4. Boucles

- **`for` loop** : Pour itérer sur des éléments d'un tableau ou une plage de nombres.

```
javascript

for (let i = 0; i < 5; i++) {
  console.log(i);
}
```

- **`while` loop** : Une boucle qui s'exécute tant qu'une condition est vraie.

```
javascript

let i = 0;
while (i < 5) {
  console.log(i);
  i++;
}
```

5. Fonctions

- **Déclaration de fonction** : Syntaxe de base.

```
javascript

function saluer(nom) {
  console.log("Bonjour " + nom);
}

saluer("Alice");
```

- **Fonctions fléchées** : Syntaxe plus concise.

```
javascript

const addition = (a, b) => a + b;
console.log(addition(2, 3)); // Affiche 5
```

6. Introduction au DOM

- **Manipulation du DOM avec JavaScript** : Modifier le contenu ou les styles d'un élément HTML via JavaScript.
  - Sélectionner un élément : `document.getElementById()`, `document.querySelector()`.
  - Modifier un élément : `.innerText`, `.style`, `.classList`.

```
javascript

let titre = document.getElementById("mon-titre");
titre.innerText = "Nouveau Titre";
titre.style.color = "blue";
```

- **Ajouter un événement** : Réagir aux actions de l'utilisateur.

```
javascript

let bouton = document.getElementById("mon-bouton");
bouton.addEventListener("click", function() {
  alert("Bouton cliqué !");
});
```

7. Conclusion et récapitulatif

- Récapitulation des concepts abordés : variables, types de données, conditions, boucles, fonctions, manipulation du DOM.
- Importance de la pratique pour bien comprendre JavaScript.

Pour tester JavaScript, tu peux utiliser plusieurs outils et logiciels. Voici quelques suggestions adaptées pour débuter avec JavaScript :

1. Éditeurs de texte locaux (à installer sur ton ordinateur)

Tu peux écrire ton code JavaScript directement sur ton ordinateur et le tester en l'exécutant dans ton navigateur. Voici les étapes à suivre :

Éditeurs de texte populaires :

- **Visual Studio Code** (VS Code) : Très populaire et puissant pour les développeurs. Il propose des fonctionnalités comme l'autocomplétion, la coloration syntaxique et des extensions pour faciliter le développement web.
  - **Télécharger VS Code** : Visual Studio Code
- **Sublime Text** : Un éditeur léger et rapide, parfait pour écrire rapidement du JavaScript.
  - **Télécharger Sublime Text** : Sublime Text
- **Atom** : Un éditeur open-source développé par GitHub, avec des fonctionnalités similaires à VS Code.
  - **Télécharger Atom** : Atom

Étapes :

1. **Créer un fichier HTML** : Crée un fichier `index.html` pour intégrer ton code JavaScript.
2. **Ajouter du code JavaScript** : Utilise une balise `<script>` dans ton fichier HTML pour inclure du code JavaScript. Exemple :

```
html

<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <title>Exercice JavaScript</title>
</head>
<body>
    <h1>Bienvenue sur ma page</h1>
    <button id="mon-bouton">Clique-moi</button>
    <script>
        document.getElementById('mon-bouton').addEventListener('click',
function() {
            alert("Bonjour !");
        });
    </script>
</body>
</html>
```

3. **Ouvrir le fichier dans un navigateur** : Double-clique sur le fichier `index.html` pour l'ouvrir dans un navigateur. Tu peux tester tes exercices en modifiant le code et en rafraîchissant la page.

2. Outils en ligne pour tester du JavaScript (sans installation)

Si tu ne veux pas installer d'éditeur sur ton ordinateur, il existe plusieurs outils en ligne très pratiques pour tester ton code JavaScript directement dans ton navigateur.

CodePen :

Un environnement en ligne qui permet d'écrire et de tester du HTML, CSS et JavaScript en temps réel.

- **Site web** : CodePen
- **Avantages** : Interface simple, résultats instantanés, possibilité de partager le code facilement.
- **Exemple** : Crée un "pen" et écris ton HTML et JavaScript directement dans la fenêtre de l'éditeur.

JSFiddle :

Un autre excellent outil en ligne pour tester du JavaScript avec HTML et CSS.

- **Site web** : JSFiddle
- **Avantages** : Très simple, interface claire, permet d'ajouter des bibliothèques externes comme jQuery.
- **Exemple** : Crée un nouveau "fiddle", puis entre ton HTML et JavaScript dans les zones appropriées.

