プログラミング

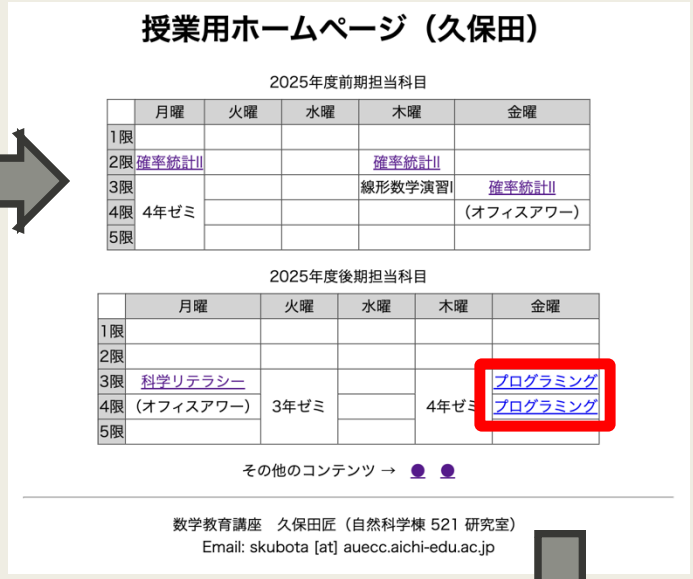
第7回
繰り返し(1)

久保田 匠

[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。



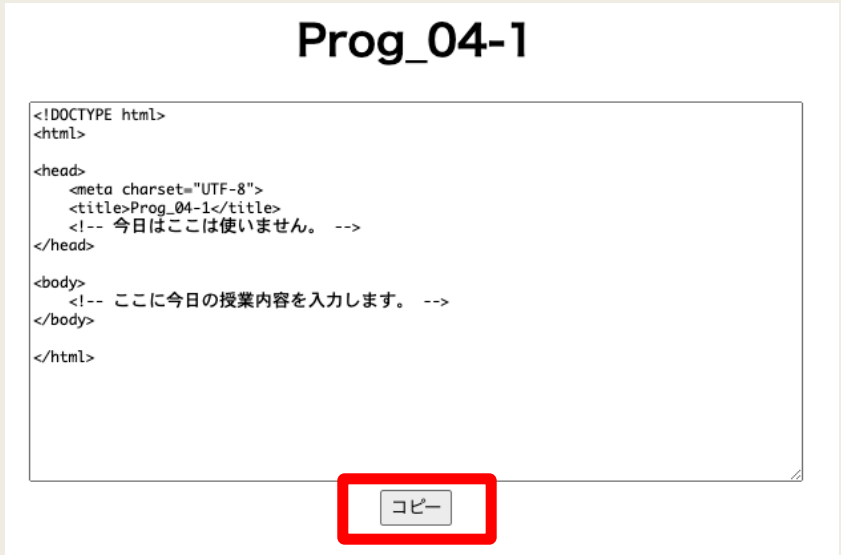
プログラミング

	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	●	Prog_01-1
第2回	Webページを構築する(HTML)	●	Prog_02-1
第3回	Webページの見栄えを整える(CSS)		Prog_03-1 Prog_03-2
第4回	JavaScriptに触れてみよう		Prog_04-1
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)		Prog_08-1
第9回	繰り返し(3)		（なし）

