

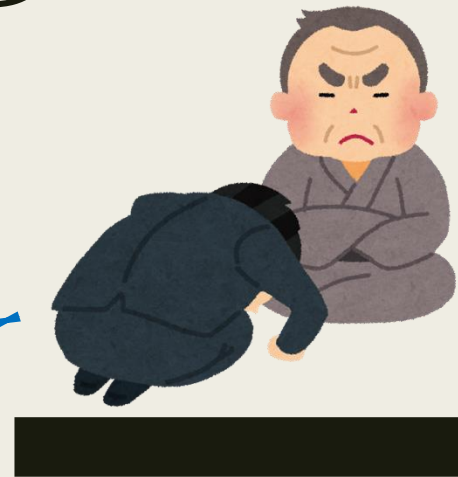
# プログラミング

## 第13回 イベントハンドラ

久保田 匠

### 金曜3限クラスの皆様

先週の第12回の授業資料は昨年度のものでした。  
現在は最新バージョンをアップしているので  
追加された演習問題と合わせて  
確認しておいてください m(\_ \_)m



# [連絡] 第15回はオンデマンド

- 再来週（2月13日）は久保田出張のためオンデマンド講義。
- 来週指示する最終課題に取り組んでください。

## プログラミング

	内容	資料	コード
第1回	いろいろなプログラミング言語 VSCode のインストール	●	<a href="#">Prog_01-1</a>
第2回	Webページを構築する(HTML)	●	<a href="#">Prog_02-1</a>
第3回	Webページの見栄えを整える(CSS)	●	<a href="#">Prog_03-1</a> <a href="#">Prog_03-2</a>
第4回	JavaScriptに触れてみよう	●	<a href="#">Prog_04-1</a>
第5回	変数と演算	●, ★	(なし)
第6回	条件文	●, ★	(なし)
第7回	繰り返し(1)	●, ★	(なし)
第8回	繰り返し(2)	●, ★	(なし)
第9回	繰り返し(3) (オンデマンド)	●	(なし)
第10回	オブジェクト	●, ★	(なし)
第11回	配列	●, ★	<a href="#">Prog_11-1</a>
第12回	ユーザー定義関数	●, ★	<a href="#">Prog_12-1</a>
第13回	イベントハンドラ	●	(なし)
第14回	数式の表示(TeXについて) 最終課題の案内	▲	<a href="#">Prog_14-1</a>
第15回	ウェブツールを開発してみよう (オンデマンド)	▲	<a href="#">課題提出例</a>

この回は授業なし

# [準備]授業資料にアクセス

いつもの作業

- 久保田の授業ホームページに資料がアップロードされている。
- まずは「愛教大 数学」と検索してみよう。



## 授業用ホームページ（久保田）

2025年度前期担当科目

	月曜	火曜	水曜	木曜	金曜
1限					
2限	確率統計II			確率統計II	
3限				線形数学演習I	確率統計II
4限	4年ゼミ				(オフィスアワー)
5限					

2025年度後期担当科目

	月曜	火曜	水曜	木曜	金曜
1限					
2限					
3限	科学リテラシー				プログラミング
4限	(オフィスアワー)	3年ゼミ		4年ゼミ	プログラミング
5限					

その他のコンテンツ → ● ●

数学教育講座 久保田匠（自然科学棟 521 研究室）  
Email: skubota [at] auecc.aichi-edu.ac.jp

## プログラミング

	内容	資料	コード
第1回	いろいろなプログラミング言語 VSCode のインストール	●	<a href="#">Prog_01-1</a>
第2回	Webページを構築する(HTML)	●	<a href="#">Prog_02-1</a>
第3回	Webページの見栄えを整える(CSS)	●	<a href="#">Prog_03-1</a> <a href="#">Prog_03-2</a>
第4回	JavaScriptに触れてみよう	●	<a href="#">Prog_04-1</a>
第5回	変数と演算	●, ★	(なし)
第6回	条件文	●, ★	(なし)
第7回	繰り返し(1)	●, ★	(なし)
第8回	繰り返し(2)		(なし)
第9回	繰り返し(3)		(なし)
第10回	オブジェクト		(なし)
第11回	配列		<a href="#">Prog_11-1</a>
第12回	ユーザー定義関数		<a href="#">Prog_12-1</a>
第13回	イベントハンドラ	●	(なし)
第14回	数式の表示(TeXについて)		<a href="#">Prog_14-1</a>
第15回	ウェブツールを開発してみよう		<a href="#">課題提出例</a>

