

# 小学生から始める プログラミング教室



Kid's  
programmer  
-master course-

## ベーシックコース

ゆうしゃ ぼうけん  
**勇者の冒険** スクラッチ 3.0 ver.  
～クローンを利用して要素を増やそう～



ぎゃくてん  
“逆転のアイデア”で  
しゅじんどう  
主人公を動かすしくみを  
作るよ！

1 回目授業	年	月	日
2 回目授業	年	月	日

名前

練習ファイルには、必要に応じて  
一部の命令ブロックが記述してある場合があります。



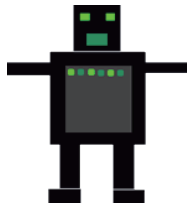
- ◆文科省 ICT活用教育アドバイザー事務局掲載  
学校ICT化サポート事業者
- ◆長期コースによる、プログラミングの普及

# 変数

## 変数とは

プログラムの世界では、状況にあわせて変化させたい値を「変数」として決めることができます。例として「ゲームの制限時間」や「体力」などがあげられます。

### 変数例



### 主人公 A の変数例

- 体力：100
- 素早さ：100
- 攻撃力：100
- 守備力：100

### 変数例



### 主人公 B の変数例

- 体力：200
- 素早さ：200
- 攻撃力：200
- 守備力：200

キャラクターごとに変数の値を変えることで、設定値を変えることができます。

## 作り方

①クリック

②クリック

③入力

④クリック

⑤確認

手順⑤のように、変数ができていれば成功です。  
✓を入れると、画面上で変数の値を確認できます。

## 表示の種類について

ゲーム画面の変数上で  
右クリック

ひょうじほうほう  
表示方法を変えられます。

普通の表示

大きな表示

スライダー



変数は、スクラッチ 1.4 ver. にもありました。プログラミングを実行する上で必要な構文となるので必ず覚えて、使えるようになりましょう。



# 関数

## ■ 関数とは

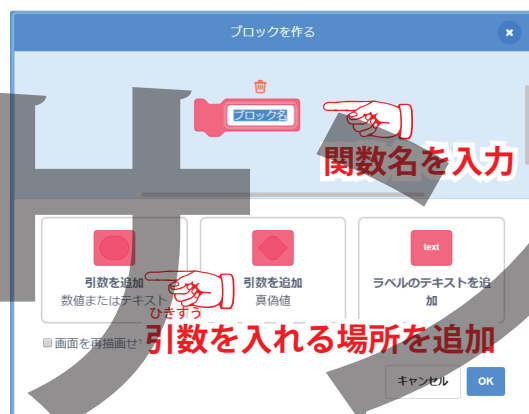
スクラッチ 3.0 ver. では、オリジナル関数（めいれい命令のかたまり）を定義（ていぎ）することができます。  
定義するには、下のようになります。

### ■ 作り方

#### ① 「ブロックを作る」をクリック

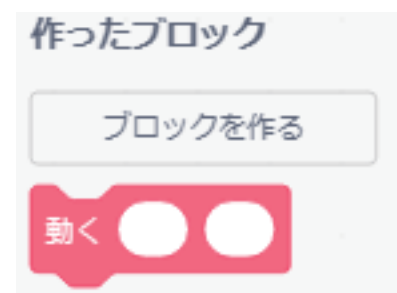


#### ② 関数を定義



ブロック名・・・関数名  
ひきすう  
引数・・・関数に使う材料  
ざいりょう

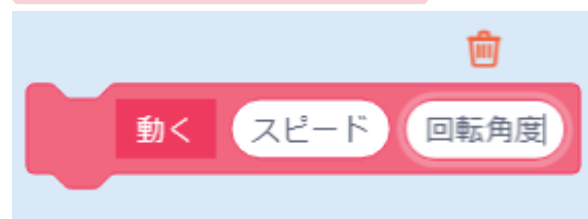
#### ③ 確認



### ■ 使い方

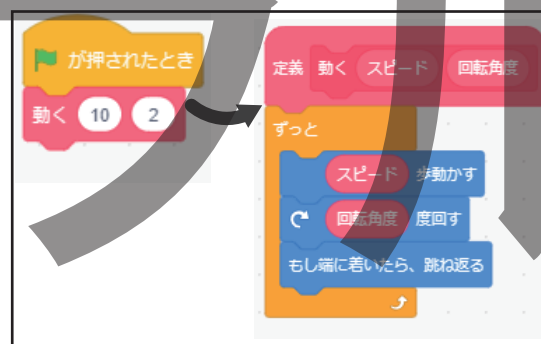
#### 作成例

関数名「動く」  
引数「スピード」「回転角度」  
かいてんかくど

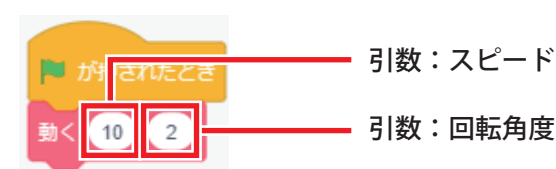


#### 使用例

旗がクリックされたとき、  
「動く関数」が実行されます。



引数に値を入力して、関数を実行させます。



#### 動作例

対象のSpriteが、  
10 歩ずつ動きながら、2 度ずつ回転しながら  
いどう  
移動し続けます。



関数のブロックを使えるようになると、プログラムをより短く書くことができます。



## 完成ゲームを確認しよう

「チャレンジ > 2日目\_チャレンジ完成.sb3」を開こう

そうさてじゅん

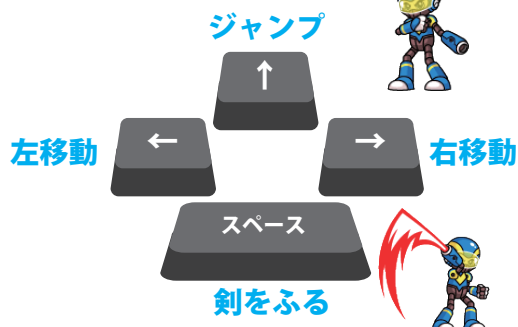
### 操作手順1 旗を押してゲームをスタートします

主人公を動かして、60秒でどれだけ「宝石」を取れるか挑戦するゲームです。

体力が1より少なくなったり、ステージから落ちたりしたら、ゲームオーバーです。



宝石の数 10



### 操作手順2 宝石や回復アイテムを、たくさん取ります

宝石・敵・回復アイテムは、ランダムに出てきます。

剣をふって、敵を“げきは”（うちやぶること）したときも、宝石の数が増えます。

ゲット!

宝石

…3種類の宝石がランダムに出てきます

てき 敵

…げきはできると、宝石が増えます  
できなかったときは、主人公の体力が減ります **ヒット**

回復アイテム

…主人公の体力を取りもどします

⇒ 「攻撃力」や「回復量」はそれぞれ違います  
(2日目チャレンジで作ります)

こうげきりょく (主人公の体力を減らす数)

5 10 15

回復量 (主人公の体力を増やす数)

5 10

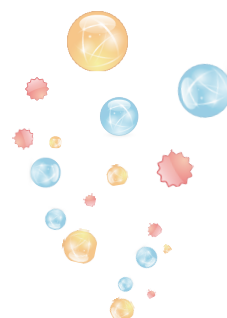
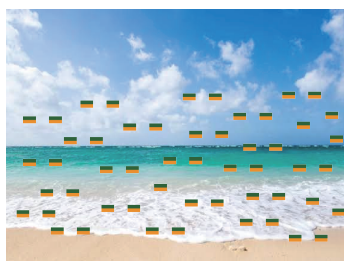
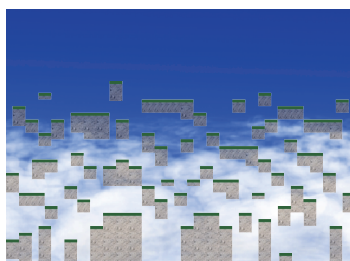
### 操作手順3 取った宝石の数が10を超えたら、次のステージに進みます

