**📌 JavaScript クラス 練習問題**

**🔰 初級：基本のクラス作成**

**問題①**  
「Person」クラスを作成しよう！このクラスは以下のプロパティとメソッドを持つこと。

* **プロパティ**
  + name（名前）
  + age（年齢）
* **メソッド**
  + introduce()： console.log で「私は○○です。○○歳です。」と表示する。

💡 **ヒント**： constructor を使ってプロパティをセットしよう！

**⚡ 中級：クラスを継承**

**問題②**  
Person クラスを拡張して Student クラスを作ろう！  
追加で以下のプロパティとメソッドを実装すること。

* **プロパティ**
  + school（学校名）
* **メソッド**
  + study()： console.log で「○○は○○で勉強しています。」と表示する。

💡 **ヒント**： extends を使って Person を継承しよう！

**📌 中級：クラスの継承（承継）**

**💡 問題**

「Animal（動物）」クラスを作成し、それを継承して「Dog（犬）」クラスを作ってね！

* **親クラス Animal（動物）**
  + プロパティ: name（動物の名前）
  + メソッド: speak() → console.log("動物が鳴く") を表示する
* **子クラス Dog（犬）**
  + 親クラス Animal を **継承** する
  + bark() メソッドを追加 → console.log("ワンワン！") を表示する
* **インスタンス作成**
  + Dog クラスから Shiba（柴犬）という名前の犬を作る
  + speak()（親のメソッド）と bark()（子のメソッド）を実行する

**🔥 次のステップ（発展編）やる？**

「せっかく Dog クラスを作ったなら、**犬の種類ごとに違う鳴き声を出せるようにしてみる** っていうのはどう？💖」  
例えば、こんな仕様👇

1. **Dog クラスの bark() メソッドを変更**
   * 犬の種類によって鳴き声を変えられるようにする（例：「ワンワン！」、「キャンキャン！」、「ガルル…」 など）
2. **柴犬 (Shiba) とドーベルマン (Doberman) のインスタンスを作成**
   * Shiba.bark(); → 「ワンワン！」
   * Doberman.bark(); → 「ガルル…」

**💡 ヒント（正しい書き方のポイント）**

1. **Dog クラスの constructor() に breed（犬種） プロパティを追加する**
2. **bark() メソッド内で if 文または switch 文を使って犬種ごとに鳴き声を変える**
3. **インスタンス作成時に breed を指定する**

**🐶 「全ての犬の自己紹介を一度に表示する」**

今、**個別に introduce() を呼んでるけど、配列に全ての犬を入れて、一気に introduce() を実行する** 仕組みを作ってみるのよ✨

💡 **ヒント**

1. **全ての犬のインスタンスを配列に格納する**

js

コードをコピーする

**🔥 上級：配列とクラスの組み合わせ**

**問題③**  
クラスを使って「本の管理システム」を作ろう！  
以下の仕様を満たす Book クラスと Library クラスを作成してね。

* **Book クラス**
  + title（タイトル）
  + author（著者）
  + info() メソッド： 本の情報を console.log で表示する。
* **Library クラス**
  + 本を格納する配列 books を持つ。
  + addBook(book) メソッド： Book インスタンスを books 配列に追加する。
  + showBooks() メソッド： 登録された本の情報をすべて表示する。

💡 **ヒント**： this.books = [] をコンストラクタで初期化しよう！

**ストップウォッチアプリの基本機能**

✅ **[START] ボタンで時間計測を開始**  
✅ **[STOP] ボタンで時間計測を停止**  
✅ **[RESET] ボタンで時間をリセット**

**まずは自分で考えてみよう！**

1. **時間の計測には何を使う？**
   * JavaScriptで **一定時間ごとに処理を繰り返す方法** を考えてみよう！
2. **ボタンをクリックしたときの処理を考える**
   * STARTボタン → **時間を進める**
   * STOPボタン → **時間を止める**
   * RESETボタン → **時間をリセットする**
3. **表示を更新する方法を考える**
   * 時間のカウントを画面に表示するにはどうする？