# [準備]コードの新規作成①

いつもの作業

- 授業用ホームページからサンプルコードをコピーしよう。

## プログラミング

	内容	資料	コード
第1回	いろいろなプログラミング言語 VSCode のインストール	●	<a href="#">Prog_01-1</a>
第2回	Webページを構築する(HTML)	●	<a href="#">Prog_02-1</a>
第3回	Webページの見栄えを整える(CSS)	●	<a href="#">Prog_03-1</a> <a href="#">Prog_03-2</a>
第4回	JavaScriptに触れてみよう	●	<a href="#">Prog_04-1</a>
第5回	変数と演算	●, ★	(なし)
第6回	条件文	●, ★	(なし)
第7回	繰り返し(1)	●, ★	(なし)
第8回	繰り返し(2)		(なし)
第9回	繰り返し(3)		(なし)
第10回	オブジェクト		(なし)
第11回	配列		<a href="#">Prog_11-1</a>
第12回	ユーザー定義関数		<a href="#">Prog_12-1</a>
第13回	イベントハンドラ	●	(なし)
第14回	数式の表示(TeXについて)		<a href="#">Prog_14-1</a>
第15回	ウェブツールを開発してみよう		<a href="#">課題提出例</a>



## Prog\_04-1

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="UTF-8">
  <title>Prog_04-1</title>
  <!-- 今日はここは使いません。 -->
</head>