[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。

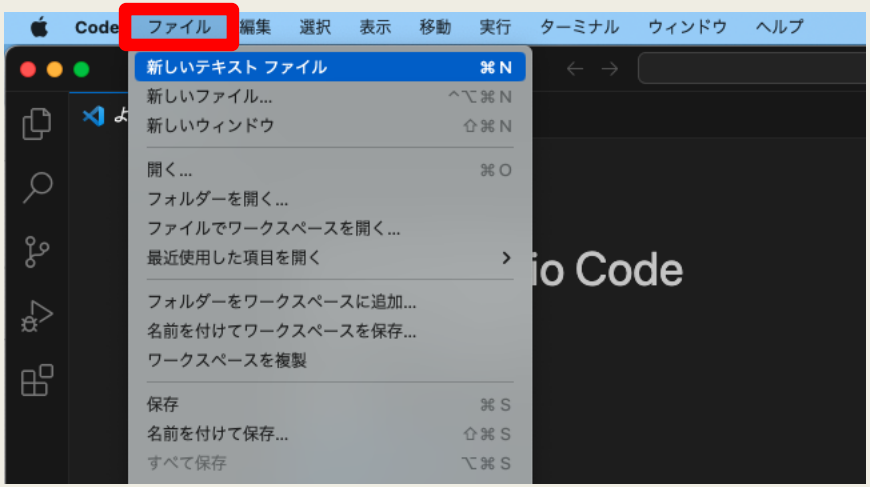


今日も「Prog_04-1」を選択してください。

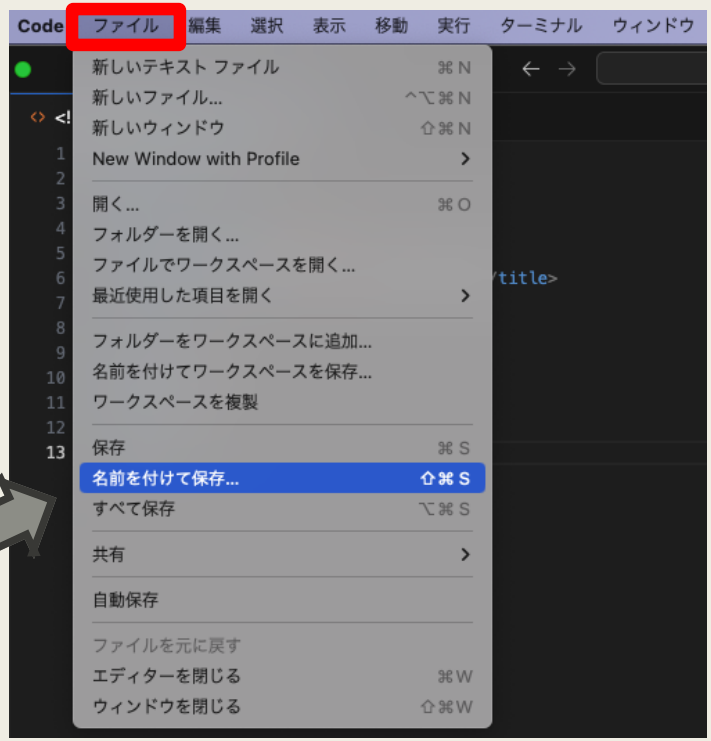
[準備]コードの新規作成②

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」。



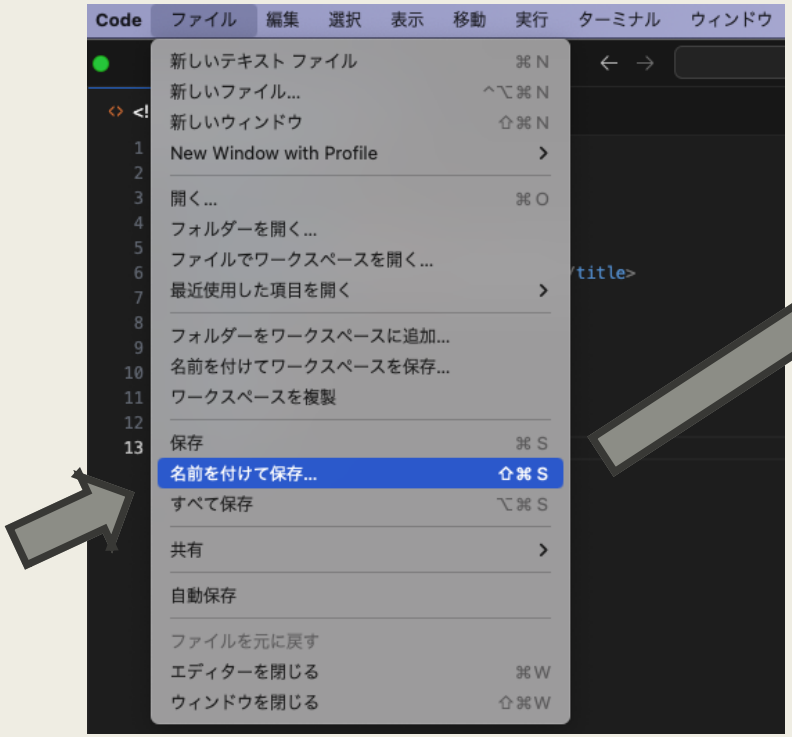
Ctrl + V



[準備]コードの新規作成②

いつもの作業

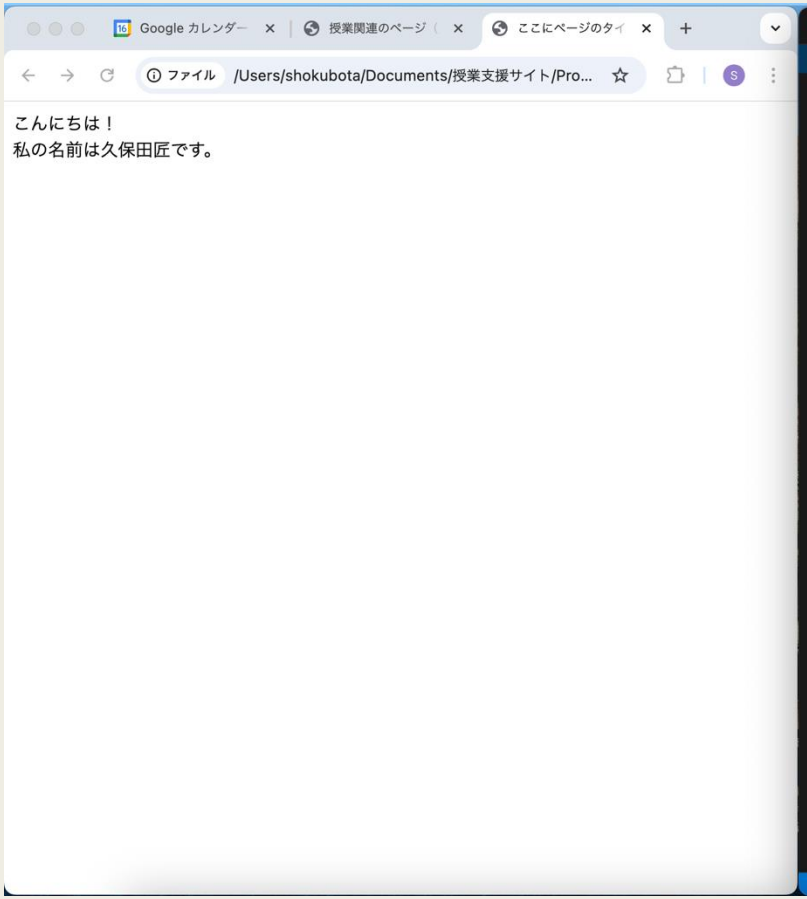
- VSCode を起動し「ファイル」から「新しいテキストファイル」を選択。
- そのあと、さきほどコピーした文書をペースト（Ctrl + V）して「名前をつけて保存」。