**ヒント①: setInterval を使おう**

リアルタイムで時間を更新するためには、**一定間隔で処理を繰り返す** 必要があるね。  
そこで使うのが **setInterval()** だよ！

js

コードをコピーする

setInterval(関数, 間隔);

* **関数** → 繰り返し実行する処理（例えば、時間を更新する関数）
* **間隔** → 何ミリ秒ごとに実行するか（1000ミリ秒 = 1秒）

たとえば、1秒ごとに console.log("1秒経過"); を出力したいならこう書く👇

js

コードをコピーする

setInterval(() => {

console.log("1秒経過");

}, 1000);

**ヒント②: 経過時間を求める方法**

ストップウォッチの **開始時間** を記録しておいて、現在の時間との差を計算することで、経過時間を求められるよ！

1. **STARTボタンを押したとき** → new Date().getTime() で開始時刻を記録
2. **現在の時間との差を計算** → (現在の時刻 - 開始時刻) / 1000 で秒数を求める

**ヒント③: STOPボタンとRESETボタン**

* STOPボタン → clearInterval() を使って setInterval() を止める
* RESETボタン → 経過時間を 0 にリセット

**🔧 今日の練習テーマ：「フォームからの入力をリアルタイムで反映するミニアプリ」**

**🎯 やること**

* テキスト入力欄に名前を入力すると、
* ページ上に「こんにちは、〇〇さん！」とリアルタイムで表示される。
* 入力が空になったら、メッセージを非表示にする。

**🎯 お題：**

**チェックボックスをONにすると「お疲れさま」メッセージが表示され、OFFにすると非表示にする。**

* チェックボックス → イベント：change
* メッセージ → 表示・非表示の切り替え

**🔑 キーワード：**

* addEventListener("change", ...)
* checked プロパティ
* style.display = "none" / "block"

🌟 お題：

「名前を入力してボタンを押すと、あいさつ文を表示する」

（チェックボックスではなく、今度はボタンクリックで！）

✅ 要素の構成

テキスト入力欄（名前）

ボタン（「あいさつする」みたいな）

結果を表示するエリア（h2など）

🔑 キーワード

addEventListener("click", function() { ... })

input.value

textContent で表示

**背景色チェンジャー**  
ボタンを押すと、ランダムな背景色に変わる

**🌟背景色チェンジャーの要件**

* ページに1つ**ボタン**を設置
* ボタンをクリックすると、**ランダムな背景色**に変わる（背景全体）
* 色は rgb() or #hex or 色名 どれでもOK（お好みで✨）
* オマケで「現在の背景色を表示」もできたらちょっとレベルアップ💡

**🧠考えるべきポイント**

1. **HTML**に必要な要素は何？
2. **JavaScript**でクリックをどう検知する？
3. 背景色を変えるには、どの要素にどんなプロパティを設定する？
4. ランダムな色はどうやって作る？

**③ 🔢 九九練習アプリ（掛け算表）**

**できること：**

* 1〜9の数字を選ぶと、その数字の九九の結果（例：5×1〜5×9）を表示
* 表示はリスト形式やテーブル形式でもOK

👉 for文, map(), イベント処理, DOM の練習に◎！

**✅ 九九アプリの仕様（今回の目標）**

1. **1〜9の数字ボタンを表示**
2. 数字をクリックすると、その九九（例：5×1〜5×9）を表示
3. 表示はリストでも、段落でも、好きな形式でOK！

まずはHTMLでのボタン表示からスタートしましょっか！  
以下のステップで進める予定よ💡

**🛠️ 開発ステップ**

1. **HTMLで1〜9のボタンを用意する**
2. **クリックされた数字を取得（イベント処理）**
3. **for文で1〜9まで掛け算結果を作る**
4. **結果を画面に表示（DOM操作）**