

# プログラミング

## 第5回 変数と演算

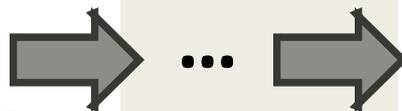
久保田 匠

スライドの右下に  
ページ数をつけました

# [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。



久保田匠の授業関連のページ

2024年度前期担当科目

	月曜	火曜	水曜	木曜	金曜
1限					
2限	<a href="#">確率統計II</a>	<a href="#">確率統計II</a>			
3限				<a href="#">線形数学演習II</a>	<a href="#">確率統計II</a>
4限	(オフィスパワー)				
5限					

2024年度後期担当科目

	月曜	火曜	水曜	木曜	金曜
1限					
2限					
3限	<a href="#">科学リテラシー</a>				<a href="#">プログラミング</a>
4限	(オフィスパワー)	3年ゼミ			<a href="#">プログラミング</a>
5限					

## 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	●	
第6回	剰余演算、条件文(1)	●	
第7回	条件文(2)、繰り返し(1)	●	
第8回	繰り返し(2)	●	
第9回	オブジェクト	●	



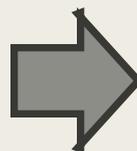
# [準備]コードの新規作成①

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	●	
第6回	剰余演算、条件文(1)	●	
第7回	条件文(2)、繰り返し(1)	●	
第8回	繰り返し(2)	●	
第9回	オブジェクト	●	