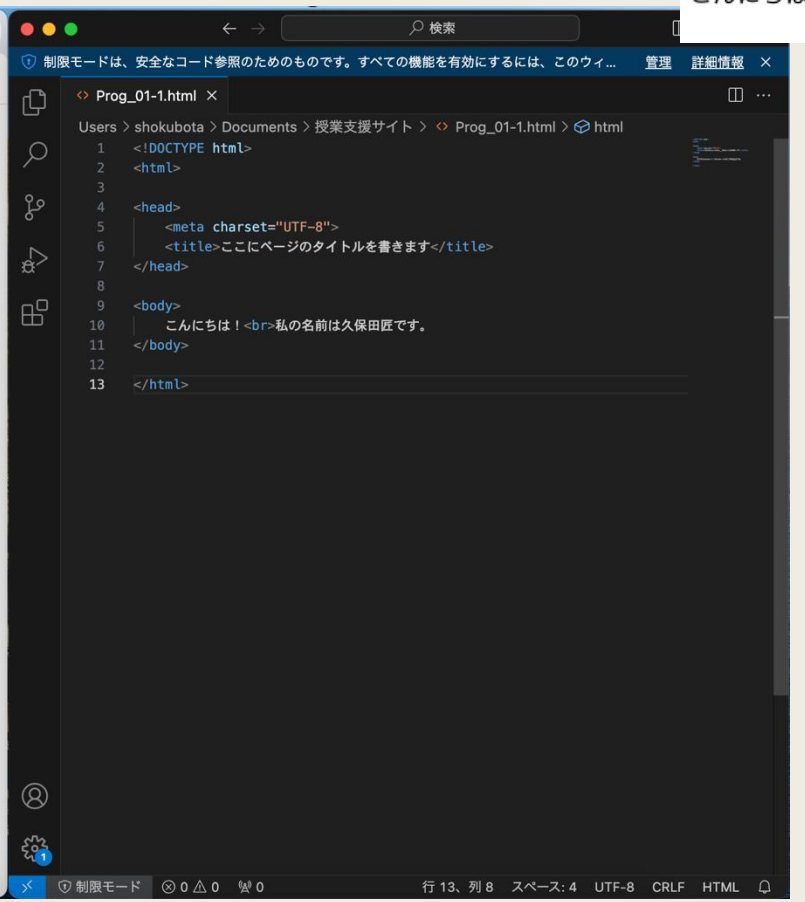


[準備]作業環境を整える

いつもの作業

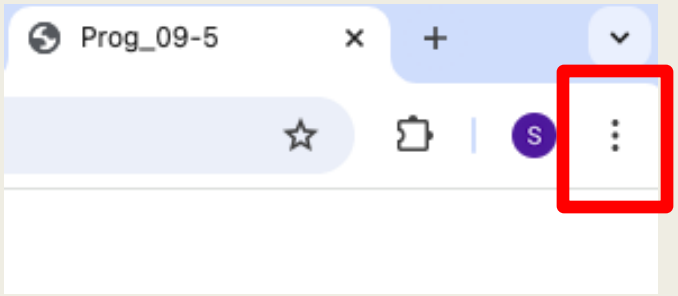
- 保存したhtmlファイルをダブルリックして開いておく。
- PCの画面をふたつに分け、片方はブラウザ、もう片方は VSCode を開いておくと便利。

