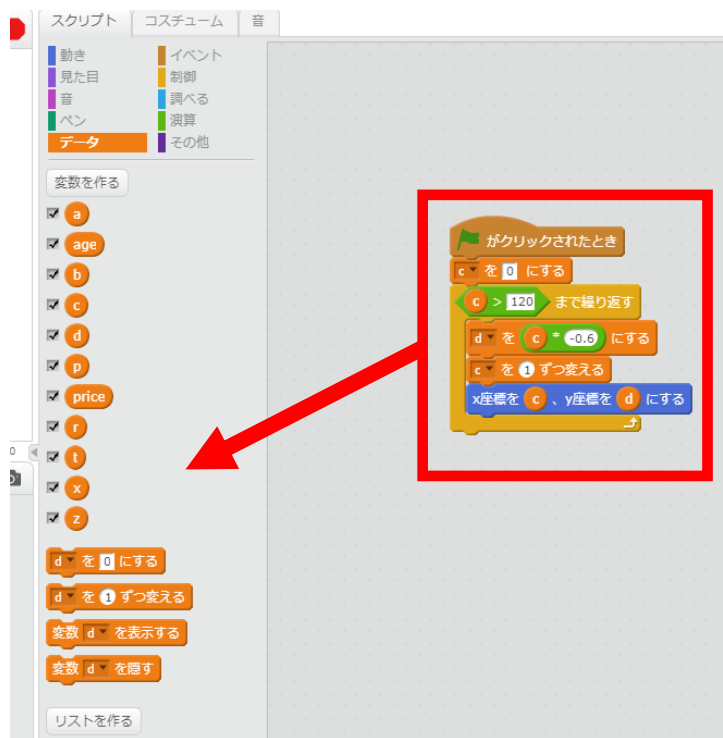
- 今度は、次のブロックを使う



演習



- 今まで作成したブロックは不要なので、ブロックをマウスの右ボタンを押しながら、中央エリアにドラッグする。



ドラッグすると
消える

演習



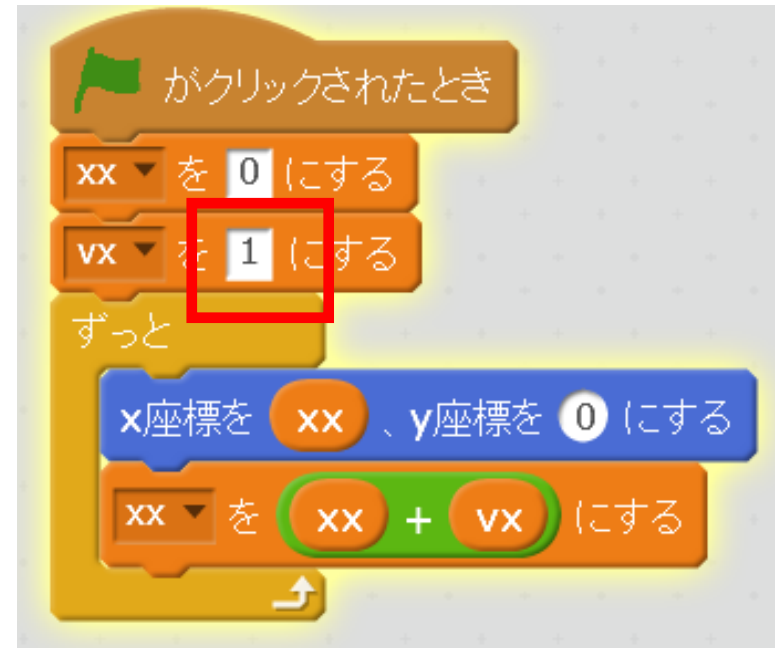
次の2つの変数を作る

- XX
- VX

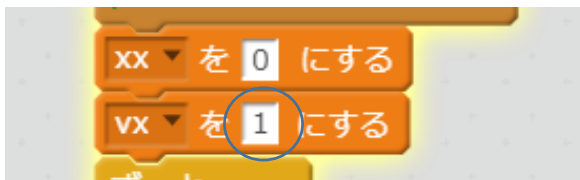
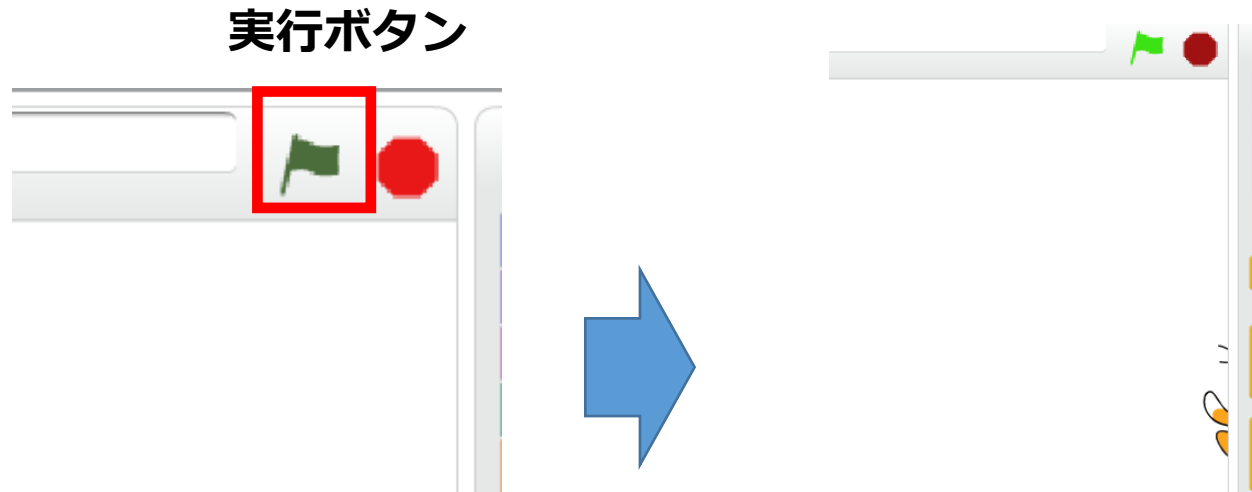
演習



- 「ねこ」のキャラクタを、
- 速度： 1
に設定して、動かしていく。
 - 一度、右のようにブロックを組み立てなおしなさい



- 実行ボタンを押すとキャラクターが動くことを確認
(キャラクターが右端まで行くと、プログラムが自動で止まる)



「vx を 4 にする」、 「vx を 0.1 にする」、
「vx を -1 にする」など、いろいろ変えてみなさい