10

宝石の数

⇒ 主人公のスピードが速くなります

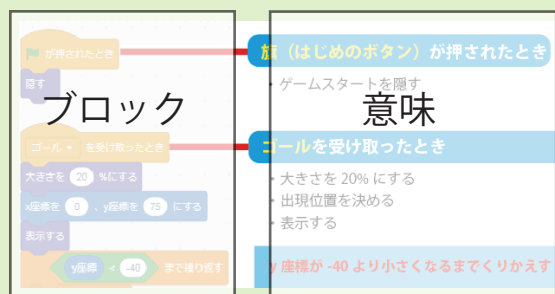
⇒ ステージの背景や、陸地の形が変わります





## テキストの見方

ブロックの横に、その意味が書いてあります

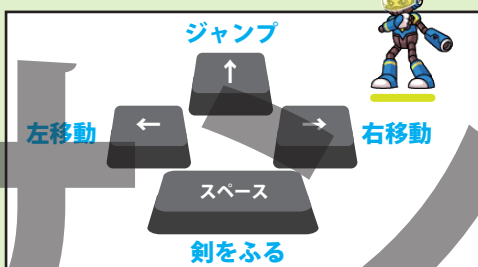


## 学習の手順

1

主人公の基本動作を作ります。  
色（■：足 ■：地面）によって  
動きを分けるプログラムにします。

左・右・上キーに合わせて、  
主人公を動かします。



ジャンプ	落下動作
↑ 60	↓ -3 ↓ -3 ↓ -3

### 学習項目

- 色によって、条件分岐を行うしくみ
- キーで主人公を動かすしくみ



1日目  
3章

2

$3 \times 1 = 3$  (右方向に移動)

$3 \times -1 = -3$  (左方向に移動)

左・右キーを押して、陸地をキーと反対の方向へ移動させます。  
こうすることで、主人公が動いているように見えます。

### 学習項目

- 陸地を移動させるしくみ  
(後ろのものを動かすことで、手前のものが動いているように見せる)



1日目  
4章

3



### 学習項目



- 重要  
クローンを使って要素を増やすしくみ



1日目  
5章

2日目  
6章 ~

ほかのアイテムも、陸地にそって移動するしくみをつくります。  
また、クローンを使って要素を増やす方法を学びます。  
取った宝石の数が10を超えたら主人公の速さや背景を変えるプログラムも、追加します。



## 使用変数 / スプライト

### 使用変数

#### p.1 「変数」を確認

- 主人公のスピード ..... 主人公が移動するスピード
- 主人公の体力 ..... 主人公の今の体力
- 宝石の数 ..... 取った宝石の数
- 残り時間 ..... ゲームをプレイできる時間
- 向き ..... 主人公が向いている方向
- ★敵の攻撃力 ..... 主人公の体力を減らす数（「敵」だけにある変数）
- ★回復量 ..... 主人公の体力を増やす数（「回復アイテム」だけにある変数）

### スプライト

- 主人公 ..... 陸地を冒険する勇者。左右に進んだりジャンプしたりできます
- 陸地 ..... 主人公やアイテム、敵が動き回る地面
- 宝石 ..... 主人公が集める、きれいな石
- 回復アイテム ..... 主人公の体力を取りもどすアイテム
- ゲームオーバー ..... 主人公の体力がなくなったり、ステージから落ちたときの表示
- タイムアップ ..... 残り時間がなくなったときの表示
- 敵 ..... 主人公にダメージを与えるモンスター
- スピードアップ ..... 宝石の数が 10 より多くなったときに出てくる表示



### コスチューム

スプライトには「コスチューム」と呼ばれる、見た目があります。  
それぞれのコスチュームはすでに作ってあるので、確認してみましょう。

●スピードアップ	●陸地	●主人公	●タイムアップ	●ステージ	●敵	●宝石	●回復アイテム
1 スピードアップ 304 x 57	1 陸地1 490 x 248	1 右向き 83 x 117	1 タイムアップ1 418 x 79	1 背景1 480 x 360	1 敵1 180 x 115	1 宝石1 98 x 98	1 回復1 100 x 100
	2 陸地2 454 x 175	2 左向き 82 x 117	●ゲームオーバー 1 ゲームオーバー 188 x 29	2 背景2 480 x 360	2 敵2 129 x 116	2 宝石2 92 x 92	2 回復2 100 x 100
		3 右向き 127 x 148			3 敵3 70 x 103	3 宝石3 100 x 100	3 体力回復 72 x 98
		4 右向き 145 x 172			4 ヒット 200 x 200	4 ゲット 262 x 184	
		5 右向き 135 x 160					
		6 左向き 128 x 148					
		7 左向き 148 x 173					
		8 左向き 135 x 161					



# 1日目

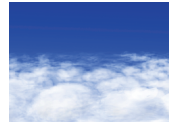
## 1

## ステージと画面表示の作成

### 1

### ステージを選ぼう

ステージでは、ゲームに必要な変数を決めます。  
残り時間をカウントするしくみも作ります。



ステージ

旗が押されたとき

背景を 背景1 にする

主人公の体力 を 50 にする

主人公のスピード を 3 にする

宝石の数 を 0 にする

残り時間 を 60 にする

ずっと

1 秒待つ

残り時間 を -1 ずつ変える

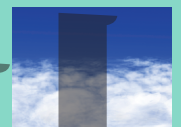
もし 残り時間 = 0 なら

タイムアップ を送る

旗（はじめのボタン）が押されたとき

- ・背景を「背景1」にする
- ・主人公の体力を 50 にする
- ・主人公のスピードを 3 にする
- ・宝石の数を 0 にする
- ・残り時間を 60 にする

背景 1



ずっと

- ・1秒ごとに
- ・残り時間を1ずつ減らす

もし 残り時間が 0 なら

- ・タイムアップを送る

## プログラムを確かめよう

旗をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 旗を押したときに、残り時間が1秒きざみで減りましたか？

60 ~  
すうじ  
数字が減る

残り時間

0



## 2

## タイムアップ と ゲームオーバーの作成

### 1 タイムアップを選ぼう

ステージから「タイムアップ」の命令が送られてくると、表示されます。

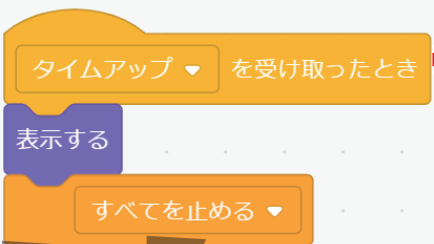
タイムアップ!!

タイムアップ



旗（はじめのボタン）が押されたとき

- ・ 隠す



タイムアップを受け取ったとき

- ・ 表示する
- ・ すべてを止める

### 2 ゲームオーバーを選ぼう



穴埋め問題にチャレンジ

ゲームオーバー

ゲームオーバー

右の説明を見ながら、①に何が入るか考えて、プログラムを完成させよう！  
上の「タイムアップ」と似ているので、複製（コピー）して作るとかんたんです。  
（複製のしかたは p.9 にあります）



旗（はじめのボタン）が押されたとき

- ・ 隠す



ゲームオーバーを受け取ったとき

- ・ 表示する
- ・ すべてを止める





## プログラムを確かめよう



をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 残り時間が0になったら、「タイムアップ!!」が表示されましたか？