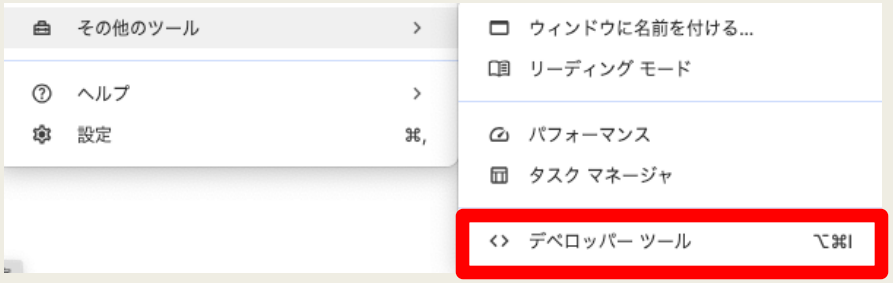


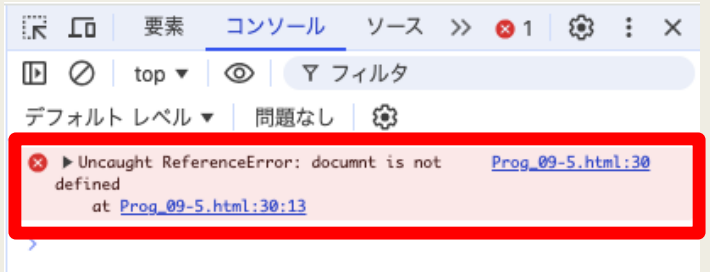


[再掲]デベロッパーツール

- 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]ユーザーから入力を受け取る

- これまではコード内で変数を宣言し、値を代入していた。
- しかし、これでは後から値を変更したいときにそのたびにコードを編集しなければならない。
- 頻繁に値が変わる場合、ブラウザを通してユーザーに入力してもらえれば便利である。
- 前回、prompt という命令を使い、ユーザーに変数の値を決めてもらうプログラムを書いた。

```
1   <!DOCTYPE html>
2   <html>
3
4   <head>
5       <meta charset="UTF-8">
6       <title>Prog_06-2</title>
7   </head>
8
9   <body>
10      <p>
11          <script>
12              let year = prompt("今は西暦何年？");
13              document.write("今は令和" + (year - 2018) + "年です。");
14          </script>
15      </p>
16  </body>
```

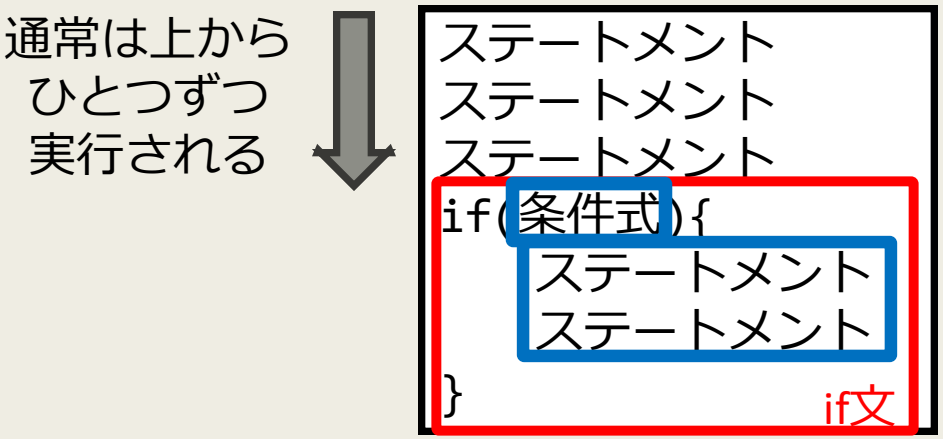
入力ダイアログボックス

今は令和6年です。

```
// 13行目は次のように入力してもよい。
document.write(“今は令和”, year - 2018, “年です。”);
```

[復習]もし〇〇ならば△△を行う

- if文は、プログラムが特定の条件を満たすかどうかを確認し、その条件が真(true)であれば処理を実行するという命令。
- 偽(false)であればその処理をスキップする。



①条件式が成立したら
②{ }内のステートメントが実行される
③条件式が成立しなかったら{ }内は実行されない

- 前回、右のコードを入力した。
- 14行目の「>=」は「≧」と同じ意味。

```
1   <!DOCTYPE html>
2   <html>
3
4   <head>
5       <meta charset="UTF-8">
6       <title>Prog_06-3</title>
7   </head>
8
9   <body>
10      <p>
11          <script>
12              let point = prompt("試験の点数を0から100以下の整数値で入力してください。");
13              document.write("あなたの成績は" + point + "点でした。");
14              if(point >= 60){
15                  document.write("単位取得おめでとう！");
16              }
17          </script>
18      </p>
19  </body>
20
21  </html>
```

[復習]比較演算子

- if文内の条件式には「true」か「false」かを返すステートメントをいれる。
- そのために用いられる記号（演算子）を 比較演算子 という。

```
ステートメント
ステートメント
ステートメント
if(条件式){
    ステートメント
    ステートメント
}
```

比較演算子	使用例	説明
==	a == b	a と bが等しければ true
!=	a != b	a と bが等しくなければ true
<	a < b	a が bより小さければ true
<=	a <= b	a が b 以下ならば true
>	a > b	a が bより大きければ true
>=	a >= b	a が b 以上ならば true

「=」ではなく「==」なので注意

[復習]そうでないならば□□を行う

- if文のあとに　else　をつけると条件式が成立しなかった場合の処理を記述することができる。
- 前回、試験の点数が60点以上とならなかった場合に「不合格です。また来年頑張ってください。」と表示するプログラムを作った。

```
if(条件式){
    ステートメント
    ステートメント
} else {
    ステートメント
    ステートメント
}
```

①条件式が成立したら
②{ }内のステートメントが実行される
③条件式が成立しなかったら
④else { }内のステートメントが実行される

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8">
    <title>Prog_06-3</title>
</head>

<body>
    <p>
        <script>
            let point = prompt("試験の点数を0から100以下の整数値で入力してください。");
            document.write("あなたの成績は" + point + "点でした。");
            if(point >= 60){
                document.write("単位取得おめでとう！");
            } else {
                document.write("不合格です。また来年頑張ってください。");
            }
        </script>
    </p>
</body>

</html>
```

[復習]条件を細かく設定する

- else の部分にさらに if を入れることもでき、より細かい条件判断を行うことができる。

上から順に
条件式が
成立するかを
調べていく

