Exercice 1 — Mouvement d'un rectangle interactif

Objectifs pédagogiques :

- Manipuler le **DOM** avec JavaScript.
- Gérer des **événements utilisateurs** (clic, maintien).
- Modifier dynamiquement des **styles CSS**.
- Comprendre la notion de **positionnement et transformation d'éléments** dans une page.

Objectif :

Créer une petite interface web en **HTML, CSS et JavaScript** permettant d'afficher un rectangle divisé en quatre parties colorées, centré au milieu de la fenêtre du navigateur, et pouvant se déplacer vers la gauche ou la droite à l'aide de boutons.

Consignes :

1. **Structure de base (HTML)**
    - ✓ Crée une page index.html.
    - ✓ Ajoute un élément principal représentant le **rectangle global**.
    - ✓ Ce rectangle doit contenir **4 sous-rectangles** (divs) disposés de manière à former une grille 2x2.
    - ✓ En dessous du rectangle, ajoute deux boutons :
        - Un bouton **"Gauche"**
        - Un bouton **"Droite"**
2. **Mise en forme (CSS)**
    - ✓ Le rectangle principal doit être **centré horizontalement et verticalement** dans la page.
    - ✓ Chaque sous-rectangle doit avoir une **couleur différente**.
    - ✓ Le rectangle global doit avoir une taille fixe (par exemple 200px de haut × 200px de large).
    - ✓ Les sous-rectangles doivent se partager l'espace de manière égale.
3. **Comportement (JavaScript)**
    - ✓ Lorsque l'utilisateur **maintient le clic** sur le bouton "Droite", le rectangle **se déplace progressivement vers la droite**.

- ✓ Lorsque l'utilisateur **relâche** le bouton, le rectangle **revient au centre**.
- ✓ Même principe pour le bouton "Gauche".
- ✓ Le déplacement doit se faire de manière fluide (utilisation de setInterval() ou requestAnimationFrame() possible).

Astuces :

- Pense à manipuler la position du rectangle via **CSS (transform: translateX(...))** ou **positionnement absolu (left, right)**.
- Utilise les événements mousedown, mouseup, ou mouseleave pour détecter quand un bouton est maintenu ou relâché.
- Tu peux utiliser window.innerWidth pour obtenir la taille du navigateur.

Exercice 2 — Menu latéral rétractable (Sidebar)

Objectifs pédagogiques :

- Manipuler le **DOM** avec JavaScript.
- Gérer des **événements de clic**.
- Comprendre les **animations CSS simples** et les **transformations**.
- Travailler la **mise en page dynamique** avec Flexbox et positionnement.

Objectif :

Créer une page web contenant un **menu latéral à gauche** (sidebar) pouvant être **rétracté** et **déployé** à l'aide d'un bouton.
Un texte principal doit rester **toujours centré** dans la partie principale de la page, même lorsque le menu est masqué.

Consignes :

1. **Structure (HTML)**
    - Crée une page index.html.
    - La page doit contenir deux zones principales :
        - Un **menu latéral gauche** ( ), qui occupe environ **20% de la largeur**.

        - Une **zone principale de contenu** ( ) qui occupe le reste de l'espace.
    - En haut du menu, ajoute un **bouton** (<button id="toggle-btn">) qui permettra d'ouvrir ou de fermer le menu.
        - Ce bouton doit afficher une **flèche vers la gauche ("◀")** lorsque le menu est ouvert.
        - Et une **flèche vers la droite ("▶")** lorsque le menu est fermé.
    - Dans la zone principale (#main), affiche un texte (par exemple : *"Bienvenue sur ma page !"*) **centré horizontalement et verticalement**.

2. **Mise en forme (CSS)**

- Le menu doit être **fixé à gauche** de l'écran.
- Donne-lui une couleur de fond (par exemple, gris clair).
- Lorsqu'il est **fermé**, il doit se **rétracter vers la gauche** jusqu'à disparaître (tu peux le faire avec transform: translateX(-100%) ou width: 0).
- Le contenu principal doit s'ajuster automatiquement en largeur.
- Le texte dans la zone principale doit rester **parfaitement centré**, peu importe l'état du menu.

3. **Comportement (JavaScript)**
    - Quand l'utilisateur clique sur le **bouton du menu**, le menu :
        - **Se cache** s'il est actuellement visible.
        - **Réapparaît** s'il est caché.
    - Le **symbole du bouton** doit changer en conséquence :
        - ◀ quand le menu est visible,
        - ▶ quand il est masqué.
    - Le mouvement du menu doit être **fluide** (ajoute une transition CSS sur la propriété `transform` ou `width`).

4. **Bonus**

- Ajoute une **ombre** sur le menu quand il est visible.
- Fais en sorte que le menu reste visible uniquement sur grand écran (disparaît par défaut sur mobile).

Astuces :

- Utilise classList.toggle() pour activer/désactiver une classe CSS (.hidden, par exemple).
- Utilise transition en CSS pour rendre le mouvement fluide.
- Pour centrer le texte principal, tu peux utiliser **Flexbox** :

  display: flex;

  align-items: center;

  justify-content: center;

  height: 100vh;