### ●ためしてみよう！

☐ タイムアップのコスチュームを、変更してみよう！



## 3

## 主人公の作成

### 1

### 主人公を選ぼう

最初に、押したキーに合わせて主人公が左右に向く動きを作ります。



主人公

ゲームオーバー ▼ を受け取ったとき

隠す

ゲームオーバーを受け取ったとき

- ・隠す

左向き矢印 ▼ キーが押されたとき

コスチュームを 左向き ▼ にする

向き ▼ を 1 にする

左向き矢印 キーが押されたとき

- ・コスチュームを「左向き」にする
- ・変数「向き」を1にする

ひだりむ  
左向き



右向き矢印 ▼ キーが押されたとき

コスチュームを 右向き ▼ にする

向き ▼ を 2 にする

右向き矢印 キーが押されたとき

- ・コスチュームを「右向き」にする
- ・変数「向き」を2にする

みぎむ  
右向き



## プログラムを確かめよう



をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 右キーを押すと、主人公は「右向き」になりましたか？

☐ 左キーを押すと、主人公は「左向き」になりましたか？







## ■ プログラムの複製 (違うスプライトへの複製)

複製したいプログラムの上でクリックします。画面右下に表示されているスプライトへ、ドラッグで移動しましょう。移動したいスプライトに重なったら、マウスから指をはなします。



重なると、スプライトがプルプルします



## ■ 「～を送る」の性質 (スクラッチ 3.0 ver.)

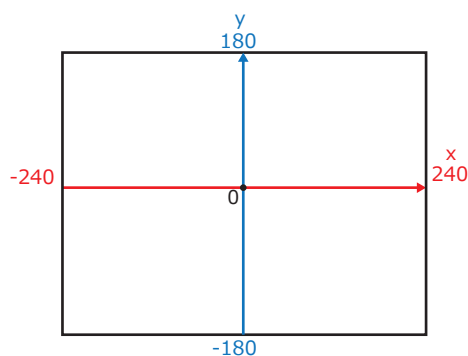


### ブロックの性質

- 「～を送る」は、「～を受け取ったとき」と、セットで使用します。同じスプライトの中のほか、違うスプライトへ命令を送るときにも使えます。

## ■ 「座標」と「正負の数」 (次のページから出てきます)

### 座標



座標とは、位置を数字で管理する手法です。  
x (横方向) と y (縦方向) で表し、  
スプライトの位置を決定します。

画面の中心は、「x (0) ・ y (0)」と決められています。  
スクラッチでは、座標の範囲はこうになります。

**x 座標は「-240 ~ 240」**

**y 座標は「-180 ~ 180」**

### 正負の数

0 より小さい数を「**負の数 (マイナス)**」、0 より大きい数字を「**正の数 (プラス)**」といいます。  
普段の生活で使っている 0 より大きい数はすべて「プラス」の数です。プラスの記号は通常、特に表記されず、省略されています。



マイナスの数「負の数」

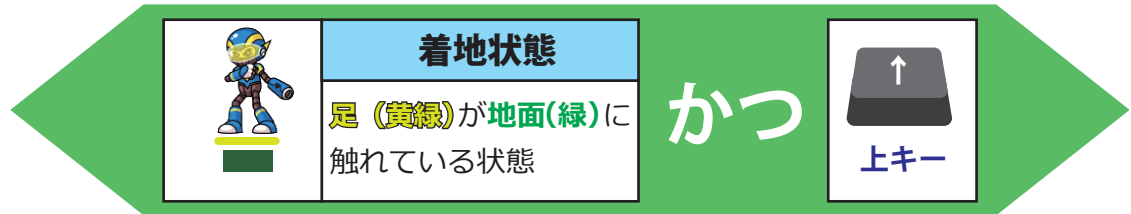
プラスの数「正の数」



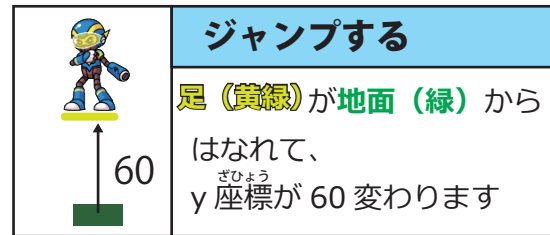
## ■主人公がジャンプするしくみ

もし、主人公が<sup>ちやくち</sup>着地している状態で、<sup>じやうたい</sup>上キーが押されたら、ジャンプします。

もし



主人公のジャンプ動作 60



## ■主人公が地面に落ちるしくみ

主人公が地面に着地をしていなければ (ジャンプ中なら)、地面に向かって落ちます。

でなければ (ジャンプしているなら)

主人公のジャンプ動作 -3



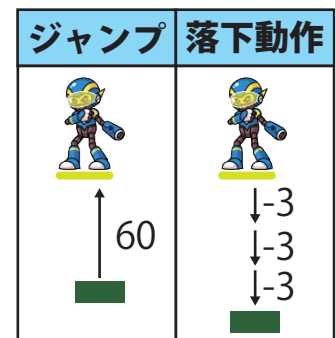
## ■ジャンプする関数を知ろう！

主人公が「ジャンプ」または「落下」するときは  
「ジャンプの高さ (変える y 座標の数)」を使います。

定義 主人公のジャンプ動作 ジャンプの高さ

y 座標を ジャンプの高さ ずつ変える

もし、主人公の y 座標が -175 より下になったら、  
ゲームオーバーを送る





## 2

### 主人公の基本動作を作る



主人公

主人公は **足の色** と **地面の色** を判断して、「ジャンプ」や「落下」をします。

旗が押されたとき

コスチュームを 左向き ▾ にする

表示する

x座標を 0 、y座標を 0 にする

ずっと

もし 色が 色に触れた かつ 上向き矢印 ▾ キーが押された なら

主人公のジャンプ動作 60

でなければ

色が 色に触れた まで繰り返す

主人公のジャンプ動作 -3

もし 主人公の体力 < 1 なら

ゲームオーバー ▾ を送る

旗（はじめのボタン）が押されたとき

- ・コスチュームを「左向き」にする
- ・表示する
- ・最初の位置を決める

ずっと

もし「足が地面に触れていて」「上キー」が押されたなら

- ・「主人公のジャンプ動作」関数の値を 60 にして実行

でなければ

「足が地面に触れる」まで繰り返す

- ・「主人公のジャンプ動作」関数の値を -3 にして実行

もし 主人公の体力が1より小さいなら

- ・ゲームオーバーを送る

ジャンプ

落下

### 色を選ぶ方法

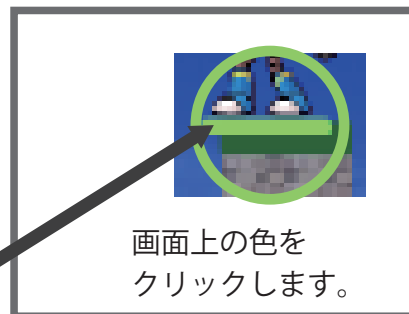
色に触れた

色をクリック



【色番号を指定する場合】  
ここで番号を指定します

【画面上の色を使う場合】



画面上の色をクリックします。



### 3 主人公がジャンプする関数を作る

ジャンプも落下も、「ジャンプの高さ」の数字を使うしくみです。  
ステージから落ちたら、「ゲームオーバー」を表示する命令を送ります。



主人公

定義 主人公のジャンプ動作 ジャンプの高さ

y座標を ジャンプの高さ ずつ変える

もし y座標 < -175 なら

ゲームオーバー を送る

主人公のジャンプ動作が実行されたとき

・y座標を「ジャンプの高さ」ずつ変える

もし y座標が -175 より小さいなら

・ゲームオーバーを送る

### プログラムを確かめよう

をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 主人公が着地しているときに上キーを押したら、主人公がジャンプしましたか？

●ためしてみよう！

☐ 主人公のジャンプ力を 200 にしてみよう！

### 4 陸地の作成

#### 1 陸地の移動スピード・方向の計算

左・右キーを押すと、陸地はそれと逆<sup>ぎゃく</sup>の方向に移動します。

移動するスピードと方向は、「主人公のスピード」× 方向（1 または -1）で決めています。

左キーで「方向」が 1（右）に、右キーで「方向」が -1（左）になります。

もし、主人公のスピードが 3 なら…

x座標を 主人公のスピード × 方向 ずつ変える

「主人公のスピード」×「方向」 ずつ、  
x座標を変えます。



を押したとき

方向

1

正の方向

3

×

1

=

3

（右方向に移動）



を押したとき

方向

-1

負の方向

3

×

-1

=

-3

（左方向に移動）



## 2

## 陸地のプログラムのしくみ

主人公をキーで動かしているのではありません。主人公はその場でジャンプしているだけです。  
陸地を左右に動かしているので、まるで主人公が左右に動いているように見えるのです。