```
if(条件式１){
    //条件式１が成立した場合の処理
} else if(条件式２){
    //条件式１が成立せず条件式２が成立した場合の処理
} else if(条件式３){
    //条件式２が成立せず条件式３が成立した場合の処理
…
} else {
    //すべての条件式が成立しなかった場合の処理
}
```

指定した回数だけ処理を繰り返す

- if文のような条件判断に並び、プログラムに欠かせない概念にループ（繰り返し）がある。
- ループを記述する方法は主に for文 と while文 の2種類がある。
- for文は繰り返し回数が分かっている場合に使い、while文は繰り返し回数が定まっていない場合（特定の条件を満たすまで繰り返すなど）に用いる。
- まずは for文を扱う。
- for文の基本的な書式は以下の通り。慣れるまでは難しく感じられるかもしれない。頑張ろう。

基本的な書式

```
for(初期化式; 条件式; 制御変数の更新){
    ステートメント
    ステートメント
    ステートメント
}
```

具体例（“やきにく”と10回言うプログラム）

```
for(let i = 1; i <= 10; i++){
    document.write(“やきにく”);
}
```

i++ は 変数 i の値を 1 増やすという意味。
i = i+1 や i += 1 と同じ。

「やきにく」と10回言うプログラム

- 次は「やきにく」と10回言うプログラムである。

```
1   <!DOCTYPE html>
2   <html>
3
4   <head>
5       <meta charset="UTF-8">
6       <title>Prog_07-3</title>
7   </head>
8
9   <body>
10      <p>
11          <script>
12              for(let i=1; i<=10; i++){
13                  document.write("やきにく");
14              }
15          </script>
16      </p>
17  </body>
18
19  </html>
```

- i は 1 から 10 まで動き、処理ごとに i は1ずつ増える。
- i = 1 のときの処理
  - 「やきにく」と表示する。
- i = 2 のときの処理
  - 「やきにく」と表示する。
- 同じ処理を i = 10 になるまで処理する。
- i = 10のときに処理が終わったら15行目に進む。

やきにくやきにくやきにくやきにくやきにくやきにくやきにくやきにくやきにくやきにく

- 上のコードを自分のPCでも入力してみよう。

1 から 100 までの和を計算する

- 次のコード（1 から 100 までの和を計算するプログラム）を入力してみよう。
- 単に丸写しするのではなく、コードの意味を考えながら入力していこう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_07-4</title>
7  </head>
8
9  <body>
10     <p>
11         <script>
12             let sum = 0;
13             for(let i=1; i<=100; i++){
14                 sum = sum + i;
15             }
16             document.write("1から100までの和は" + sum + "です。");
17         </script>
18     </p>
19 </body>
20
21 </html>
```

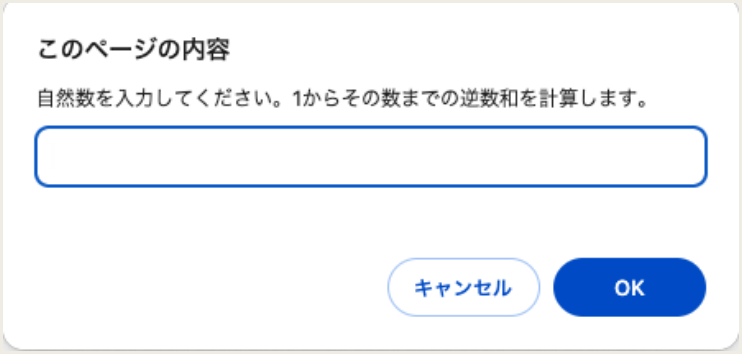
1から100までの和は5050です。

14行目は sum += i;
と書いてもよい

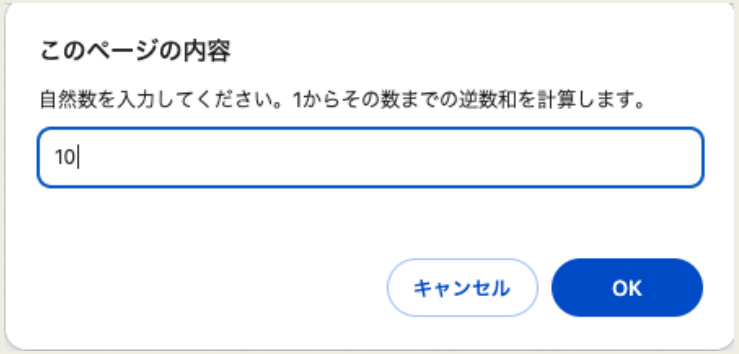
- i は 1 から 100 まで動き、処理ごとに i は1ずつ増える。
- i = 1 のとき
  - 変数 sum に 1 を加える（この時点で sum = 1）。
- i = 2 のとき
  - 変数 sum に 2 を加える（この時点で sum = 3）。
- i = 3 のとき
  - 変数 sum に 3 を加える（この時点で sum = 6）。

[演習] 1 から n までの逆数和を計算する

- ユーザーに自然数 n を入力してもらい、1 から n までの逆数の和（の近似値）を計算するプログラムを作れ。
- 次の出力例も参考にせよ。



$$\sum_{k=1}^{n} \frac{1}{k} = \frac{1}{1} + \frac{1}{2} + \cdots \frac{1}{n}$$



逆数和は2.9289682539682538です。

[演習]合同方程式を解く

$18x \equiv 42 \pmod{17}$

- for文 と if文 を組み合わせると有用なプログラムが書けるようになる。
- 例えば、合同方程式 $3x \equiv 7 \pmod{17}$ を考えてみよう。