<body>
  <!-- ここに今日の授業内容を入力します。 -->
</body>

</html>
```

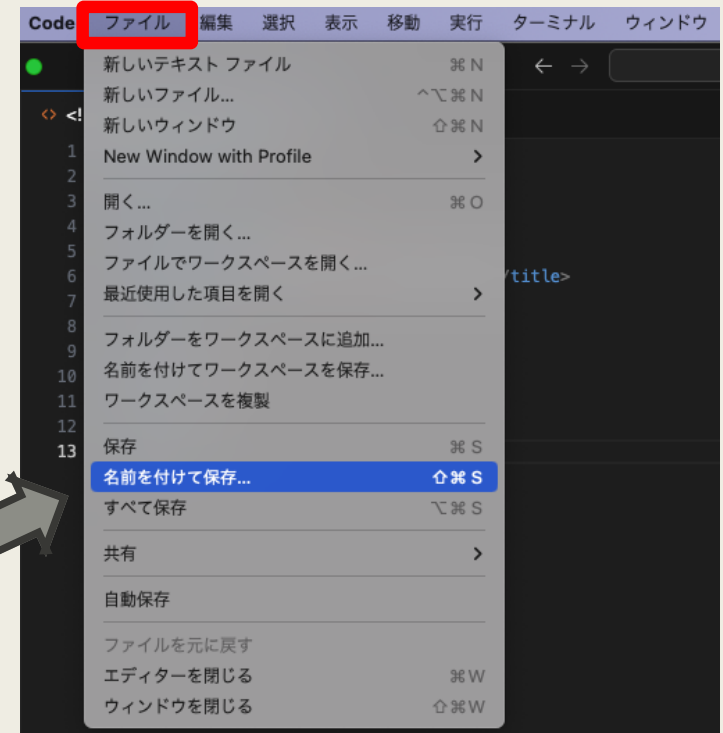
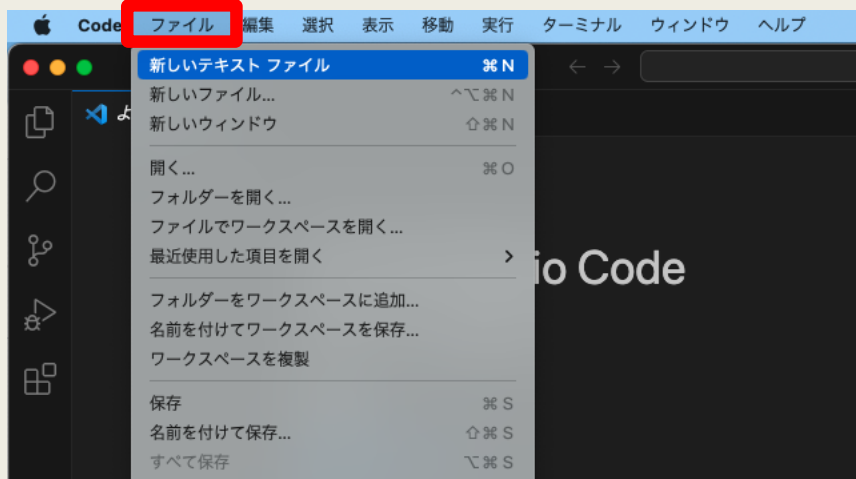
コピー

今日は「Prog\_04-1」を  
選択してください。

# [準備]コードの新規作成②

いつもの作業

- VSCode を起動し「ファイル」から「新しいテキストファイル」を選択。
- そのあと、さきほどコピーした文書をペースト（Ctrl + V）して「名前をつけて保存」。

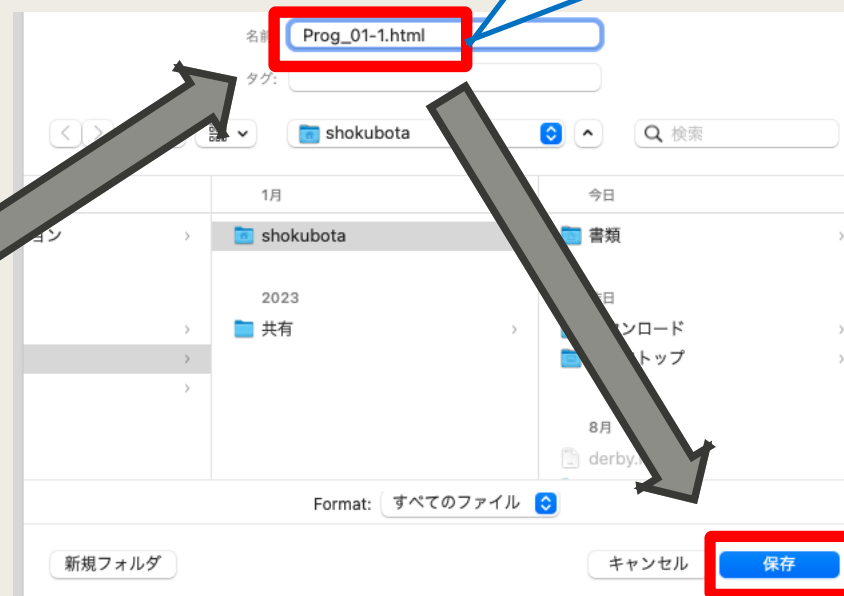
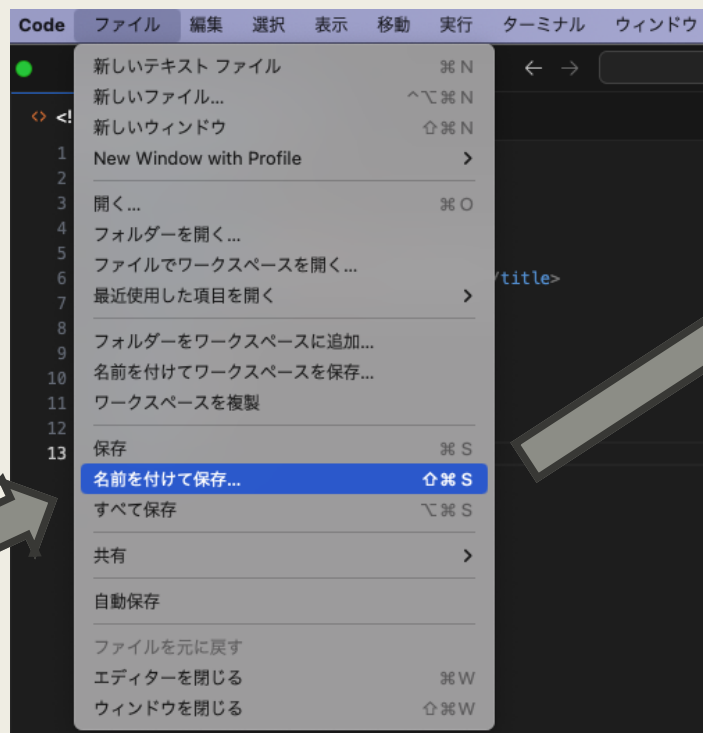


# [準備]コードの新規作成②

いつもの作業

- VSCode を起動し「ファイル」から「新しいテキストファイル」を選択。
- そのあと、さきほどコピーした文書をペースト（Ctrl + V）して「名前をつけて保存」。

今日は  
「Prog\_13-1.html」  
とつける。

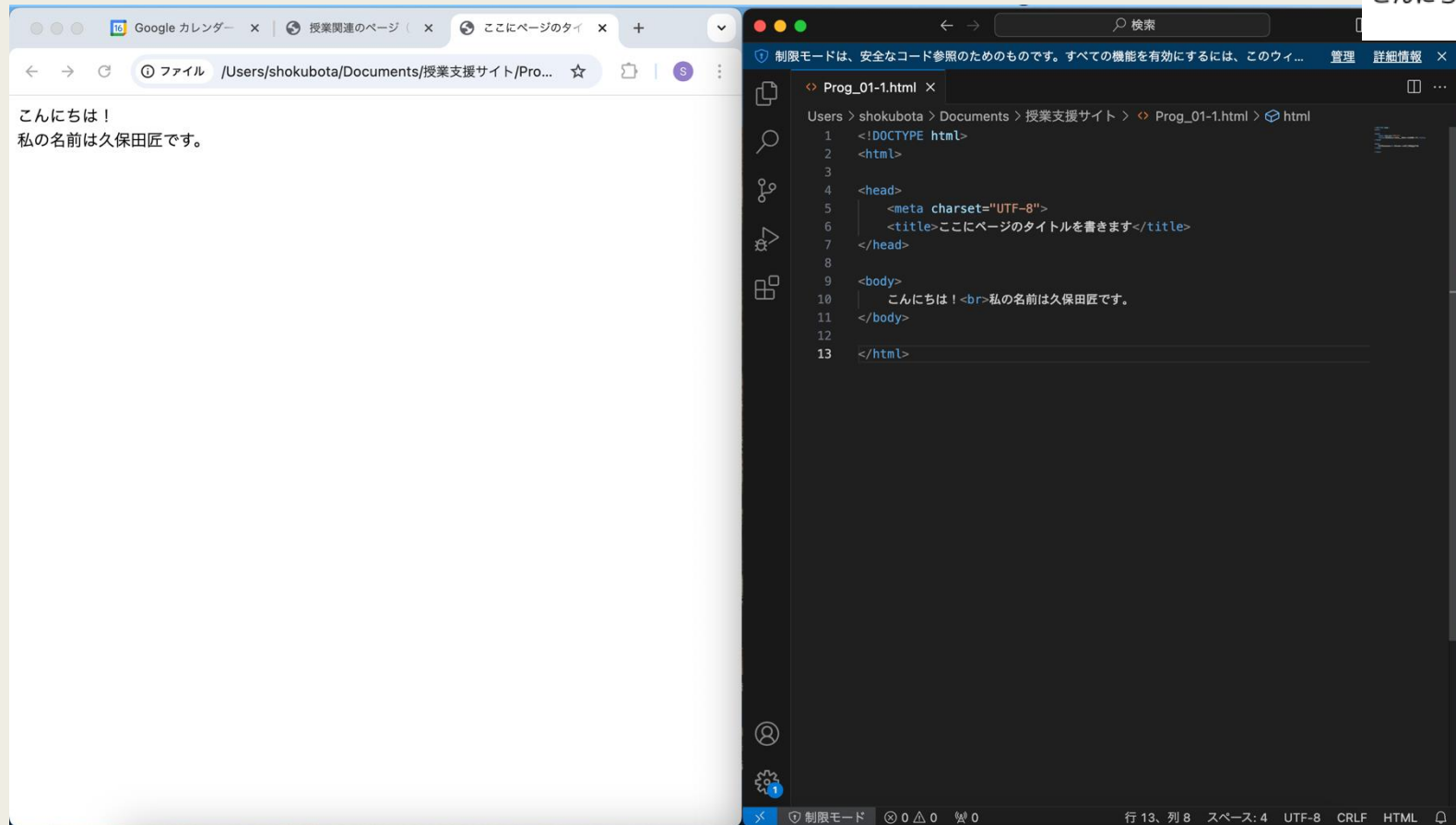


# [準備]作業環境を整える

いつもの作業

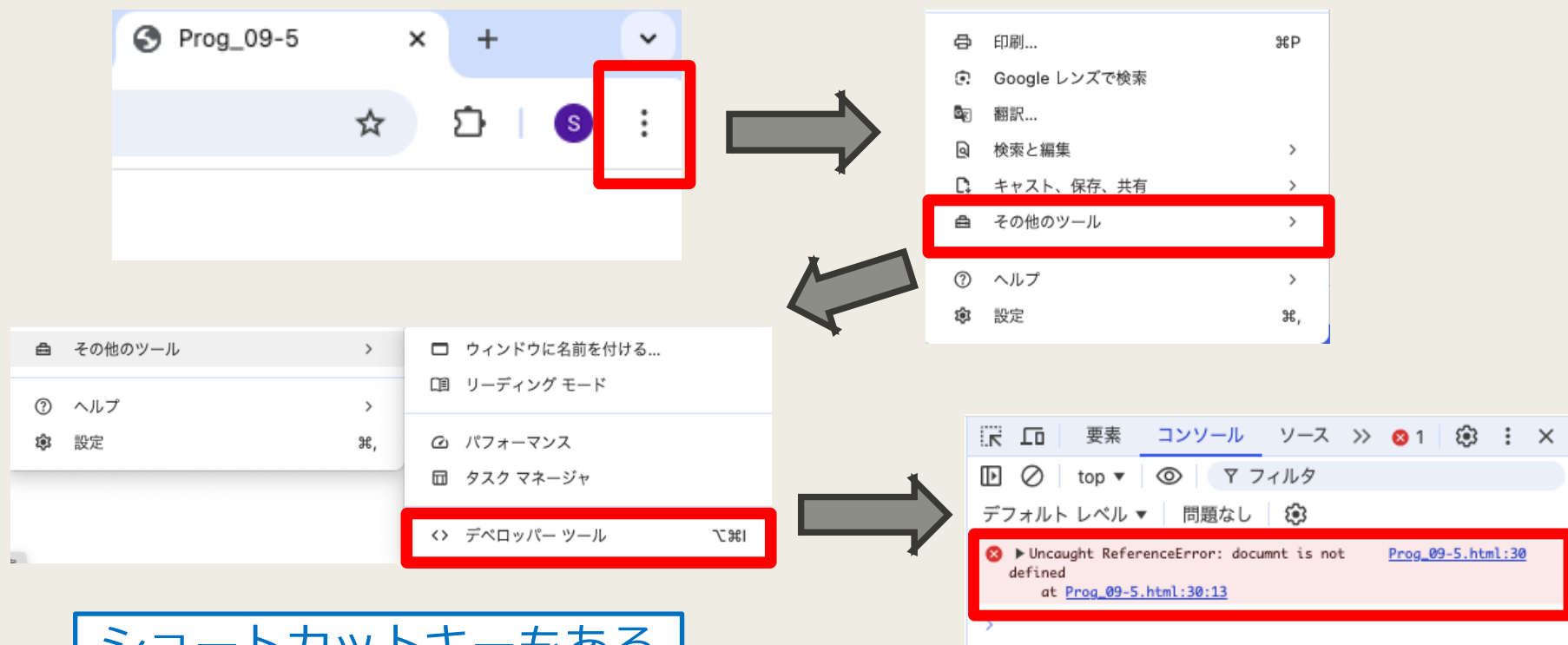
- 保存したhtmlファイルをダブルクリックして開いておく。
- PCの画面をふたつに分け、片方はブラウザ、もう片方はVSCodeを開いておくと便利。

ダブル  
クリック



# [再掲]デベロッパーツール

- 画面に何も表示されないときや、途中までしか表示されないときはプログラムに間違いがある可能性が高い。
- そのときは「デベロッパーツール」を開き、何行目でエラーが発生しているかを見てみよう。