### ■ 陸地が動くしくみ

あいだ  
x座標 -390 ~ 390 の間を、陸地が動きます。

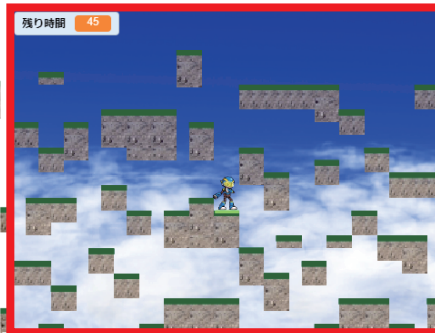
-390

390

#### ぎゃくてん 逆転のアイデア！

動いているのは  
主人公ではなく、陸地です  
プログラミングでは、  
逆に考えるとうまくいく  
ことが、よくあります

見えているゲーム画面



→ を押したとき  
陸地 ⇒ 左に動く  
主人公 ⇒ そのままの位置

陸地が左に動くことにより、  
主人公が右に動くように  
見えます。

↑ 陸地の左側のくぎり -390

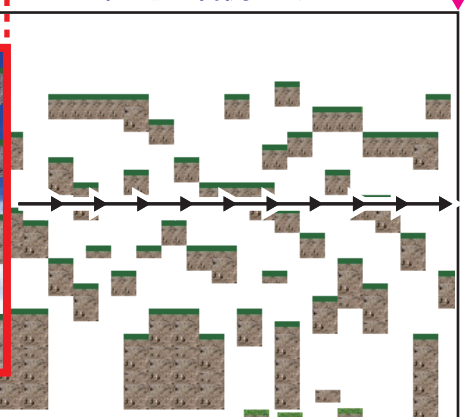


← を押したとき

陸地 ⇒ 右に動く  
主人公 ⇒ そのままの位置

陸地が右に動くことにより、  
主人公が左に動くように  
見えます。

↓ 陸地の右側のくぎり 390

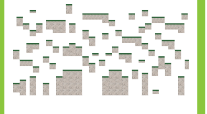




### 3

## 陸地を選ぼう！

陸地を動かすプログラムを作ります。



陸地

ゲームオーバー ▼ を受け取ったとき

隠す

### ゲームオーバー を受け取ったとき

- ・隠す

旗 (はじめのボタン) が押されたとき

0.1 秒待つ

コスチュームを 陸地1 ▼ にする

最背面 ▼ へ移動する

表示する

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

陸地の左右移動 -1

もし 左向き矢印 ▼ キーが押された なら

陸地の左右移動 1

### 旗 (はじめのボタン) が押されたとき

- ・0.1 秒待つ
- ・コスチュームを「陸地 1」にする
- ・画面のいちばん後ろへ移動する
- ・表示する
- ・最初の位置を決める

ずっと

もし「右キー」が押されたなら

- ・「陸地の左右移動」関数の値を -1 にして実行

陸地を左にずらす

もし「左キー」が押されたなら

- ・「陸地の左右移動」関数の値を 1 にして実行

陸地を右にずらす

定義 陸地の左右移動 方向

もし  $-390 < x座標$  かつ  $x座標 < 390$  なら

x座標を 主人公のスピード \* 方向 ずつ変える

### 陸地の左右移動が実行されたとき

もし x 座標が -390 より大きく

x 座標が 390 より小さいなら

- ・x 座標を  
「主人公のスピード」×「方向」ずつ変える





## プログラムを確かめよう

旗をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 左右のキーを押したとき、陸地が動いていましたか？

### ●ためしてみよう！

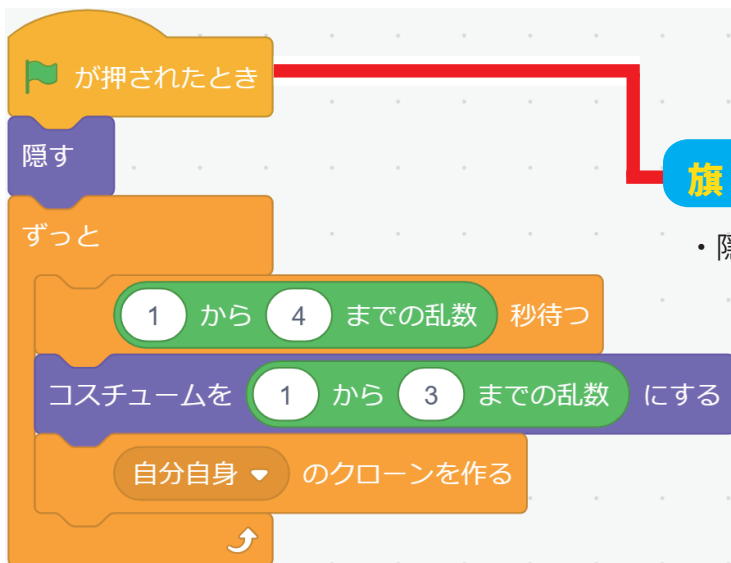
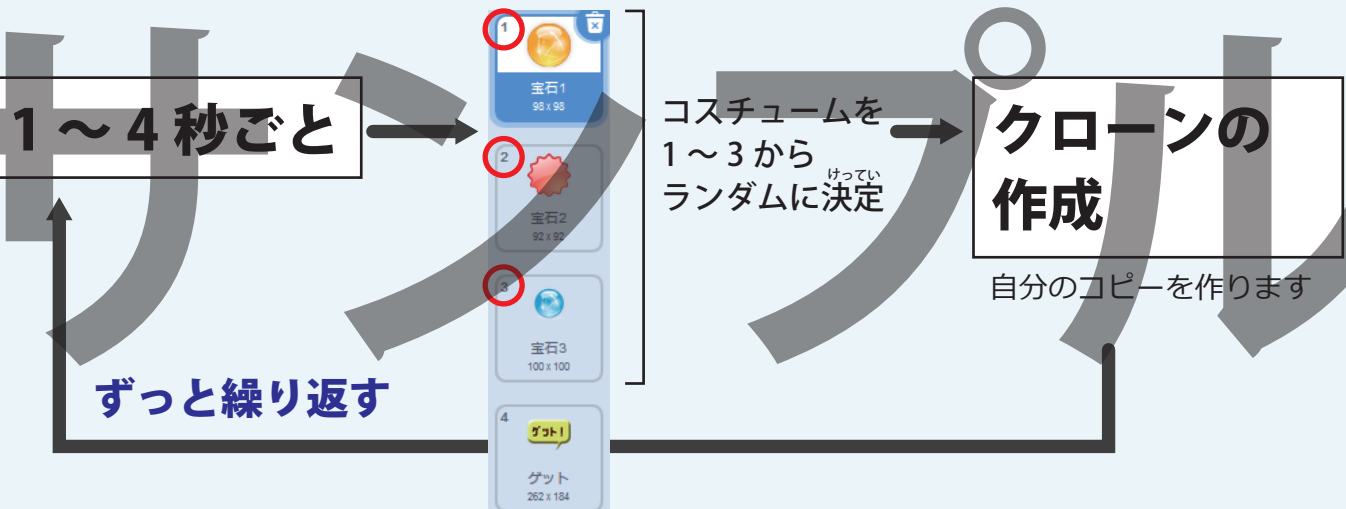
☐ 主人公がまるで倍速で動いているように、陸地の移動速度を2倍にしてみよう！