3x を 17 で割った余りが 7 となる x を求めよ。

- 両辺を6倍すると、解は $x \equiv 8 \pmod{17}$ であると分かる。
- 別解として、解の候補 $x \equiv 0, 1, 2, \ldots, 16 \pmod{17}$ をしらみつぶしに調べる、という方法も考えられる。
- コンピュータは解をしらみつぶしに全探索することが得意である。
- for文 と if文 を使って、次の合同方程式を解け。

$$x^3 + x^2 + x + 1 \equiv 0 \pmod{365}$$

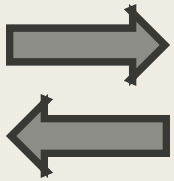
合同方程式の解は x ≡ 27, 72, 119, 173, 192, 218, 319, 338, 364, (mod 365) です。

[余裕のある人向けの課題]
最後のコンマを表示させないようにするにはどうすればよいか？

条件が成立している間、処理を繰り返す

- 次は while文 を扱う。
- for文では for の直後のカッコに「初期化式」「条件式」「制御変数の更新」の3つを記述した。
- while文では while の直後のカッコには「条件式」のみを記述する。
- その条件式が満たされる間(while)、処理を繰り返す。
- for文 は while文 に書き換えることができる（実は逆も真）。

```
for(let i = 1; i <= 10; i++){
    document.write(“やきにく”);
}
```



互いに
書き換え可能

```
let i=1;
while(i <= 10){
    document.write(“やきにく”);
    i++;
}
```

i <= 10 が真である限り
「やきにく」と言い続ける

パスワード

- while文は繰り返し回数が分からない（定まらない）場合に用いられる。
- 次はパスワードが正しく入力されるまで繰り返すプログラムである。コードの意味を考えながら入力してみよう。

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8">
    <title>Prog_08-1</title>
</head>

<body>
    <p>
        <script>
            let password;
            let correctPassword = 1234;
            while(password != correctPassword){
                password = prompt("4桁のパスワードを入力してください。");
            }
            document.write("正しいパスワードが入力されました。");
        </script>
    </p>
</body>

</html>
```

正しいパスワードが入力されました。

無限ループ

- while文は条件式を満たす限り処理を繰り返すため、プログラムに誤りがあるとループから脱却できなくなることがある。
- これを無限ループという。
- 次は「やきにく」と10回言うプログラムを while 文で記述したつもりだったが、制御変数 a を更新し忘れているために無限ループに陥った例である。入力して実行してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_08-2</title>
7  </head>
8
9  <body>
10     <p>
11         <script>
12             let a = 1;
13             while(a <= 10){
14                 document.write("やきにく");
15             }
16         </script>
17     </p>
18 </body>
19
20 </html>
```

14行目のあとに a++; を入れ忘れたため
永久に「やきにく」と言い続けるプログラム

繰り返しを中断する（break文）

- 繰り返しの処理を記述する際、何らかの条件が成立した時点でループを中断したいといったケースがある。
  - 方程式の解をひとつでも見つけられれば良い状況など。
- そのような場合には break文 を使う。
- プログラムは break; に遭遇した時点でループから抜ける。

```
while(true){

    処理

    if(条件式){
        break;
    }

    処理

}
```

無限ループ前提で条件式に
true を置くことがある。

この条件式が満たされれば
while のループから脱却する。

[演習]パスワードその２

- 次の出力結果を参考に、3回パスワードを間違えたらパスワードが入力できなくなるプログラムをかけ。

ヒント

```
<script>
    let password;
    let correctPassword = 1234;
    let counter = 1;
    while(true){
        pas
        if(
        }
        if(
        }
        counter++;
    }
rint>
```

考えてみよう

正しいパスワードが入力されました。

パスワードを3回間違えました。

3回間違える

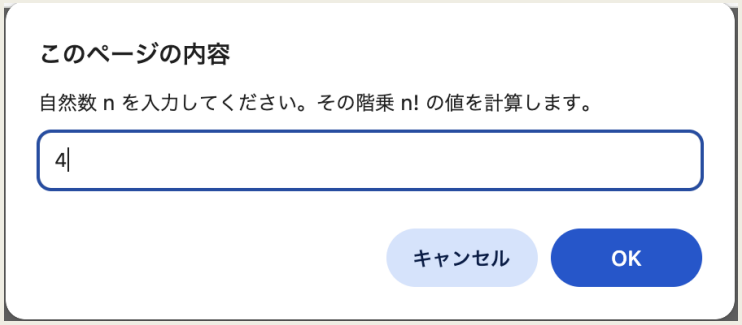
[演習]教科書を熟読しよう

- 今日の内容は教科書の p110～p124 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。
- 教科書も読み終わった人は、以降のスライドの演習問題（予習）にもチャレンジしてみよう。