## 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>
```

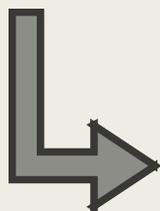
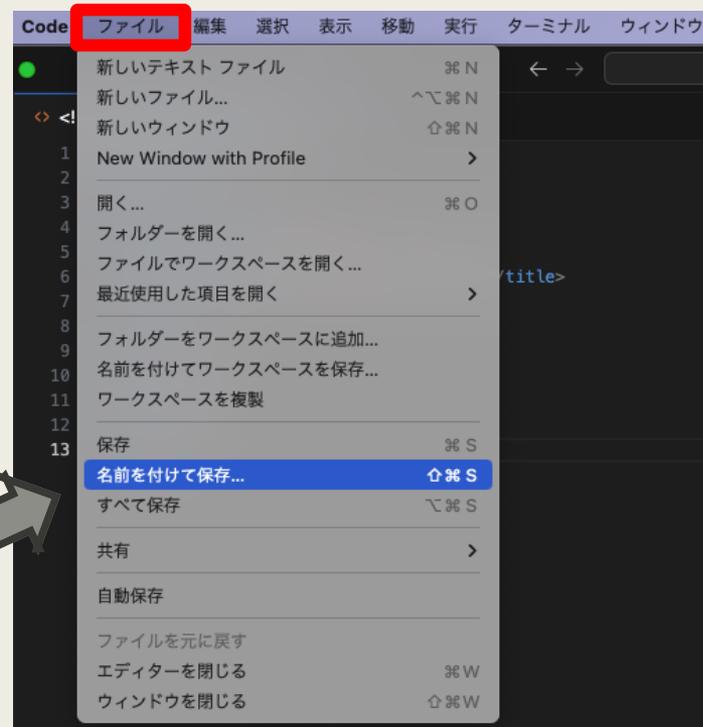
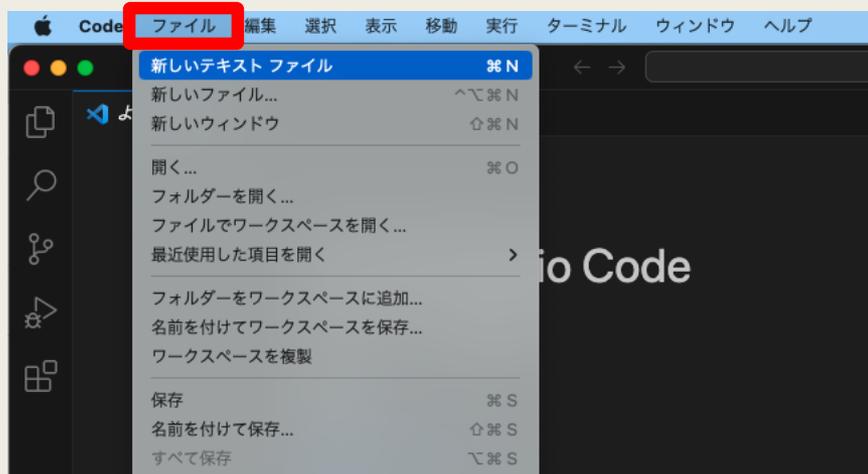
コピー

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」。



Ctrl + V

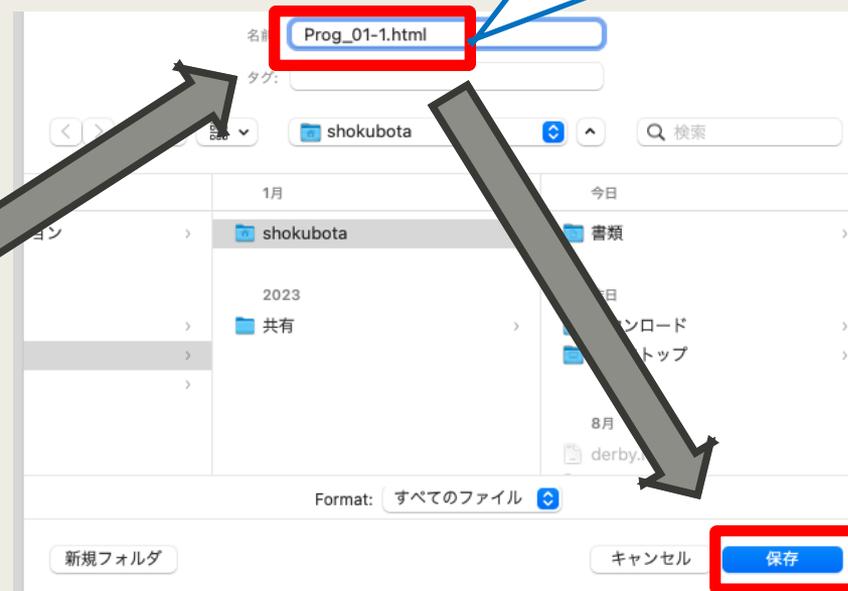
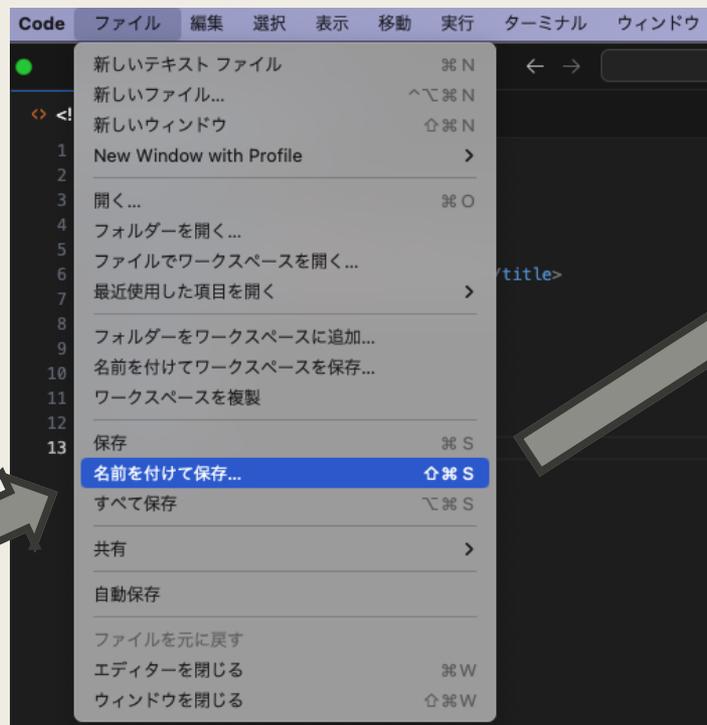
```
ようこそ <<!DOCTYPE html> Untitled-1