## 5

## 宝石の作成

### ■「クローン」で宝石をたくさん出す

スタートの合図で、宝石は1～4秒ごとに「クローン」を作ります。  
これはひとつのスプライトからコピーをいくつも作って、画面にたくさん出す方法です。



旗（はじめのボタン）が押されたとき

・隠す

ずっと

- ・1～4秒ごとに
- ・コスチュームを1～3からランダムに決める
- ・クローンを作る





## ■ 宝石（敵・回復アイテム）が移動するしくみ① ⇒ 陸地と一緒に移動させる！

「宝石」「敵」「回復アイテム」は、**左・右キーを押したとき**、**陸地と一緒に動くようにします**。  
もし陸地と一緒に動かなかつたら、左・右キーを押したときに、  
宝石がその場に取り残されたり、主人公につかまらずに逃げていくように見えたりしてしまいます。

x座標を **主人公のスピード** \* **方向** ずつ変える

「主人公のスピード」×「方向」 ずつ、  
x座標を変えます。



を押したとき **方向 1 正の方向** **(右方向に移動)**



を押したとき **方向 -1 負の方向** **(左方向に移動)**

**陸地と同じ動き**  
(方向の決め方は  
p.12 と同じです)

クローンされたとき

x座標を **-240 から 240 までの乱数** y座標を **-180 から 180 までの乱数** にする

表示する

ずっと

1 歩動かす

もし **右向き矢印** キーが押された なら

宝石の左右移動 **-1**

もし **左向き矢印** キーが押された なら

宝石の左右移動 **1**

**色に触れた** まで繰り返す

宝石の落下動作

もし **主人公** に触れた なら

宝石が主人公に触れる動作

もし **端** に触れた なら

隠す

クローンされたとき

- ・位置をランダムに決めて
- ・表示する

ずっと

- ・1歩ずつ右に動く

**宝石を左にずらす**

もし「**右キー**」が押されたなら

- ・「**宝石の左右移動**」関数の値を **-1** にして実行

**宝石を右にずらす**

もし「**左キー**」が押されたなら

- ・「**宝石の左右移動**」関数の値を **1** にして実行

**宝石の落下動作**

**緑（地面）** に触れるまで繰り返す

- ・「**宝石の落下動作**」関数を実行

**主人公に触れたら**

もし **主人公** に触れたなら

- ・「**宝石が主人公に触れる動作**」関数を実行

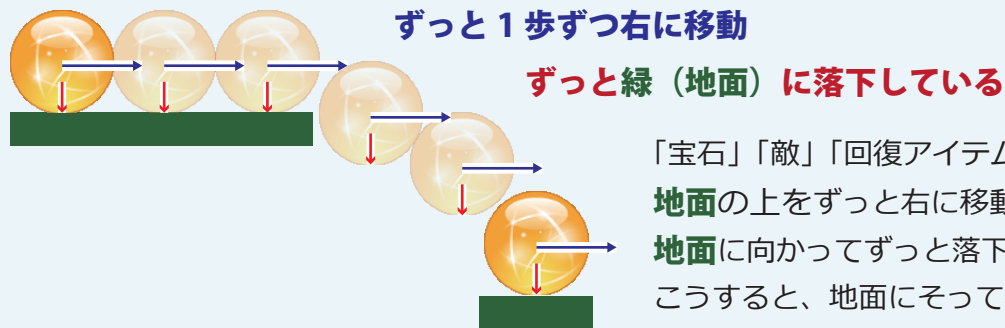
**端に触れたら**

もし **端** に触れたなら

- ・隠す



## ■ 宝石（敵・回復アイテム）が移動するしくみ② ⇒ 地面にそって移動させる！



「宝石」「敵」「回復アイテム」は、**地面**の上をずっと右に移動しながら、**地面**に向かってずっと落下しています。こうすると、地面にそって移動していきます。



定義 宝石の左右移動 方向

宝石の左右移動が実行されたとき

x座標を 主人公のスピード \* 方向 ずつ変える

・ x 座標を「主人公のスピード」×「方向」ずつ変える

定義 宝石の落下動作

宝石の落下動作が実行されたとき

y座標を -3 ずつ変える

・ y 座標を -3 ずつ変える（下へ落ちる）

もし y座標 < -175 なら

もし y 座標が -175 より小さいなら

隠す

・ 隠す

定義 宝石が主人公に触れる動作

宝石が主人公に触れる動作が実行されたとき

宝石の数 を 1 ずつ変える

・ 宝石の数を 1 ずつ増やす

コスチュームを ゲット にする

・ コスチュームを「ゲット」にする

0.5 秒待つ

・ 0.5 秒待つ

隠す

・ 隠す

ゲット

ゲット！

ゲームオーバー を受け取ったとき

ゲームオーバー を受け取ったとき

隠す

・ 隠す



## プログラムを確かめよう

🚩 をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

- ☐ 1～4秒ごとに、3種類の宝石がランダムに出てきましたか？
- ☐ 宝石が主人公に触れたら、「宝石の数」が1つ増えて、その宝石が消えましたか？

### ●ためしてみよう！

- ☐ 宝石が0.3秒ごとに出現するようにプログラムを変更してみよう！



早くできた生徒は取り組もう

### チャレンジ1：改造してみよう！

練習ファイル チャレンジ＞1日目\_チャレンジ練習.sb3

完成ファイル チャレンジ＞1日目\_チャレンジ完成.sb3

#### ★ ① 残り時間が20秒になったら、主人公のスピードが5になるプログラムを作ろう！

ヒント：主人公のプログラムに「もし～なら」のブロックを使って作成しよう！

#### ★ ② 主人公が宝石を取ったとき、体力が「10」回復する命令を追加しよう！

ヒント：宝石のプログラムに「～を～ずつ変える」ブロックを加えて作成しよう！



早くできた生徒は取り組もう

### チャレンジ2：みんなの意見を取り入れよう

プログラミングのお仕事では、お客さんの願いを聞いて取り入れることが多くあります。  
先生やお友だち、おうちの人などに今日作ったゲームを見せて、  
もっとよくするアイデアを聞いてみよう！（おうちで聞いたことも、忘れずにメモしておいてね！）