ショートカットキーもある

Windows → Ctrl + Shift + i

Mac → Option + Command + i

30行目でエラーが発生。  
documnt が未定義と言われている  
(スペルミスが発生していた)



# [復習]プログラムは細分化された行動リスト

- コンピュータのプログラムは、いわば「細分化された行動リスト」のようなもので、コードが長くなると読み手は何をやっているか把握しづらい。
- 一方で、多くのプログラムは、いくつかの意味のある「処理のかたまり」に分割できることが多い。

- ・ 玄関の扉の鍵を開ける
- ・ 玄関の扉を開ける
- ・ 外に出る
- ・ 玄関の扉を閉める
- ・ 玄関の扉の鍵を閉める

- ・ 駐車場まで歩く
- ・ 車の鍵を開ける
- ・ 車のドアを開ける
- ・ 車に乗る
- ・ 車のドアを閉める
- ・ シートベルトを閉める
- ・ エンジンをつける

車に乗って外出する

- ・ 変数  $x$  を宣言する
- ・ ユーザーに数値  $x$  を決めてもらう
- ・ 変数  $y$  を宣言する
- ・ ユーザーに数値  $y$  を決めてもらう

- ・ 変数  $x$  の値は0以上?
  - └ Yes なら  $x$  はそのまま
  - └ No なら  $x$  に  $-x$  を代入

- ・ 変数  $y$  の値は0以上?
  - └ Yes なら  $y$  はそのまま
  - └ No なら  $y$  に  $-y$  を代入

- ・ 変数  $z$  を宣言する
- ・  $z$  の値を  $x+y$  とする
- ・  $z$  の値を表示する

$|x| + |y|$  の値を出力

# [復習]プログラムの抽象化と構造化

## ■ 次のうち、意図が分かりやすいのはどちらだろうか？

- 変数  $x$  を宣言する
