# Exercices Avancés sur les Boucles en C

## Challenge 1 : Table de Multiplication Inversée

Créez un programme qui affiche une table de multiplication en ordre décroissant (de 10 à 1).

**Exemple pour le nombre 5 :**

5 × 10 = 50
5 × 9 = 45
...
5 × 1 = 5

**Objectif :** Maîtriser les boucles avec décompte décroissant.

---

## Challenge 2 : Pyramide d'Étoiles à Largeur Impaire

Développez un programme qui dessine une pyramide avec un nombre impair d'étoiles par ligne.

**Pattern pour 7 lignes :**

```
    *      (1 étoile)
   ***     (3 étoiles)
  *****    (5 étoiles)
 *******   (7 étoiles)
********* (9 étoiles)
*********** (11 étoiles)
************* (13 étoiles)
```

**Formule :** Ligne i → `(2*i - 1)` étoiles avec `(n - i)` espaces

**Objectif :** Combiner boucles imbriquées et calculs de motifs géométriques.

---

## Challenge 3 : Générateur de Nombres Premiers

Créez un programme qui affiche tous les nombres premiers de 1 à n.

**Définition :** Un nombre premier n'est divisible que par 1 et lui-même.

**Algorithme :** Pour chaque nombre, tester la divisibilité de 2 à √nombre

**Objectif :** Implémenter un algorithme de test de primalité avec boucles imbriquées.

---

## Challenge 4 : Inverseur de Nombre (Sans Tableaux)

Développez un programme qui inverse les chiffres d'un nombre entier.

**Exemple :** 12345 → 54321

**Méthode :**

1. Extraire le dernier chiffre avec `nombre % 10`
2. Construire le nombre inversé
3. Supprimer le dernier chiffre avec `nombre / 10`

**Objectif :** Manipuler les chiffres individuels avec des opérations arithmétiques.

---

## Challenge 5 : Sommateur de Nombres Naturels

Créez un programme qui calcule la somme des n premiers nombres naturels.

**Exemple :** n = 5 → 1 + 2 + 3 + 4 + 5 = 15

**Vérification :** Comparer avec la formule `n(n+1)/2`

**Objectif :** Utiliser des boucles pour des calculs cumulatifs.

---

## Challenge 6 : Recherche de Facteurs

Développez un programme qui trouve tous les facteurs (diviseurs) d'un nombre.

**Exemple :** 36 → Facteurs : 1, 2, 3, 4, 6, 9, 12, 18, 36

**Méthode :** Tester la divisibilité de 1 à n

**Optimisation :** Tester seulement jusqu'à √n

**Objectif :** Appliquer des tests de divisibilité avec des boucles.

---

# Challenge 7 : Suite de Fibonacci

Créez un programme qui génère les n premiers termes de la suite de Fibonacci.

**Définition :** F(0) = 0, F(1) = 1, F(n) = F(n-1) + F(n-2)

**Séquence :** 0, 1, 1, 2, 3, 5, 8, 13, 21...

**Objectif :** Implémenter une suite récurrente avec gestion de deux variables précédentes.

---

# Challenge 8 : Recherche Dichotomique

Développez un programme qui recherche un élément dans un tableau trié par dichotomie.

**Principe :** Diviser l'espace de recherche en deux à chaque étape

**Algorithme :**

1. Calculer l'index du milieu
2. Comparer avec la valeur recherchée
3. Ajuster les bornes selon le résultat

**Objectif :** Implémenter un algorithme de recherche efficace (O(log n)).

---

# Challenge 9 : Calculateur de Puissance (Boucle While)

Créez un programme qui calcule base^exposant en utilisant une boucle `while`.

**Méthode :** Multiplication répétée de la base

**Cas particuliers :**

- Exposant = 0 → Résultat = 1
- Base = 0 → Résultat = 0

**Objectif :** Maîtriser la boucle `while` pour des calculs itératifs.

---

# Challenge 10 : Générateur de Mot de Passe Aléatoire

Développez un programme qui génère un mot de passe aléatoire de longueur n.

**Caractères autorisés :**

- Lettres majuscules (A-Z)
- Lettres minuscules (a-z)
- Chiffres (0-9)

**Méthode :** Utiliser `rand()` pour sélectionner aléatoirement dans ces ensembles

**Objectif :** Combiner génération aléatoire et manipulation de caractères.

---

# Challenge 11 : Calculateur de Moyenne (Saisie Terminée par 0)

Créez un programme qui calcule la moyenne d'une série de nombres positifs.

**Condition d'arrêt :** L'utilisateur saisit 0

**Note :** Le 0 ne doit pas être inclus dans le calcul

**Objectif :** Utiliser une boucle avec condition d'arrêt dynamique.

---

# Challenge 12 : Tri à Bulles

Développez un programme qui implémente l'algorithme de tri à bulles.

**Principe :** Comparer et échanger les éléments adjacents si nécessaire

**Algorithme :**

1. Parcourir le tableau
2. Comparer chaque paire d'éléments adjacents
3. Échanger si mal ordonnés
4. Répéter jusqu'à ce qu'aucun échange ne soit nécessaire

**Objectif :** Implémenter un algorithme de tri avec boucles imbriquées.

---

# Challenge 13 : Table de Multiplication avec Somme Totale

Créez un programme qui affiche une table de multiplication et calcule la somme des produits.

**Exemple pour 4 :**

4 × 1 = 4
4 × 2 = 8
...
4 × 10 = 40
Somme totale: 220

**Objectif :** Combiner affichage et calcul cumulatif.

---

# Challenge 14 : Sélecteur de Jours de la Semaine

Développez un programme qui affiche les jours suivant un jour sélectionné.

**Fonctionnalité :**

1. Afficher la liste des jours (1=Lundi, 2=Mardi...)
2. L'utilisateur choisit un jour
3. Afficher ce jour et les jours suivants dans la semaine

**Gestion :** Traiter le passage de Dimanche à Lundi

**Objectif :** Utiliser des boucles avec logique circulaire (modulo).

---

# Challenge 15 : Calculateur de Factorielle (Boucle For)

Créez un programme qui calcule n! en utilisant une boucle `for`.

**Définition :** n! = 1 × 2 × 3 × ... × n

**Cas particulier :** 0! = 1

**Objectif :** Renforcer l'utilisation de la boucle `for` pour des calculs multiplicatifs.

---

# Conseils pour les Boucles Avancées

**Boucles imbriquées :**

- Bien identifier quel compteur contrôle quoi
- Attention aux conditions d'arrêt
- Optimiser quand possible (éviter les calculs redondants)

**Algorithmes efficaces :**

- Recherche dichotomique : O(log n) vs linéaire O(n)
- Optimisation des facteurs : tester jusqu'à √n
- Éviter les calculs inutiles dans les boucles

**Gestion des cas limites :**

- Nombres négatifs, zéro
- Tableaux vides
- Débordements de variables