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>ここにページのタイトルを書きます</title>
7 </head>
8
9 <body>
10   こんにちは！
11 </body>
12
13 </html>
```

# [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\_05-1.html」  
とつける。



# [準備]htmlファイルを開く

いつもの作業

- 保存したhtmlファイルは...
  - ダブルクリックするとhtmlファイルが開かれwebページが立ち上がる。
  - 右クリックして「Visual Studio Code」で開くを選択すると、コードを編集できる。

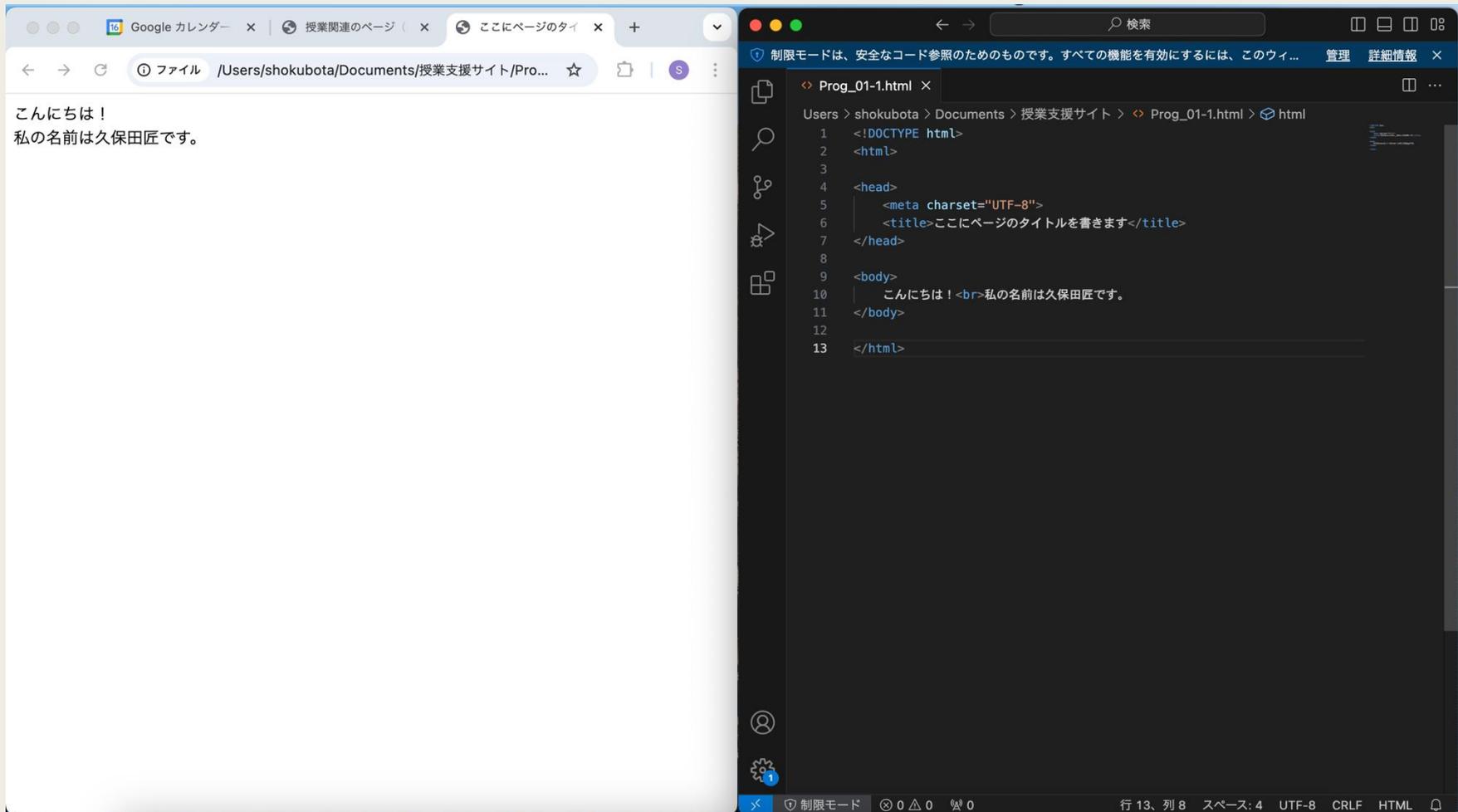


ダブルクリックで「開く」  
右クリックで「編集」

# [準備]作業環境の準備（任意）

いつもの作業

- 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。



# [復習]HTML

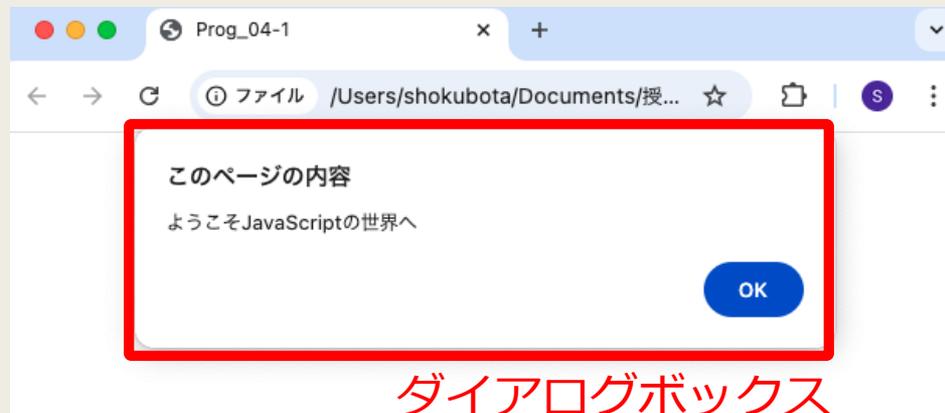
- JavaScript は Web ページで動くプログラミング言語。
- JavaScript を使うためにはまず Web ページを構築する必要がある。
- Web ページを構築するために HTML を用いる。
- HTML は Web ページの土台を作るためのツール。