- ユーザーに数値  $x$  を決めてもらう
- 変数  $y$  を宣言する
- ユーザーに数値  $y$  を決めてもらう
- 変数  $x$  の値は0以上？
  - └ Yes なら  $x$  はそのまま
  - └ No なら  $x$  に  $-x$  を代入
- 変数  $y$  の値は0以上？
  - └ Yes なら  $y$  はそのまま
  - └ No なら  $y$  に  $-y$  を代入
- 変数  $z$  を宣言する
- $z$  の値を  $x+y$  とする
- $z$  の値を表示する

- ◎ ユーザーに  $x, y$  の値を決めてもらう
- ◎  $|x|$  を求める
- ◎  $|y|$  を求める
- ◎  $|x|+|y|$  を表示する

「数値  $x$  の絶対値を求める」  
という処理をひとかたまりにしておくと  
他のプログラムを書くときも  
その処理を使う（再利用）ことができる

- 右のように、おおまかな処理を一度言葉で記述しておくことで、プログラムの構造が瞬時に把握でき、コードの可読性（見やすさ）が向上する。
- さらに、処理の一部をかたまりにしておくことでプログラムの再利用性も高まる。

# [復習]関数

必要なときに呼び出す

- プログラミングでは「処理のかたまり」を **関数** という。
  - 数学の関数（や写像）とはまた意味が違うので注意。
- プログラミングにおける関数は **標準関数** と **ユーザー定義関数** に分けられる。
- 標準関数とは、JavaScript で用意されている関数のこと。
- ユーザー定義関数とは、プログラムの書き手がオリジナルで作った関数（処理のかたまり）のこと。
- ユーザー定義関数は、通常 html ファイル内の head部 で定義する。
- 関数を定義するときは以下の形式に沿って定義する。

