UNIVERSITÉ DE BISKRA **M1 Info IA-RTIC-GLSD**

Informatique Année **2024 - 2025**

Contrôle TP : AAP

Exercice 1. Questions de compréhension

1. Questions liées à la bibliothèque **MPI**

   (a) Soit le tableau global `tab=[6 5 8 1 4 2 9 3 7]` réparti sur 3 processeurs (`tab_local` sur chaque processeur), quel est le résultat res sur le processeur $pid = root$ de l'opération

   ```
   MPI_Reduce(tab_local,res,3,MPI_INT,MPI_SUM,root,MPI_COMM_WORLD)
   ```

   .....

   (b) Soit un tableau de 21 entiers initialisé sur le processeur `root` sachant qu'on dispose de 4 processeurs. Si on exécute une diffusion par

   ```
   MPI_Scatterv(tab,sendcounts, displs,MPI_INT,tab_recu,n,MPI_INT,root, MPI_COMM_WORLD)
   ```

   - Quels sont les processeurs qui exécutent cette commande ?

     .....

   - Quelle est la valeur de n sur le dernier processeur ?

     .....

   - Quel est le contenu de `sendcounts` et quel(s) processeur(s) le définit ?

     .....

     .....

Exercice 2. OpenMP- MPI

Pour calculer le nombre d'éléments pairs dans un vecteur v de taille n avec n divisible par le nombre de processus de l'exécution parallèle.

(a) Proposez une version parallèle en utilisant les directives OpenMP.

```
1  #include <stdio.h>
2  #include <omp.h>
3
4  int main(int argc, char * argv [])
5  {
6     .....
7     .....
8     .....
9     .....
10    .....
11    .....
12    .....
13    .....
14    .....
15    .....
16    .....
17    .....
18    .....
19    .....
20    .....
21    .....
22    .....
23 return 0;
24 }
```

(b) Proposez une version parallèle MPI en utilisant des communications collectives pour implémenter correctement les échanges entre processus, où le processus **root** est le seul à avoir alloué de la mémoire pour le vecteur v et à avoir initialisé v.

```
1  #include <stdio.h>
2  #include <mpi.h>
3
4  const int tag = 10;
5
6  int main(int argc, char **argv)
7  {
8    int pid, nprocs;
9    int n = atoi(argv[1]); // taille du vecteur v
10   .....
11   .....
12   .....
13   .....
14   .....
15   .....
16   .....
17   .....
18   .....
19   .....
20   .....
21   .....
22   .....
23   .....
24   .....
25   .....
26   .....
27   .....
28   .....
29   .....
30   .....
31   .....
32   .....
33   .....
34   .....
35   .....
36
37   MPI_Finalize();
38   return 0;
39 }
```

(c) Expliquez ce qu'il faudrait modifier si n devient indivisible par le nombre de processus de l'exécution parallèle.

.....

.....

.....

.....