```
Prog_01-1.html ×
Users > shokubota > Documents > 授業支援サイト > Prog_01-1.html > html
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>ここにページのタイトルを書きます</title>
7 </head>
8
9 <body>
10   こんにちは！<br>
11   私の名前は久保田匠です。
12 </body>
13
14 </html>
```

# [復習]JavaScript に触れてみよう

- JavaScript は Web ページに「動き」を与える。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。
- JavaScript の命令は <script>タグ内に記述する。
- alert は**ダイアログボックス**に何かを表示させる命令。
- 文字列を出力させる場合はダブルクォート(“)と(”)で囲む。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_04-1</title>
7 </head>
8
9 <body>
10  <script>
11    alert("ようこそJavaScriptの世界へ");
12  </script>
13 </body>
14
15 </html>
```



# [復習]足し算について

ステートメント (ひとつの命令)

```
alert(“ようこそJavaScriptの世界へ”);
```

引数 (ひきすう)

セミコロンを  
忘れないで

- 文字列同士の足し算 → 連結

```
alert(“やき” + “にく”); //やきにく
```

↑JavaScript におけるコメントアウト

- 数値同士の足し算 → 計算した結果を出力

```
alert(33+4); //37
```

- 文字列 + 数値 → 数値を文字列に変換して連結

```
alert(“33” + 4); //334
```

# [復習]画面に表示させる命令

- `alert( ***)` → ダイアログボックスに表示

```
alert(“やきにく”)
```

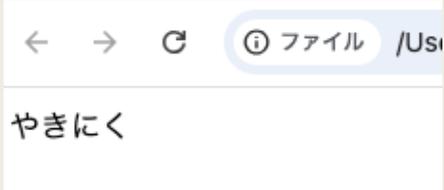


このページの内容  
やきにく

OK

- `document.write( ***)` → ブラウザに表示

```
document.write(“やきにく”)
```

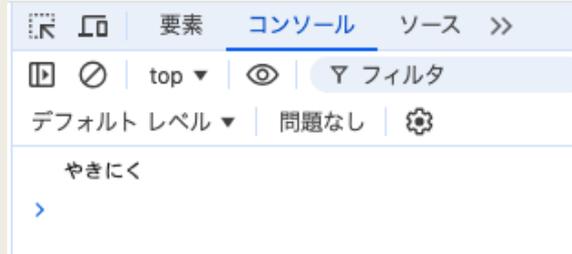


← → ↻ ⓘ ファイル /Usi

やきにく

- `console.log( ***)` → デベロッパーツールのコンソールに表示

```
console.log(“やきにく”)
```



🔍 📄 要素 コンソール ソース >>

📄 🗑️ top 📄 フィルタ

デフォルト レベル ▼ | 問題なし | ⚙️

やきにく

>

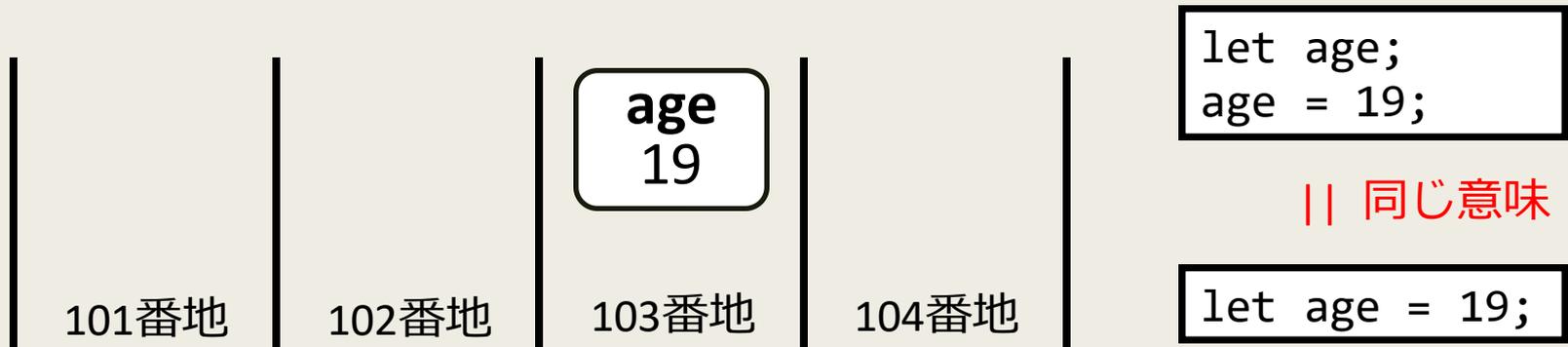
- デベロッパーツールの開き方は第4回の資料を参照のこと。ショートカットキーは

- Windows: `Ctrl + Shift + i`
- Mac: `Option + Command + i`

# 変数

- プログラミング言語において最も重要な要素のひとつに、値を一時的に格納する**変数**がある。
- **変数**は、値を入れる箱のようなもの。
- それぞれの箱には**変数名**と呼ばれる名前がある。
- **アドレス**と呼ばれる、箱を置いてある場所を表す数字の列もある（後の学習のために頭の片隅に入れておくとよい）。

例：年齢を管理する変数「age」を用意して19を代入



実際のアドレスは `0x7ffee10b2a4c` のようにもっと複雑

# 変数を使ってみよう

- 次のコードを入力してみよう。
- age に入る数字は自分の年齢を入れても良い。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="UTF-8">
6    <title>Prog_05-1</title>
7  </head>
8
9  <body>
10   <p>
11     私の年齢は
12     <script>
13       let age = 19;
14       document.write(age);
15     </script>
16     歳です。
17   </p>
18 </body>
19
20 </html>
```

私の年齢は 19 です。

変数 age に入っている  
数字が表示された

- `let age = 19;` ← 変数「age」を宣言し、19を代入する。
- 変数「age」を宣言したいだけのときは `let age;` と入力する。
- 「=」は「左辺に右辺の値を代入する」という意味の演算子。
  - 数学の「=」とは少し意味が違うので注意。

# 年齢を増やす

- 先のファイルの `<script>` タグ内に「`age = age + 1;`」を追加し、出力結果を見てみよう。