```
function 関数名(引数1, 引数2, ...){  
  
    // 実行したい処理を記述  
  
    return 戻り値;  
}
```

# [再掲]第10回以降に学ぶこと

逆行列

ボタンをクリックすると  
答えが表示される

逆行列

[第14回]  
きれいな数式を  
表示する

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は

である。

答え

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は

$\begin{bmatrix} 1 & -2 & 1 \\ 2 & -3 & 1 \\ 6 & -10 & 5 \end{bmatrix}$  である。

答え

[第10回]  
乱数を発生させる

[だいたい済]  
生成した問題に  
対して答えを計算  
(透明色で表示)

[第13回]  
ボタンを押したときに  
特定の処理を行う

[第12回]  
処理のかたまりを  
定義する

- 第11回では配列を扱う。
- 配列はひとつの変数名で複数のデータをまとめて管理できるようにしたもの。
- 例えば、上の例で配列を使わずにプログラムすると、問題の行列と答えの行列の各成分で合計18個の変数を用意しなければならない。

# JavaScriptとは何だったか

- そもそも JavaScript とはWebページに「動き」を与えるものだった。例えば...
  - ボタンをクリックしたときに何かが起こるようにする。
  - フォームに入力されたデータをチェックする。
  - 時間ごとに表示が変わるアニメーションを作る。
- このようなプログラムを書くために重要な概念が「イベント」である。

## 逆行列

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は

である。

答え

## 逆行列

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は  $\begin{bmatrix} 1 & -2 & 1 \\ 2 & -3 & 1 \\ 6 & -10 & 5 \end{bmatrix}$  である。

答え

ボタンをクリックすると答えが表示される



## 逆行列

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は

である。

答え

## 逆行列

行列  $\begin{bmatrix} -5 & 0 & 1 \\ -4 & -1 & 1 \\ -2 & -2 & 1 \end{bmatrix}$  の逆行列は  $\begin{bmatrix} 1 & -2 & 1 \\ 2 & -3 & 1 \\ 6 & -10 & 5 \end{bmatrix}$  である。

答え

ボタンをクリックすると答えが表示される

- イベントとは、ユーザーの操作やブラウザの動作といった「何か起きたこと」を知らせる合図のこと。例えば...
  - ボタンをクリックする
  - マウスポインタが特定の領域に置かれる
  - ページが読み込まれる
- イベントハンドラは、イベントが発生したときに実行される関数。
- 上にある「ボタンをクリックすると答えが表示されるウェブサイト」の仕組みは、
  - ① 問題と答えを生成し、答えは透明色の字で表示しておく。
  - ② 「答え」ボタンをクリックされたら「透明色の字を黒字に変える関数」を発動。

# ボタンをクリックすると文字の色が変わる

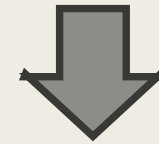
- 次はボタンをクリックすると文字の色が変わるプログラム。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog. 12-1</title>
7   <script>
8     function changeColor(){
9       document.getElementById("001").style.color = "red";
10    }
11  </script>
12 </head>
13
14 <body>
15   <p>
16     右にボタンを用意します。<button type="button" onclick="changeColor()">ボタン</button> <br>
17     このボタンを押すと
18     <span id="001"> この文字の色が </span>
19     変わります。
20   </p>
21 </body>
22
23 </html>
```

