JS学習教材

第一章：文字を出力する

cosole.log();

- コンソールへ出力する

注意点：

- 文字列は(')シングルクォーテーション、(")ダブルクォーテーションで囲む
  - 文字列とは、「Hello World」などの文字
  - 数値とは、「1」などの数値
    → (+)足し算、(-)引き算、(*)掛け算、(/)割り算、(%)余り
- 末尾に(;)セミコロンをつける

```
//コンソールへ「Hello World」と表示しよう
console.log("Hello World");

//コンソールへ「Hello  + Worldの結果」を表示しよう
console.log("Hello" + "World");

//コンソールへ、「3 + 5 の結果」を表示しよう
console.log(3 + 5 );

//コンソールへ文字列として「3 + 5 」と表示しよう
console.log("3 + 5");
```

//（コメントアウト）

- コメントアウト（コメント、メモとなり、無視される）

```
//コンソールへ「Hello World」と表示しよう
console.log("Hello World");

//コンソールへ、「3 + 5 の結果」を表示しよう
console.log(3 + 5 );

//コンソールへ文字列として「3 + 5 」と表示しよう
console.log("3 + 5");
```

```
//console.log("コメントアウト");をコメントアウトしてみよう
console.log("コメントアウト");
```

第二章：変数と定数

変数

- データ（値）を格納する入れ物

メリット：

- 同じ値を繰り返し使える
- 変更に対応しやすい
- 値の意味が分かりやすい

使い方：

- 変数を定義する(変数名「name」へ、データ（値）「Menthuthuyoupi」)

```
let name = "Menthuthuyoupi";
```

- 変数を使用する(コンソールへ「Menthuthuyoupi」を出力する)

```
console.log(name);

//(")ダブルクォーテーションで囲むと、「name」が出力される
console.log("name");
```

命名のルール：（※が付いているものは、エラーは出ないが、ふさわしくない）

○	
number	英単語
oddNumber	２語以上のの場合は大文字で区切る

×	
1number	数字開始
bango	ローマ字※
番号	日本語※

上書き：

- 一度代入した値を更新できる
- 更新するときはletをつけない

```
let name = "Menthuthuyoupi";
console.log(name);
//nameを更新
name = "Neferpitou";
console.log(name);

//変数自体を使用して更新できる
let number = 3;
console.log(number);
//numberに5を足して更新
number = number +5;
```

加算などを省略して書くことも可能

```
x = x + 3;
x = x - 3;
x = x * 3;
x = x / 3;
x = x % 3;
```

```
x += 3;
x -= 3;
x *= 3;
x /= 3;
x %= 3;
```

定数

- 変数と違い、値を更新できません

使い方：

- constを使用して定義します

```
const name = "Shaiapouf";
```

テンプレートリテラル

- 文字列の中に、定数（変数）を埋め込むことができます

使い方：

- (`)バッククォーテーションで囲む
- 定数（変数）を${}で囲む

```
const name = "Morel";
const max = 216;

console.log(`${name}は、最大で${max}体のディープパープルを出せます。`)
```

第三章：条件分岐

if文

- 「もし〇〇ならば▲▲を行う」という条件分岐をすることができます
- 条件式が「成り立つ = 真(true)」、「成り立たない = 偽(false)」となります
- 条件を追加する場合は、`else if` を使用します
- その他の場合は、`else` を使用します

使い方：

- if(条件式){ 処理 }
  条件式がtrueの場合に、処理を実行する
- if文の最後には(;)セミコロンはいらない

```
const number = 12;

if(number > 20) {
    console.log("20よりも大きいです");
}else if(number > 10){
    console.log("10よりも大きいです");
}else {
    console.log("10未満です");
}

//条件式の結果を出力することもできます。この場合は、「true」と出力されます
console.log(number > 10);
```

条件式：

1. 不等号

```
//bより大きい
a > b
//bより小さい
```

```
a < b

//b以上(bを含む)
a >= b
//b以下(bを含む)
a <= b
```

2. 等価演算子(文字列と数値は区別しない)(a === 2 と a === "2"は等しくない)

```
//等しい(a == 2 と a == "2"は等しい)
a == b

//異なる
a != b
```

3. 厳密等価演算子(文字列と数値を区別する)(a === 2 と a === "2"は等しくない)

```
//等しい
a === b

//異なる
a!== b
```

条件式を複合：

1. かつ（どちらもtrue）

```
a >= 2 && b >=3
```

2. または（どちらかtrue）

```
a >= 2 || b >= 3
```

swich文

- 値に応じて処理を分岐する

使い方：

- swich(値){処理}
- swich文の末尾には(;)セミコロンはいらない

- `break` を使用して処理を打ち切る
- `default` を使用してどれにも当てはまらない場合の処理

```
const nen_category = "Transmuter";

swich(nen_category){
    case Enhancer:
        console.log("単純で一途");
        break;
    case Transmuter:
        console.log("気まぐれで嘘つき");
        break;
    case Manipulator;
        console.log("理屈屋、マイペース");
        break;
        .
        .
        .
    default:
        console.log("何も見えないでしょ？見えるようになったらもっぺんおい
        break;
}
```

第四章：配列

第五章：オブジェクト

第六章：繰り返し処理

第七章：関数

第八章：アロー関数

第九章：クラス

第十章：コールバック関数