例) 宝石のほかに、すてきなアイテムを増やしたいな！



1 日目

しゅうりょう  
終了です



2日目

## 6 主人公のこうげき

### 1 主人公<sup>えら</sup>を選ぶ

スペースキーを押したときに、主人公が剣をふるようにしましょう。



主人公

#### 旗（はじめのボタン）が押されたとき

・変数「向き」を1にする

#### ずっと

##### もし「スペースキー」が押されたなら

##### もし「向き」が1（左向き）なら

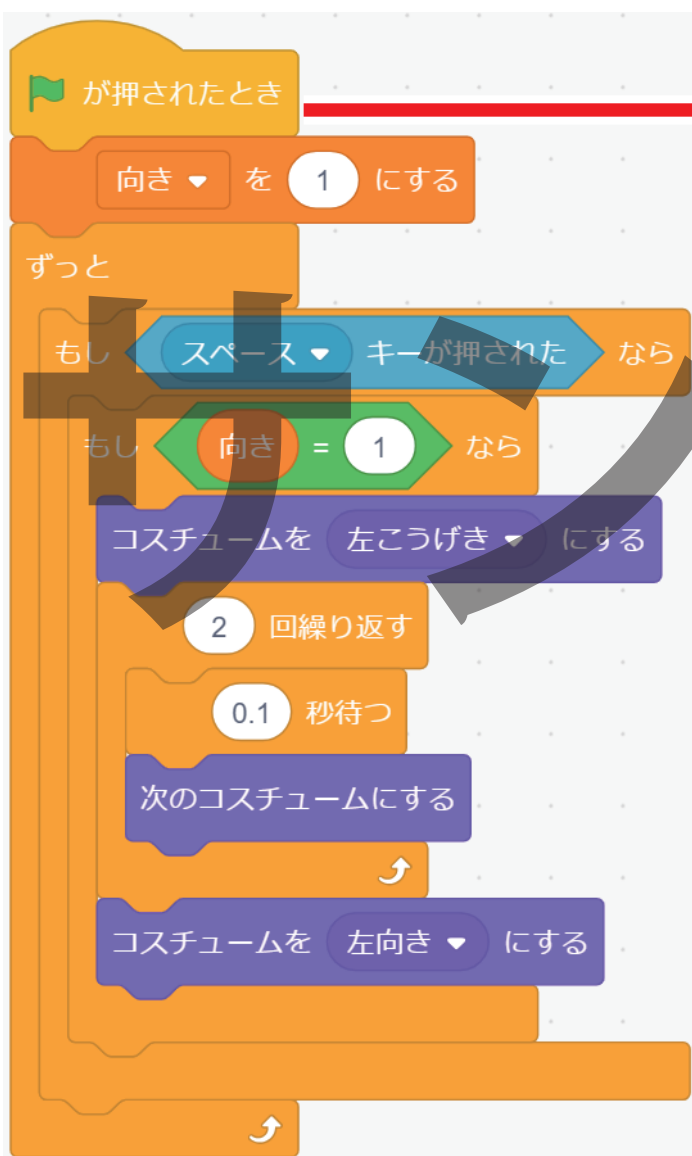
・コスチュームを「左こうげき」にする

##### 2回繰り返す

・0.1秒後に次のコスチュームにする



・コスチュームを「左向き」にもどす



### プログラムを<sup>たし</sup>確かめよう



をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 主人公が左を向いた状態でスペースキーを押すと、左こうげきの動きになりましたか？



主人公

が押されたとき

向き を 1 にする

ずっと

もし スペース キーが押された なら

もし 向き = 1 なら

コスチュームを 左こうげき にする

2 回繰り返す

0.1 秒待つ

次のコスチュームにする

コスチュームを 左向き にする

追加

もし 向き = 2 なら

コスチュームを 右こうげき にする

2 回繰り返す

0.1 秒待つ

次のコスチュームにする

コスチュームを 右向き にする

ずっと

もし「スペースキー」が押されたなら

もし「向き」が2（右向き）なら

・コスチュームを「右こうげき」にする

2 回繰り返す

・0.1 秒後に次のコスチュームにする



・コスチュームを「右向き」にもどす



プログラムを確かめよう



をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 主人公が右を向いた状態でスペースキーを押すと、「右こうげき」の動きになりましたか？



# 7

## 敵の出現

### 1

### 敵を選ぼう

敵のプログラムは、宝石のプログラムを複製して作ると、かんたんです。  
(複製のしかたは p.9 にあります)

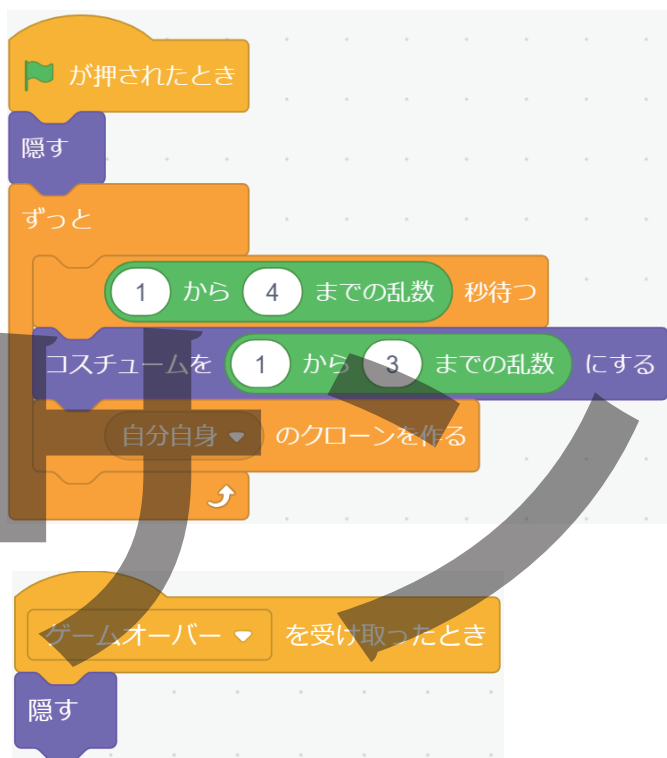


敵



**宝石**のプログラムを複製して作ろう！

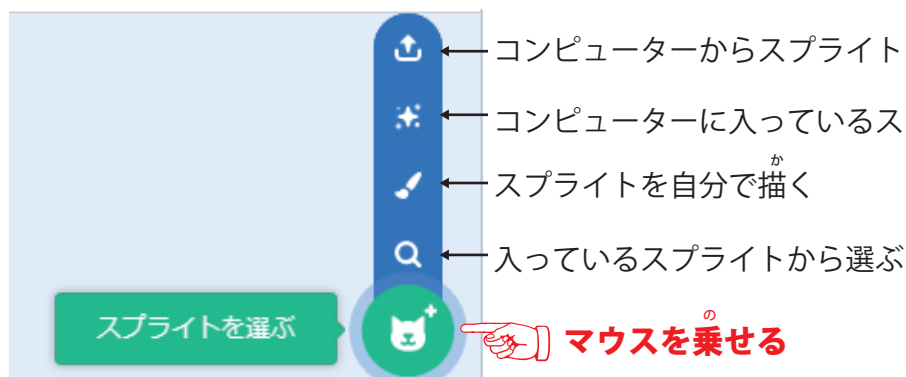
「敵」と「回復アイテム」のしくみは  
「宝石」のプログラムと、ほぼ同じです。



コスチュームを  
1～3から  
ランダムに決定

### 改造するとき便利！ 新しいスプライトを追加する方法

ゲーム完成後、改造してスプライトを増やすときは、スクラッチに入っているものを使うと便利です。



画面の右下の「スプライトを選ぶ」マークから、好きなスプライトを追加できます。

「敵」のほかに、  
「なかま」を作ろうかな～





宝石のプログラムを複製して作ろう！



敵

クローンされたとき

x座標を -240 から 240 までの乱数、y座標を -180 から 180 までの乱数 にする

表示する

ずっと

1 歩動かす

敵を左にずらす

もし 右向き矢印 キーが押された なら

敵の左右移動 -1

敵を右にずらす

もし 左向き矢印 キーが押された なら

敵の左右移動 1

敵の落下動作

色に触れた まで繰り返す

敵の落下動作

主人公に触れたら

もし 主人公 に触れた なら

もし 色に触れた なら

敵げきは

でなければ

敵ヒット

もし剣に触れたなら

・「敵げきは」関数を実行

そうでなければ

・「敵ヒット」関数を実行

端に触れたら

もし 端 に触れた なら

隠す

…宝石プログラムから複製した後に、  
変更または追加するところ

定義 敵の左右移動 方向

x座標を 主人公のスピード \* 方向 ずつ変える

定義 敵の落下動作

y座標を -3 ずつ変える

もし y座標 < -175 なら

隠す

定義 敵げきは

「敵げきは」関数が  
実行されたとき

宝石の数 を 1 ずつ変える

隠す

・宝石を1増やす  
・隠す

定義 敵ヒット

「敵ヒット」関数が  
実行されたとき

主人公の体力 を -10 ずつ変える

コスチュームを ヒット にする

0.5 秒待つ

隠す

・主人公の体力を  
10減らして、  
ヒットの表示にする  
・隠す





## プログラムを確かめよう

🚩 をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

- ☐ 敵が主人公の剣に当たると宝石の数が増えましたか？
- ☐ 敵が主人公に触れたら、主人公の体力が 10 減りましたか？

### ●ためしてみよう！

- ☐ 敵が 0.5 秒ごとに出現するようにしてみよう！
- ☐ 敵にあたると主人公の体力が -50 減るようにしてみよう！

## 8

## 回復アイテムの出現

### 1

### 回復アイテムを選ぼう

回復アイテムも、宝石のプログラムを複製して作りましょう。



回復アイテム

**宝石**のプログラムを複製して作ろう！



### 穴埋め問題にチャレンジ

右の説明を確認しながら、①に何が入るか考えて、プログラムを完成させよう！

```

    旗が押されたとき
    隠す
    ずっと
        ① から 4 までの乱数 秒待つ
        ①
        自分自身 のクローンを作る
    ゲームオーバー を受け取ったとき
    隠す
    
```