右にボタンを用意します。

ボタン

このボタンを押すと この文字の色が 変わります。



右にボタンを用意します。

ボタン

このボタンを押すと **この文字の色が** 変わります。

- <button>タグでボタンが用意される。
- ブラウザ側でボタンが押されると関数 changeColor が呼び出される。
- 関数 changeColor は id が 001 であるタグ内の文字の色を赤色に変更する処理を行う。
- ひとまず上のコードを自分で入力してみよう。

# 1次方程式を出題するプログラム

- ここまでの知識で「数学の問題を自動生成し、ボタンをクリックすると答えが表示されるウェブサイト」が作れる。
- 次は1次方程式を自動生成して「答えを表示する」というボタンをクリックすると答えが表示されるプログラムである。プログラムの仕組みを理解しながら自分で入力してみよう。

```
14 <body>
15 <!--
16 <p>
17     ... 右にボタンを用意します。<button type="button" onclick="changeColor()">ボタン</button> <br>
18     ... このボタンを押すと
19     ... <span id="001"> この文字の色が </span>
20     ... 変わります。
21 </p>
22 </body>
23 <p>
24     ... 1次方程式
25 <script>
26     ... let a = Math.floor(Math.random()*8 + 2);
27     ... let b = Math.floor(Math.random()*10 + 1);
28     ... document.write(a + "x=" + b);
29 </script>
30 </p>
31 <span style="color:transparent" id="001">
32 <script>
33     ... document.write("x=" + b/a);
34 </script>
35 </span>
36 <br>
37     ... である。<button type="button" onclick="changeColor()">答えを表示する</button>
38 </p>
39 </body>
```

id はひとつのタグに  
ひとつだけ対応するので  
先程の内容は <!-- --> で囲む

25~29行目は  
第10回でやった

答えも生成しておくが  
透明色で表示する

1次方程式  $4x=-6$  の解は  
である。



1次方程式  $4x=-6$  の解は  
 $x=-1.5$   
である。



# 分数

1次方程式  $4x = -6$  の解は  
 $x = -1.5$   
である。

- 先のプログラムでは答え（1次方程式の解）が小数で表示された。
- 答えを分数で表示させることはできないだろうか？
- 1次方程式  $ax = b$  の解は  $x = b/a$  であるから、基本的には `document.write(b + "/" + a);` のように入力すればよさそうである。しかし...
  - 分数  $b/a$  は約分できる可能性がある。
  - 分母が 1 の場合は整数の形で表示する必要がある。
- まず、約分を行う処理を考えてみよう。

# ユークリッドの互除法

- 分数を「既約分数になるまで約分する」には、分母と分子を両者の最大公約数で割ればよい。

$$\frac{24}{64} = \frac{3 \times 8}{8 \times 8} = \frac{3}{8}$$

- ふたつの整数の最大公約数を求める方法に「ユークリッドの互除法」がある。
- 整数 $a$ を整数 $b$ で割ったときの商を  $q$ , 余りを  $r$  とすると

$$\gcd(a, b) = \gcd(b, r)$$

が成り立つ。

- 一方、 $\gcd(a, 0) = a$  であるから、余りが  $0$  になるまで同じ操作を繰り返すと最大公約数が求まる。
- これをユーザー定義関数として作成しておこう。