Repl.it :

Un environnement en ligne où tu peux coder et exécuter des programmes dans plusieurs langages, y compris JavaScript.

- **Site web** : Replit
- **Avantages** : Crée un projet en JavaScript, testez-le en ligne, et accède à des projets partagés.

JSBin :

Un autre éditeur en ligne pour JavaScript et HTML.

- **Site web** : JSBin
- **Avantages** : Simple d'utilisation, supporte les bibliothèques externes, résultats en temps réel.

3. Console du navigateur (outil de développement intégré)

Tous les navigateurs modernes (Chrome, Firefox, Edge) offrent un **outil de développement** qui comprend une **console JavaScript**. Cela te permet de tester rapidement du code JavaScript sans avoir besoin de créer un fichier HTML.

Étapes pour utiliser la console :

1. Ouvre ton navigateur (par exemple, **Google Chrome**).
2. Clique sur **F12** ou **Ctrl+Shift+I** pour ouvrir les outils de développement.
3. Va dans l'onglet **Console**.
4. Tu peux maintenant taper directement du code JavaScript et voir les résultats immédiatement.

Exemple :

```
javascript

console.log("Bonjour, JavaScript !");
```

Cela affichera `"Bonjour, JavaScript !"` dans la console.

4. Node.js (option avancée)

Si tu veux aller plus loin et exécuter du JavaScript côté serveur, tu peux installer **Node.js** sur ton ordinateur. C'est un environnement d'exécution JavaScript qui te permet de tester du JavaScript dans un terminal.

- **Télécharger Node.js** : Node.js

- **Utilisation** : Une fois installé, tu peux exécuter des fichiers JavaScript en ligne de commande. Par exemple, en sauvegardant ton code dans un fichier `app.js` et en l'exécutant avec `node app.js`.

Résumé des options :

- **Éditeurs locaux** (VS Code, Sublime Text, etc.) : Idéal pour une expérience de développement complète, tu travailles sur des fichiers sur ton ordinateur et tu les testes dans un navigateur.
- **Outils en ligne** (CodePen, JSFiddle, Repl.it) : Pratique pour des tests rapides et des exercices en ligne.
- **Console du navigateur** : Très rapide pour tester de petits bouts de code.
- **Node.js** : Si tu veux exécuter du JavaScript côté serveur ou tester des scripts plus complexes.

Travaux pratiques (TP) – 2 heures

1. TP 1 : Manipulation des variables et opérateurs (30 minutes)

Objectif : S'exercer à utiliser les variables, les opérateurs, et les conditions simples.

Exercice 1 : Calculer l'âge

Demander à l'utilisateur de saisir son année de naissance et calculer son âge en fonction de l'année actuelle.

Exercice 2 : Vérification d'un mot de passe

Écrire un programme qui vérifie si un mot de passe est suffisamment long (plus de 8 caractères) et si la première lettre est une majuscule.

2. TP 2 : Les boucles et le DOM (30 minutes)

Objectif : Manipuler les boucles et modifier dynamiquement le contenu d'une page web.

Exercice 1 : Afficher une liste

Créer une liste HTML (ul) avec plusieurs éléments de fruits. Utiliser une boucle pour ajouter ces éléments dynamiquement via JavaScript.

Exercice 2 : Modifier le style d'un élément

Écrire une fonction qui change la couleur de fond de la page chaque fois que l'utilisateur clique sur un bouton.

3. TP 3 : Fonctions et événements (30 minutes)

Objectif : Utiliser des fonctions et des événements pour créer des interactions avec l'utilisateur.

Exercice 1 : Calculatrice simple

Créer une calculatrice avec des boutons pour les opérations de base (addition, soustraction, multiplication, division). Chaque bouton doit appeler une fonction qui effectue l'opération correspondante.

Exercice 2 : Formulaire interactif

Créer un formulaire HTML avec un champ de texte et un bouton. Lorsque le bouton est cliqué, afficher un message contenant le texte saisi dans le champ.

4. TP 4 : Projet final (30 minutes)

Objectif : Mettre en pratique les concepts appris pendant le cours pour créer une application simple.

Exercice 1 : To-Do List

Créer une application de gestion de tâches (to-do list) où l'utilisateur peut :

- Ajouter une tâche.
- Marquer une tâche comme terminée.
- Supprimer une tâche.

Cela impliquera l'utilisation du DOM pour manipuler les éléments HTML et de fonctions pour gérer les interactions utilisateur.