```
<script>
  let age = 19;
  age = age + 1;
  document.write(age);
</script>
```

```
<script>
  let age = 19;
  age++;
  document.write(age);
</script>
```

```
<script>
  let age = 19;
  age += 1;
  document.write(age);
</script>
```

- 「`=`」は「左辺に右辺の値を代入する」という意味の演算子。
- `age = age + 1` は「矛盾した数式」ではなく、（19が入っている状態の）変数 `age` に 1 を加えた数（= 20）を改めて変数 `age` に代入する、という意味。
- 変数の数値を 1 増やす方法はこの他にも `age++;` や `age += 1;` と入力する方法もある。
- `age = age + 5;` や `age += 5;` と入力すると変数の数値は 5 増える。

# 変数には文字列も入れられる

- プログラミングで使う変数は、いろいろな種類の値を代入することができる。
  - 値とは、変数に格納される「データそのもの」を指す。
  - [発展] 具体的には数値だけでなく、文字列、真偽値、オブジェクト、配列などをさす。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-1</title>
7 </head>
8
9 <body>
10  <p>
11    私の名前は
12    <script>
13      let name = "野比のび太";
14      document.write(name);
15    </script>
16    です。年齢は
17    <script>
18      let age = 19;
19      age = age + 1;
20      document.write(age);
21    </script>
22    歳です。
23  </p>
24 </body>
25
26 </html>
```

- 左のコードを入力してみよう。変数 name には自分の名前を入力してもよい。

私の名前は 野比のび太 です。年齢は 20 歳です。

## [余談] const

- 先ほどの例では変数を宣言するときに let という命令を使ったが、他にも const がある。
- const は一度宣言したら再代入できない。
- そのため変更しない定数や再代入しない変数に使用する。

```
const TAX_RATE = 0.1;  
TAX_RATE = 0.05;    // エラー：再代入は不可
```

- なお、その他の変数宣言キーワードに var もあるがこれは**現在では推奨されない**。
- var は再宣言も可能であるため誤って同じ名前の変数を再宣言すると元の変数が上書きされてしまい、予期しない動作やバグの原因となる。

```
var age = 19;  
age = 20;    // 再代入は可能  
var age = 45 // 再宣言も可能 ←予期しない動作の原因となる
```

# 変数名をつけるときのルール

- 変数名は基本的には自由につけられるがいくつかのルールがある。
  - ① 英字、数字、アンダースコア(\_)、ドル記号(\$)が使える。
  - ② 1文字目に数字は使えない。
    - num1 は使えるが 1num は使えない。
  - ③ 変数名に予約語は使えない。
    - 予約語とは、プログラミング言語で既に特別な意味を持っているため、変数名に使用できない文字列のこと。
    - if, for, let, const, var, function, return, break, switch など。

# 変数名をつけるときに推奨されるルール

- 必須ではないが読みやすいコード、メンテナンスしやすいコードを書くために望ましいとされるルールがある。
- 意味のある名前をつける。
  - a, b, x, y のような1文字の変数は（数学ではよく使われるが）なるべく避け、その変数の役割や意味を反映した名前にすることが望ましい。age, name, result, sum など。
- **キャメルケース**を使用する。
  - **キャメルケース**とは最初の単語は小文字、次の単語以降の頭文字を大文字にする書き方。複数の単語で構成された変数名で推奨される。 firstName, totalAmount, userAge など。
- 定数はすべて大文字で命名する。
  - 定数値（再代入されることが想定されていない変数）にはすべて大文字の名前をつける。単語間はアンダースコアで区切る。MAX\_VALUE, TAX\_RATE など。