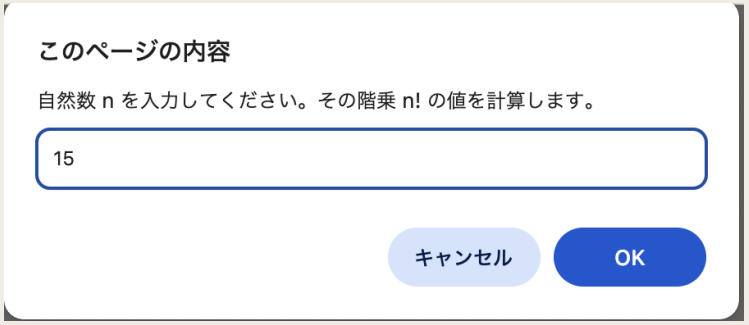
[演習]階乗の実装

今年度新たに用意した問題→ NEW

- ユーザーから入力された自然数 n に対して、nの階乗 n! を計算するプログラムをかけ。ユーザーから入力される n は自然数であることを想定してよい。



4の階乗は 24 です。



15の階乗は 1307674368000 です。

[演習]漸化式の最初の項を列挙する

- 次の漸化式で定義される数列 $\{a_n\}$ の最初の10項を求めよ。

$$a_1 = 1, \quad a_{n+1} = 2a_n + n$$

第1項は 1 です。
第2項は 3 です。
第3項は 8 です。
第4項は 19 です。
第5項は 42 です。
第6項は 89 です。
第7項は 184 です。
第8項は 375 です。
第9項は 758 です。
第10項は 1525 です。

[演習] FizzBuzz

- 「FizzBuzz」は英語圏で長距離ドライブや飲み会のときに行われる言葉遊び。
- プレイヤーは「1」から順に数字を発言していく。
- ただし、3の倍数のときは「Fizz」、5の倍数のときは「Buzz」、両方のときは「FizzBuzz」と発言する。
  - 1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, FizzBuzz, 16, 17, ...
- 1から100までの数について、3の倍数なら「Fizz」、5の倍数なら「Buzz」、両方のときは「FizzBuzz」と出力するプログラムをかけ。次の出力を参考にせよ。

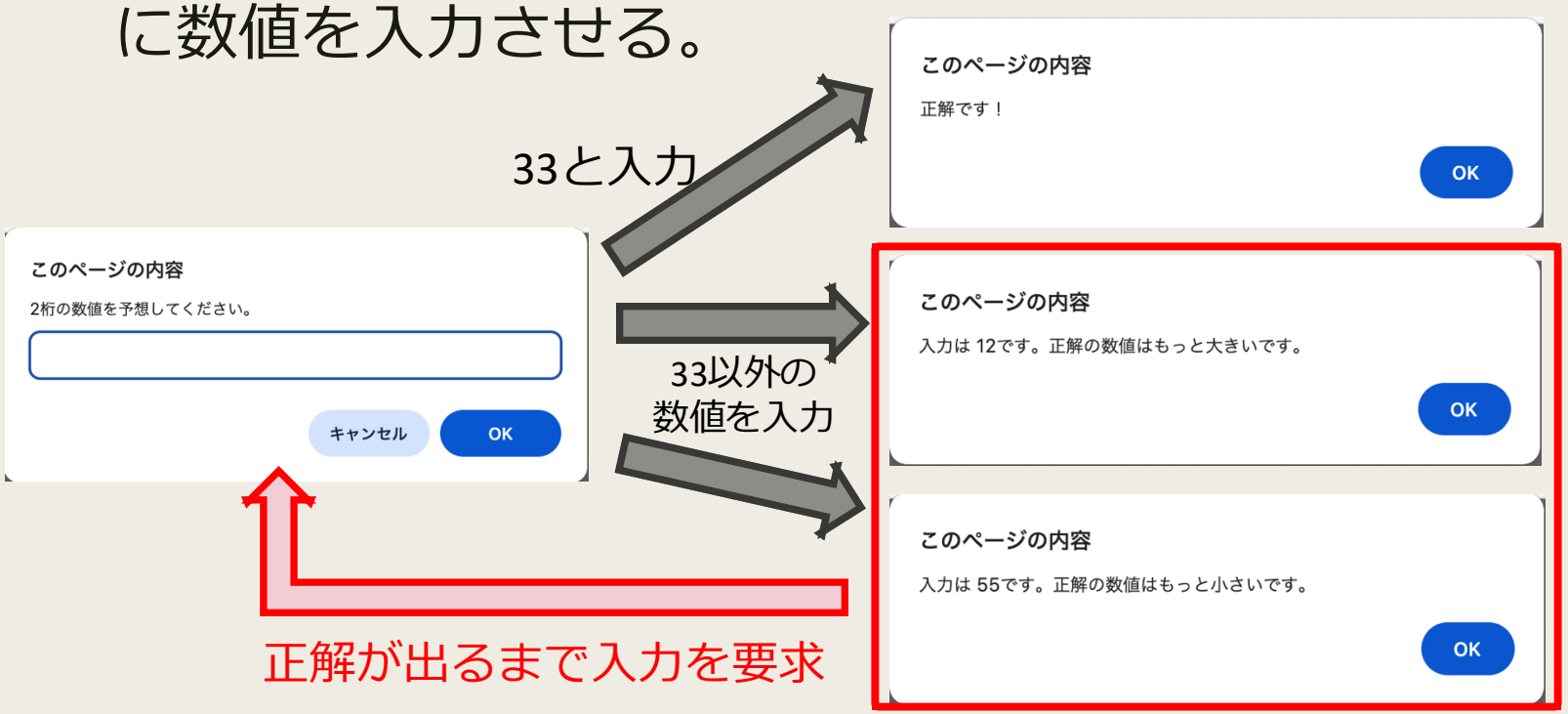
1, 2, Fizz, 4, Buzz, Fizz, 7, 8, Fizz, Buzz, 11, Fizz, 13, 14, FizzBuzz, 16, 17, Fizz, 19, Buzz, Fizz, 22, 23, Fizz, Buzz, 26, Fizz, 28, 29, FizzBuzz, 31, 32, Fizz, 34, Buzz, Fizz, 37, 38, Fizz, Buzz, 41, Fizz, 43, 44, FizzBuzz, 46, 47, Fizz, 49, Buzz, Fizz, 52, 53, Fizz, Buzz, 56, Fizz, 58, 59, FizzBuzz, 61, 62, Fizz, 64, Buzz, Fizz, 67, 68, Fizz, Buzz, 71, Fizz, 73, 74, FizzBuzz, 76, 77, Fizz, 79, Buzz, Fizz, 82, 83, Fizz, Buzz, 86, Fizz, 88, 89, FizzBuzz, 91, 92, Fizz, 94, Buzz, Fizz, 97, 98, Fizz, Buzz,

