JavaScript 入門　テキスト課題　最終補足　アロー関数

アロー関数は関数式をシンプルに記述できるようにしたものです。昔からの定義方法と比べてプログラムの動きが異なる点があります。this の取り扱いです。(this の詳細はもっと複雑で、当講座では取り扱いませんが、昔からの定義方法の関数がメソッドとして呼び出された場合、this は呼び出し元になります。アロー関数では外側の関数から this を受け継ぎます。)

保存時ファイル名：textb-arrow-sample-1.html、textb-arrow-1.html 等。

＜サンプルプログラム＞　※入力・実行は必要ありません

```
//関数定義
function dispName(name){
	document.write(name + "<br>");
}
//関数式
const dispName = function (name){
	document.write(name + "<br>");
}
```

・関数式：変数等に関数を値として代入し、後からその変数を呼び出して関数を利用する方法

```
//アロー関数化 1　function の省略
const keisyou = (name){
	return name + "様";
}
//アロー関数化 2　アロー演算子（=>）追加
const keisyou = (name) => {
	return name + "様";
}
//アロー関数化 3　()の省略可
const keisyou = name => {
	return name + "様";
}
//アロー関数化 4　{}の省略可
const keisyou = name =>
	return name + "様";

//アロー関数化 5　return の省略可
const keisyou = name =>
	name + "様";
```

・function を省略できる
※これだけではエラー

・アロー演算子を置くことで実行環境はアローの左が引数で右が処理ということを理解できる
※ここまでで、エラーは出なく正常

・引数が 1 つの関数は()を省略可（引数 0 や複数の場合は省略できない）

・関数内処理が 1 行で済む場合は{}を省略可（なお、=>のすぐ左では改行するとエラー）
※左のように return があるとエラー

・{}を省略した場合、return を省略できる

```
class Cat{
   name = "猫太郎";
   //関数
   dispThis = function(){
      return function(){ console.log( this ) };
   }
   //アロー関数
   dispThis2 = function(){
         return ()=>{console.log( this )};
   }

};
let cat = new Cat();
cat.name = "猫三郎";
cat.dispThis()();
cat.dispThis2()();
```

ややこしいので、ざっと目を通すだけで OK。
とりあえず this の取り扱いが違うことのみなんとなく覚えておいてください

・this は undefined

・this は cat2

・return で戻ってきた関数を実行
・return で戻ってきた関数を実行

＜練習問題＞

アロー関数はサンプルでいうところのアロー関数化 2 までとして下さい。

練習 1

アロー関数を定義し、変数 keikoku に代入して下さい。引数、戻り値は無し。ダイアログボックスで「注意！」を表示して下さい。

keikoku を呼び出して下さい。

正常結果例：「注意！」が表示

練習 2

アロー関数を定義し、変数 kakeruData に代入して下さい。引数 1 と引数 2 に数値が入り、かけ算した結果を戻り値にして下さい。

変数「kazu1」、「kazu2」を作成し、結果例を参考にデータを代入して下さい。
kazu1、kazu2 を引数に指定し、kakeruData を呼び出して下さい。
戻り値を使用して、結果例を参考に画面表示して下さい。

正常結果例：「1」「2」の時、「1 * 2 = 2」が表示