# いろいろな計算を試みよう

- プログラミングにおける計算の基本は四則演算と剰余演算。
- 足し算、引き算の演算子はそれぞれ「+」、「-」。
- 掛け算、割り算の演算子はそれぞれ「\*」、「/」。
- べき乗の演算子は「\*\*」。
- 次のコードを入力してみよう。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-2</title>
7 </head>
8
9 <body>
10  <p>
11    <script>
12      let a = 7;
13      let b = 4;
14      document.write("a = 7, b = 4 とする。<br>");
15      document.write("和は" + (a+b) + "<br>");
16      document.write("差は" + (a-b) + "<br>");
17      document.write("積は" + (a*b) + "<br>");
18      document.write("商は" + (a/b) + "<br>である。");
19    </script>
20  </p>
21 </body>
22
23 </html>
```

a = 7, b = 4 とする。  
和は11  
差は3  
積は28  
商は1.75  
である。

# いろいろな計算をしてみよう

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-2</title>
7 </head>
8
9 <body>
10  <p>
11    <script>
12      let a = 7;
13      let b = 4;
14      document.write("a = 7, b = 4 とする。<br>");
15      document.write("和は" + (a+b) + "<br>");
16      document.write("差は" + (a-b) + "<br>");
17      document.write("積は" + (a*b) + "<br>");
18      document.write("商は" + (a/b) + "<br> である。");
19    </script>
20  </p>
21 </body>
22
23 </html>
```

a = 7, b = 4 とする。  
和は11  
差は3  
積は28  
商は1.75  
である。

- <br> は改行を表すタグだった。
- 15行目の引数「“和は” + (a+b) + “<br>”」で、カッコを省いて「“和は” + a+b + “<br>”」と入力してしまうと「和は74」と表示される。なぜか？

# 0.2 + 0.1 = ??

- 次のように a と b の値を変更して計算結果を見てみよう。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-3</title>
7 </head>
8
9 <body>
10  <p>
11    <script>
12      let a = 0.2;
13      let b = 0.1;
14      document.write("a = 0.2, b = 0.1 とする。<br>");
15      document.write("和は" + (a+b) + "<br>");
16      document.write("差は" + (a-b) + "<br>");
17      document.write("積は" + (a*b) + "<br>");
18      document.write("商は" + (a/b) + "<br> である。");
19    </script>
20  </p>
21 </body>
22
23 </html>
```

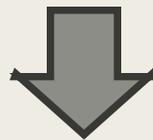
a = 0.2, b = 0.1 とする。  
和は0.30000000000000000004  
差は0.1  
積は0.02000000000000000004  
商は2  
である。

- 差と商の計算結果は正しいが、和と積の計算結果がおかしい。
- なぜこのようなことが起こるのだろうか？

# 丸め誤差

- 大前提として、コンピュータは電気のONとOFF、つまり0と1のみであらゆるデータや命令を処理している。
- 10進数の数値も、コンピュータは2進数に変換して計算する。
- 10進数表記の数 0.1 や 0.2 は2進数では正確には表せず、循環小数となるため有限の桁で打ち切られてしまう。
- そのため、わずかな誤差が発生することがある。
- この誤差を丸め誤差という。

1.10011001100110011001100110011001100110011...



入りきらない部分は捨てる

1.10011001100110011001100110011001100110011...

表現できる数の範囲

# 剰余演算

- 剰余演算(%)は余りを求める演算。
  - $13 \% 5 = 3$       ← 13 を 5 でわった余りは **3**
  - $7 \% 2 = 1$       ← 7 を 2 でわった余りは **1**
  - $27 \% 3 = 0$      ← 27 を 3 でわった余りは **0**
- 繰り返し処理を行うときや、繰り返し回数を何かでわった余りによって処理を変えたいときに用いられる。
  - 偶数番目は○○をして、奇数番目は△△をする。
  - 5で割り切れるときに改行(<br>)をいれる、など。
- 条件分岐（場合分けのようなもの）は次回、繰り返し処理はその次の授業で行う予定。

# 剰余演算

- 先のファイルに次を追記しよう。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-4</title>
7 </head>
8
9 <body>
10 <p>
11   <script>
12     let a = 0.2;
13     let b = 0.1;
14     document.write("a = 0.2, b = 0.1 とする。<br>");
15     document.write("和は" + (a+b) + "<br>");
16     document.write("差は" + (a-b) + "<br>");
17     document.write("積は" + (a*b) + "<br>");
18     document.write("商は" + (a/b) + "<br> である。");
19   </script>
20   <br> <br>
21   13 を 5 でわった余りは
22   <script>
23     document.write(13 % 5);
24   </script>
25   です。<br>
26   -5 を 3 でわった余りは
27   <script>
28     document.write((-5) % 3);
29   </script>
30   です。
31 </p>
32 </body>
33
34 </html>
```