コスチュームを  
1～2 から  
ランダムに決定

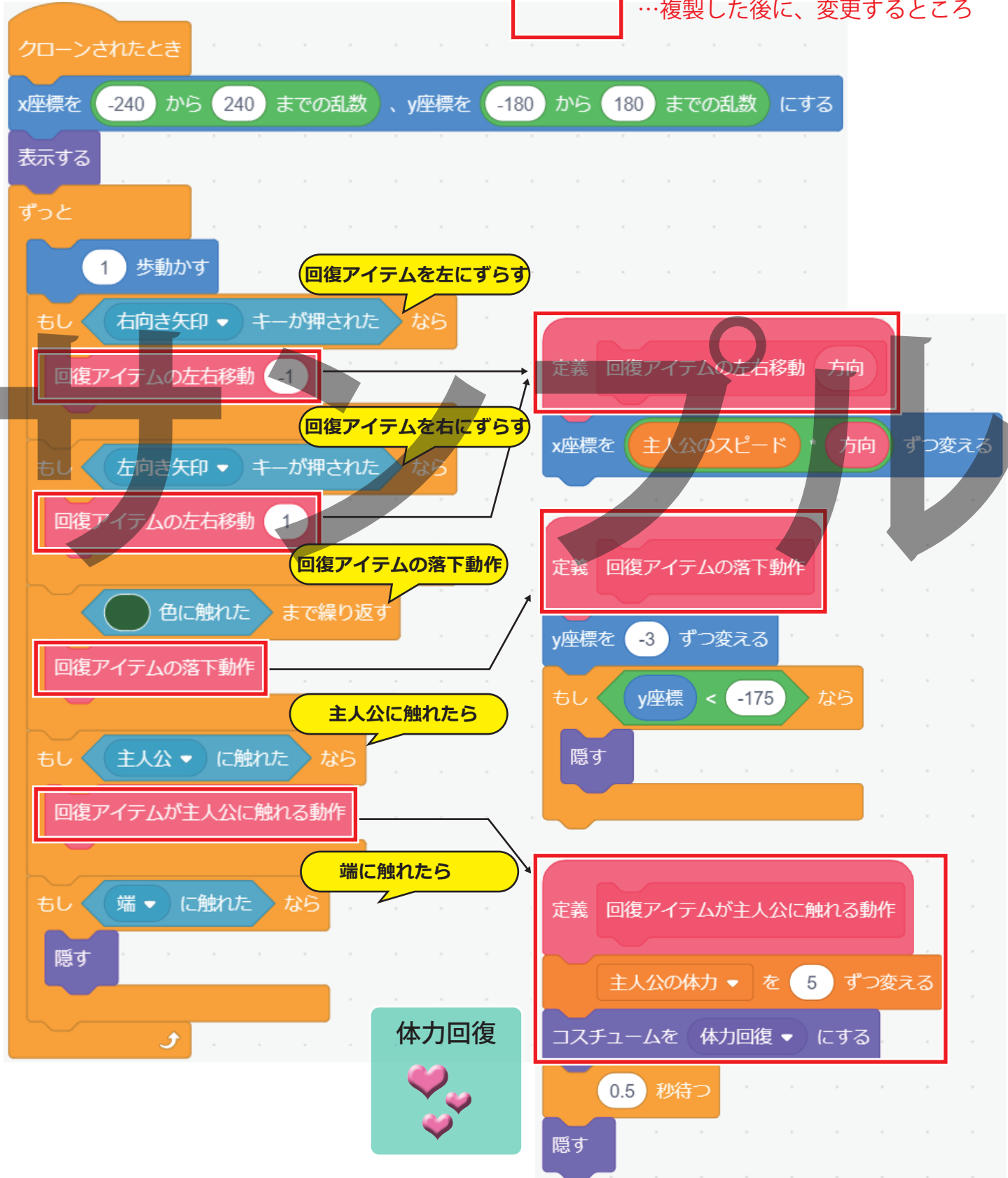


## 宝石のプログラムを複製して作ろう！

回復アイテムのしくみは、p.15～p.17の「宝石」のプログラムと同じです。



…複製した後に、変更するところ





## プログラムを確かめよう

旗をクリックして、ゲームを実行させよう。できていたら✓をつけよう。

☐ 1～4秒ごとに、2種類の回復アイテムがランダムに出てきましたか？

☐ 回復アイテムは端に触れたら、消えましたか？

### ●ためしてみよう！

☐ 回復アイテムが主人公にふれたら、主人公の体力が 50 増えるようにしてみよう！

9

## 拡張要素を追加しよう！

ここでいう「拡張」とは、ゲームを**グレードアップ**することです。  
「宝石の数」が 10 より大きくなったときに、変化を 2 つ加えます。

### 「宝石の数」が 10 より大きくなったら

- ① **主人公のスピード**を速くします。
- ② **「ステージ」と「陸地」のコスチューム**を変えます。



1

### ステージを選ぼう

宝石の数が 10 より大きくなったときに、ステージの背景を変え、主人公のスピードを 3 から 5 に変えます。



ステージ

このプログラムを追加します

旗が押されたとき

ずっと

```

もし 10 < 宝石の数 なら
  背景を 背景2 にする
  主人公のスピード を 5 にする
  スピードアップ を送る
  このスクリプトを止める
  
```

旗（はじめのボタン）が押されたとき

ずっと

もし 宝石の数が 10 より大きいなら

- ・背景を「背景 2」にする
- ・主人公のスピードを 5 にする
- ・**スピードアップ**を送る
- ・このスクリプト(プログラム)を止める

背景 2



「スピードアップ」に送る

スピードアップ



## 2 スピードアップを選ぼう

宝石の数が 10 より大きくなったときに、ステージから命令が送られてきます。  
「スピードアップ」の文字を、画面の右から左へと動かしながら表示します。

スピードアップ

スピードアップ

### ■「スピードアップ」の文字が動くしくみ

ステージからスピードアップの命令を受け取ったら、  
「スピードアップ」の文字が画面の右（x 座標 400）から左（x 座標 -450）へと、3 秒で移動します。



### 穴埋め問題にチャレンジ

上の図を確認しながら、①～⑤に何が入るか考えて、「スピードアップ」のプログラムを完成させよう！

**旗（はじめのボタン）が押されたとき**

- 隠す

**スピードアップ を受け取ったとき**

- 表示する
- x座標を ①、y座標を ② にする
- ③ 秒でx座標を ④ に、y座標を ⑤ に変える
- 隠す
- このスクリプトを止める

**旗（はじめのボタン）が押されたとき**

- ・ 隠す

**スピードアップ を受け取ったとき**

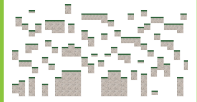
- ・ 表示する
- ・ 最初の位置 (x : 400 / y : 0) を決める
- ・ 3 秒で (x : -450 / y : 0) に移動させる
- ・ 隠す
- ・ このスクリプトを止める



### 3

## 陸地を選ぼう

陸地のコスチュームを変えます。



陸地



### あなた 穴埋め問題にチャレンジ

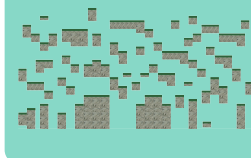
すでにしてあるプログラムに、下の部分を追加して、陸地を変化させよう！

```
when green flag clicked
  wait 0.1 seconds
  set costume to land1
  move to back
  show
  set x coordinate to 0, y coordinate to 0
  loop forever
    if right arrow pressed then
      move land left -1
    if left arrow pressed then
      move land right 1
    if [ ] then
      [1]
      [2]
```