# [演習]最大公約数を求める

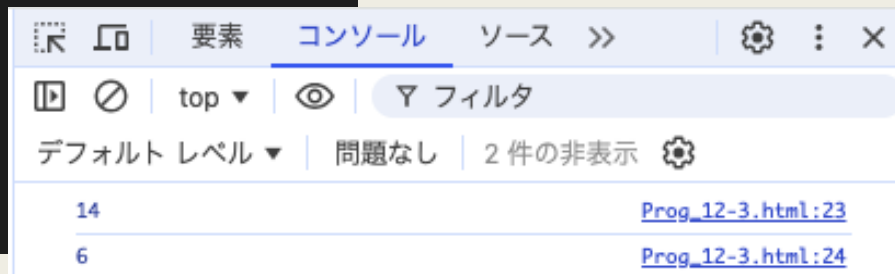
$$\begin{cases} \gcd(a, b) = \gcd(b, r) \\ \gcd(a, 0) = a \end{cases}$$

- 次のプログラムを参考にし、ふたつの整数（どちらも**非負整数**であることを想定してよい）の最大公約数を求めるユーザー定義関数 `findGCD` を作ろう。

```
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_12-3</title>
7   <script>
8     function changeColor(){
9       document.getElementById("001").style.color = "red";
10    }
11
12    function findGCD(a,b){
13      while(true){
14        if(b == 0){
15          return a;
16        }
17        // a と b を更新する
18
19      }
20    }
21
22    console.log(findGCD(56,98));
23    console.log(findGCD(48,18));
24  </script>
25 </head>
```

`b = 0` のときの処理  
関数は `return` に出会うと  
処理をそこで終了し、`a` を返す

`b ≠ 0` のときの処理  
(考えてみよう)



# [演習] 解を既約分数で表示

- さきほど作成した関数 findGCD を使って1次方程式の解を分数の形で表示するプログラムを作成しよう。

```
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_12-3</title>
7   <script>
8     function changeColor(){
9       document.getElementById("001").style.color = "red";
10    }
11
12    function findGCD(a,b){
13      while(true){
14        if(b == 0){
15          return a;
16        }
17        // a と b を更新する
18        let r = a%b;
19        a = b;
20        b = r;
21      }
22    }
23    console.log(findGCD(56,98));
24    console.log(findGCD(48,18));
25  </script>
26 </head>
```

1次方程式  $6x=4$  の解は  
 $x=2/3$   
である。

1次方程式  $3x=6$  の解は  
 $x=2$   
である。

分母が1のときは  
分子の値のみ表示

※32行目、答えを黒字になるように変更

```
28 <body>
29   <p>
30     1次方程式
31     <script>
32       let a = Math.floor(Math.random()*8 + 2);
33       let b = Math.floor(Math.random()*10 + 1);
34       document.write(a + "x=" + b);
35     </script>
36     の解は <br>
37     <span style="color:□black" id="001">
38       <script>
39         let gcd = findGCD(a,b);
40
41         考えてみよう
42
43       </script>
44     </span>
45     <br>
46     である。 <button type="button" onclick="changeColor()">答えを表示する</button>
47   </p>
48 </body>
49 </html>
```

# [次回予告]もっときれいな数式を出力したい

1次方程式  $6x=4$  の解は  
 $x=2/3$   
である。

1次方程式  $9x = 6$  の解は  
 $x = \frac{2}{3}$   
である。

- 先程の演習で、1次方程式の解を分数の形で表示するところまではできるようになった。
- しかし、教科書で見るような、もっと見栄えの良い分数（を含むあらゆる数式）を表示する方法はないだろうか？
- きれいな数式を表示させるためには TeX（てふ）と呼ばれるツールを使う。
- TeX は4年生になって卒論を書くときにも使用する可能性があるなので、この機会に雰囲気を感じておこう。

# [演習]教科書を熟読しよう

- 今日の内容は教科書の p212～p219 がベースになっている
- 残った時間で自分でも該当箇所を熟読してみよう。
- 授業で解説していないコードは自分でも入力してみてどのような出力結果になるか確かめてみよう。
- そろそろ最終課題の構想を練り始めてもよい。参考になりそうな教科書や問題集も準備しておこう。
  - 課題の難易度は題材に大きく依存するので、すでにプログラミングに苦手意識のある学生は、どのような題材を選ぶべきかよく吟味すること。