13 を 5 でわった余りは 3 です。  
-5 を 3 でわった余りは -2 です。

- JavaScript では **負の数の剰余は負の数になる仕様** なので注意が必要。

$$-5 = 3 \times (-2) + 1 \text{ より本当の余りは } 1$$

- これを回避するには、例えば  
 $(a \% b) + b$  ( $a < 0$  のとき)  
と入力する方法が考えられる。

# [演習]ふたつの変数の値を入れ替える

- 変数 a と b にそれぞれ 5 と 10 が代入されているとする。
- a と b の値を入れ替える（つまり a には 10、b には 5 が入っているようにする）プログラムを考えたい。
- **コーディング**（コードを書くこと）に着手する前に誤答例を見てみよう。

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Prog_05-5</title>
7 </head>
8
9 <body>
10  <p>
11    <script>
12      let a = 5;
13      let b = 10;
14      document.write("a = " + a + ", b = " + b + " です。<br>");
15      a = b;
16      b = a;
17      document.write("a と b の値を入れ替えると ");
18      document.write("a = " + a + ", b = " + b + " です。");
19    </script>
20  </p>
21 </body>
22
23 </html>
```



a = 5, b = 10 です。  
a と b の値を入れ替えると a = 10, b = 10 です。

a と b の値が入れ替わっていない  
本当は a = 10, b = 5 となってほしい

# [演習]ふたつの変数の値を入れ替える

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="UTF-8">
6    <title>Prog_05-5</title>
7  </head>
8
9  <body>
10   <p>
11     <script>
12       let a = 5;
13       let b = 10;
14       document.write("a = " + a + ", b = " + b + " です。<br>");
15       a = b;
16       b = a;
17       document.write("a と b の値を入れ替えると ");
18       document.write("a = " + a + ", b = " + b + " です。");
19     </script>
20   </p>
21 </body>
22
23 </html>
```

14行目

a  
5

b  
10

15行目

a  
10

b  
10

16行目

a  
10

b  
10

- 15行目で変数 a に変数 b の値を代入した時点で「a=5」という情報が消えてしまう。
- そのため「a=5」という情報を別の変数に一時的に保存しておく必要がある。
- a と b の値を入れ替える（つまり a には 10、b には 5 が入っているようにする）プログラムを書いてみよう。

# [演習]ふたつの変数の値を入れ替える

## 解答例

```
9 <body>
10 <p>
11 <script>
12   let a = 5;
13   let b = 10;
14   document.write("a = " + a + ", b = " + b + " です。<br>");
15   考えてみよう
16   document.write("a と b の値を入れ替えると ");
17   document.write("a = " + a + ", b = " + b + " です。");
18 </script>
19 </p>
20 </body>
```

14行目

a  
5

b  
10

15行目

a  
5

b  
10

tmp

16行目

a  
5

b  
10

tmp  
5

17行目

a  
10

b  
10

tmp  
5

18行目

a  
10

b  
5

tmp  
5

- 15行目で tmp という、a の値を変数を一時的に保存しておくための変数を宣言した (tmp は temporary の略)。
- 16行目で a の値を tmp に入れておくことで「a=5」という情報が失われず、a と b の値を入れ替えることに成功した。

# [演習]教科書を熟読しよう

- 今日の内容は教科書の p60～p81 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。