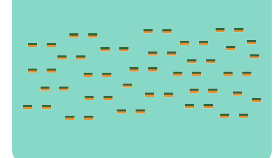
プログラムを追加しよう

もし、  
「宝石の数」が  
10 より大きくなったら  
コスチュームを  
「陸地 2」にします

陸地 1



陸地 2





早くできた生徒は取り組みよう

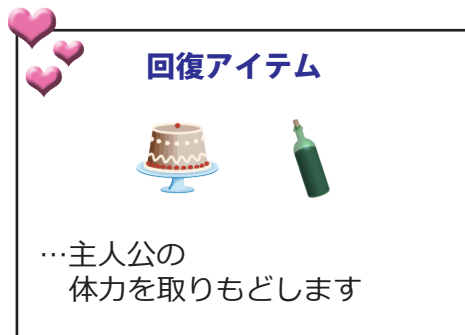
**チャレンジ：もっとグレードアップしよう！**

練習ファイル チャレンジ>2日目\_チャレンジ練習.sb3

完成ファイル チャレンジ>2日目\_チャレンジ完成.sb3

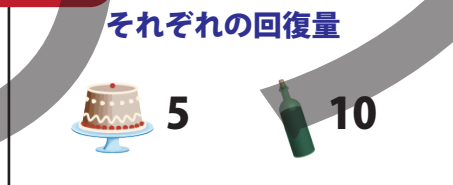
さらにプログラムを追加して、ゲームをおもしろくしましょう。

**クローンで増やしたアイテムの「回復量」「敵の攻撃力」を、  
コスチュームの番号によってそれぞれ違う数に変えます。**



すべて、回復量は5でした

変える



アイテムごとに、  
違う回復量にします



すべて、攻撃力は10でした

変える

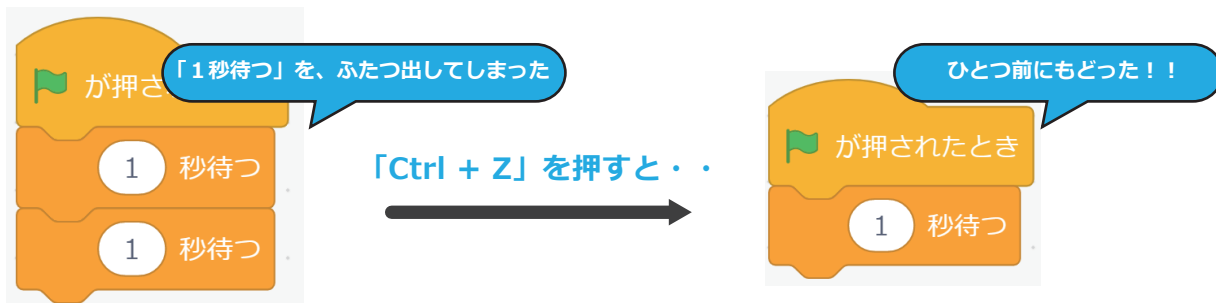


敵ごとに、  
違う攻撃力にします

正解は次のページ➡

### ショートカットキー： やりなおす

もしブロックを組みまちがえたら、**Ctrl + Z**を押すと、ひとつ前にもどることができます。  
そこからもう一度やりなおしをしたい場合は、**Ctrl + Shift + Z**を押しましょう。





## 回復アイテムを選ぼう



回復アイテム



…追加または変更しよう！

Scratch script for the recovery item selection:

- When green flag clicked (when green flag clicked)
- Hide (隠す)
- Wait (ずっと)
- Wait (1 から 4 までの乱数 秒待つ)
- Costume to (1 から 2 までの乱数 にする)
- Conditional block:
  - If costume number (コスチュームの 番号) = 1 (なら)
    - Set recovery amount (回復量) to 5 (を 5 にする)
  - If costume number (コスチュームの 番号) = 2 (なら)
    - Set recovery amount (回復量) to 10 (を 10 にする)
- Create clone of self (自分自身 のクローンを作る)

## 敵を選ぼう



敵



…追加または変更しよう！

Scratch script for the enemy selection:

- When green flag clicked (when green flag clicked)
- Hide (隠す)
- Wait (ずっと)
- Wait (1 から 4 までの乱数 秒待つ)
- Costume to (1 から 3 までの乱数 にする)
- Conditional block:
  - If costume number (コスチュームの 番号) = 1 (なら)
    - Set enemy attack power (敵の攻撃力) to -5 (を -5 にする)
  - If costume number (コスチュームの 番号) = 2 (なら)
    - Set enemy attack power (敵の攻撃力) to -10 (を -10 にする)
  - If costume number (コスチュームの 番号) = 3 (なら)
    - Set enemy attack power (敵の攻撃力) to -15 (を -15 にする)
- Create clone of self (自分自身 のクローンを作る)

定義 回復アイテムが主人公に触れる動作

Scratch script for the recovery item action:

- Set hero's health (主人公の体力) to (回復量) (を 回復量 ずつ変える)
- Costume to (体力回復 にする)
- Wait (0.5 秒待つ)
- Hide (隠す)

定義 敵ヒット

Scratch script for the enemy hit action:

- Set hero's health (主人公の体力) to (敵の攻撃力) (を 敵の攻撃力 ずつ変える)
- Costume to (ヒット にする)
- Wait (0.5 秒待つ)
- Hide (隠す)





早くできた生徒は取り組もう

チャレンジ：みんなのアイデアをもとに改造しよう！

ほかの人の意見や自分の考えをもとに、もっと楽しめるゲームに改造しましょう。  
終わったら、みんなにわかるように発表しましょう。

かいぞうれい

改造例 (追加アイテムの例)



■追加アイテム 時計

⇒残り時間を 10 増やすアイテムです！



■追加アイテム はさみ

⇒残り時間を 10 減らすアイテムです！

重要

できるプログラマーへの道：「わりこみ」してもいいんです！！

## プログラムのとちゅうで動きを変える

「キーが押されたなら（押されたとき）」という命令は、よく使いますね。  
キャラクターが動いているとちゅうで、キーが押されると  
プログラムが「あ、押されたな～」と気づいて、動きを変えます。

このように、プログラムを動かしているとちゅうで、  
何かをきっかけに別のプログラムを動かすことを、「わりこみ」とコンピューター用語でいいます。  
きっかけになった操作は、「イベント」といいます。



## 「イベント」の確認でコンピューターは大忙し

スクラッチでイベントを作るのはかんたんですが、コンピューターはむずかしい仕事をしています。

ふつう、コンピューターはいくつもの命令をいちどに動かすことはできません（※）。  
なのでプログラムを動かしながら、  
ときどき「キーが押されたかな？」と確認しているのです。

アクションゲームなど、タイミングが大事なゲームを作るときは  
イベントを確認する順番も、重要です。  
たとえば 2 人对戦のゲームだったら、確認する順番によって勝ち負けが変わるかもしれません。



## 「わりこみ」に注意しよう

複雑なプログラムを作ようになったら、イベントによる「わりこみ」を  
コンピューターでどう動かすかが、たいせつになります。  
「わりこみ」には注意が必要ながあるんだな、と頭に入れておきましょう。

(※) 近年はマルチコア・マルチスレッドのコンピューターが普通ですが、スクラッチはシングルスレッドで実行されています。

## ベーシックコース

### 勇者の冒険 スクラッチ 3.0 ver.

～クローンを利用して要素を増やそう～



#### バージョンによる表示の違い

スクラッチ 3.0 ver. はバージョンによって、一部の命令の表示やデザインが違うことがあります。  
どちらの表示でも、プログラムには問題ありません。



#### プログラム内「座標」の表記について

スクラッチ 3.0 ver. のファイル読み込み時、「座標」の表記が英語になる場合がありますが、プログラムには問題ありません。



◆文科省 ICT活用教育アドバイザー事務局掲載  
学校ICT化サポート事業者

◆長期コースによる、プログラミングの普及