ループ処理

配列を1つずつ取り出して処理したい場合に配列の数だけ処理を書くのはかなり効率が悪いのと、バグを生む原因となります。

ループ処理を作ることで配列の数が変わってもコードを変えることなく処理ができます。

for

`for`を使うことによってループの処理を作ることができます。

```
for (var i = 1; i < 10; i++) {
  console.log("ループ" + i + "回目");
};
```

これをコピペしてコンソールに表示させてみてください。

1~9までの数字が表示されたかと思います。

そのように処理されたいるのかイメージし難いと思うのでパーツごとに分けて説明します。

`for()`で繰り返しの処理を実行します。

`var i = 1;`iに初期値として1を代入しています。

`i < 10;`これはループの条件を指定しています。

iが10と等しいか大きくなった時にループの処理を終了します。

`i++`でループの処理が終わった時に1を足しています。

これを忘れてしまうと無限ループしてしまうので気をつけてください。

for in

上記の方法はループの回数を指定する必要があります。

配列は数が変わりやすいのでその都度ループ条件を書き換えるのも面倒なので`in`を使用してindex番号を引き渡して取り出す方法があります。

```
var items = ["リンゴ", "バナナ", "イチゴ"];
for (index in items) {
  console.log(items[index]);
};
```

for of

先ほどはindex番号を指定して配列を取り出していました。

`of`を使うと配列の中身ごと順番に取り出せます。

```
var teachers = [ { name: "太郎", age: 25, subject: "国語", }, { name: "浩司", age: 30, subject: "数学", }, { name: "花子", age: 26, subject: "社会", } ] for (teacher of teachers) { console.log(teacher); };
```

ループの処理を3つほど紹介しましたがどれを使うのが正しいのかは処理したいもの、結果によって変わってきます。

どれを使うのが正しいのか判断して使うようにしましょう。

課題

1. `for`を使ったループ処理を作ってください。
2. `for in`を使ったループ処理を作ってください。
3. `for of`を使ったループ処理を作ってください。

注意点

処理内容が説明したものとまったく同じだった場合、コピペしているだけと判断し再度作成していただきます。