[演習]数当てゲーム

```
let correctNumber = 33;
//let correctNumber = Math.floor(Math.random()*89 + 10);
// ↑ 正解をランダムにしたい場合
```

NEW

- 2桁の数を当てるゲーム。
- まずは簡単のために答えの数値を「33」に設定。
- 下図のようにユーザーに2桁の数値を入力してもらい…
  - 入力が正しい数値なら「正解です！」と表示。
  - 入力が正しくない場合は、答えの数値が入力よりも大きいか小さいかを知らせ、正解が入力されるまでユーザーに数値を入力させる。

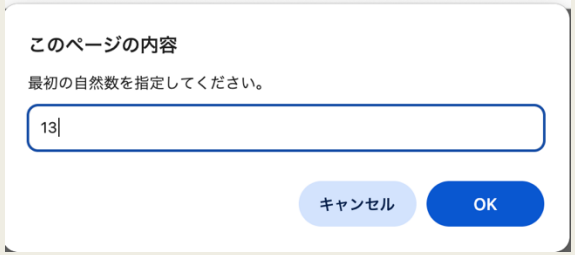


[演習] コラッツの問題

- 自然数をひとつ選び、次の操作を繰り返す。
  - その数が偶数ならば *2* で割る。
  - その数が奇数ならば *3* をかけて *1* を足す。
- 例えば、選んだ自然数が 3 なら、次を経て最終的に 1 に到達する。

$$3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1 \qquad (\bigstar)$$

- 最初の数がどんな自然数であっても、この操作を繰り返すと必ず 1 に到達するだろうか？
- この問題を コラッツの問題 といい、実は現在でも未解決の問題。
- ユーザーから入力された自然数 n に対して、 (★) のような列を表示するプログラムをかけ。



13 → 40 → 20 → 10 → 5 → 16 → 8 → 4 → 2 → 1

[演習] かけ算九九の表（二重for文）

- 二重 for 文 にも挑戦してみよう。
- 次の出力結果を参考に、かけ算九九の表を作ってみよう。
- document.wrtie 命令の引数に表に関するタグ（<tr>, <td>）を入れることで表そのものも for 文を使って構築できる。

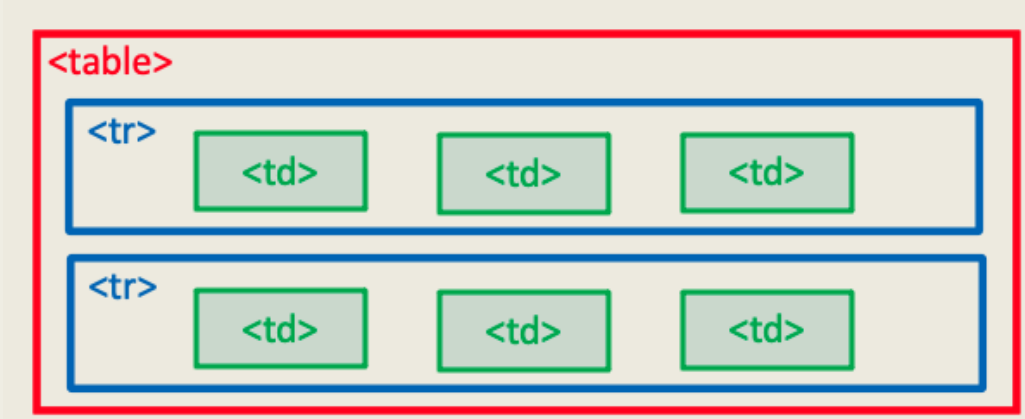
HTMLにおける表のかき方は
第2回の授業で解説しているので
適宜復習しよう。

かけ算九九の表（中身のみ）

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

[演習] 枠付きかけ算九九の表（二重for文 + if文）

- 二重 for 文 と if文 のプログラムにも挑戦してみよう。
- 少し難しいと思うが、二重 for 文と if文 が混在したコードがかけるとプログラムの幅がぐんと広がる。
- 次の出力結果を参考に、枠付きのかけ算九九の表を作ってみよう。
- 枠なしのかけ算九九の表のプログラムも参考にしてみよう。



HTMLにおける表のかき方は
第2回の授業で解説しているので
適宜復習しよう。

かけ算九九の表（枠付き）

×	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81