Projet parallèle : Résolution du *Dam Break* 1D avec les équations de Saint-Venant (OpenMP & MPI)

Nom : _____ Prénom : _____

Contexte

Le problème de rupture de barrage (dam break) est un cas test classique pour la validation de schémas numériques sur les équations de Saint-Venant. Un code séquentiel en langage **C** résolvant ce problème via le schéma de **Rusanov** vous est fourni, ainsi qu'un script **Python** calculant la solution exacte.

L'objectif de ce projet est de **paralléliser** ce code de référence en utilisant **OpenMP** (partage de boucles) et **MPI** (distribution de domaine), puis de comparer les performances.

Objectifs pédagogiques

- Comprendre et exploiter les structures de données du code séquentiel.
- Implémenter une version parallèle en **OpenMP** du calcul du flux et de la mise à jour des variables.
- Implémenter une version **MPI** basée sur une décomposition 1D du domaine avec communications de halos.
- Comparer les performances (temps CPU, scalabilité forte/faible).
- Vérifier que la solution numérique reste correcte en version parallèle.

Travail demandé

1. Analyse du code fourni

- Étudier la structure du code `dam_break_rusanov.c`.
- Identifier les boucles parallélisables : calcul de la vitesse, du flux, mise à jour des conservées.
- Comprendre le fichier de sortie `output.dat` (structure : x, $h(x)$, $u(x)$).

2. Version OpenMP

- Ajouter les directives `#pragma omp parallel for` aux boucles appropriées.
- Éviter les data races sur les tableaux.
- Mesurer les gains de performance avec différents nombres de threads (`OMP_NUM_THREADS`).
- Tracer le speedup et l'efficacité.

3. Version MPI

- Diviser le domaine en blocs de taille locale (1D) pour chaque processus.
- Échanger les variables conservées aux interfaces avec `MPI_Sendrecv`.
- Synchroniser l'avancement temporel.
- À la fin, rassembler les résultats avec `MPI_Gather` (ou écriture parallèle).
- Vérifier la cohérence de la solution globale.

4. Validation et comparaison

- Comparer la solution parallèle à la version séquentielle et à la solution exacte.
- Comparer les performances :
  - **strong scaling** : temps en fonction du nombre de threads/processus à taille fixe
  - **weak scaling** : temps à domaine par processus constant
- Fournir des figures illustrant les performances.

Livrables attendus

- `dam_break_rusanov_openmp.c` : version avec OpenMP
- `dam_break_rusanov_mpi.c` : version avec MPI
- Figures de comparaison : solution exacte vs parallèle, performances
- **Rapport PDF** de 2–3 pages :
  - analyse des performances
  - choix techniques
  - validation des résultats