Exercice1

Un hello world parallele

Q1: Je constate que:

Bonjour, je suis la tache n° 1 sur 16 taches
Bonjour, je suis la tache n° 7 sur 16 taches
Bonjour, je suis la tache n° 8 sur 16 taches
Bonjour, je suis la tache n° 10 sur 16 taches
Bonjour, je suis la tache n° 11 sur 16 taches
Bonjour, je suis la tache n° 12 sur 16 taches
Bonjour, je suis la tache n° 14 sur 16 taches
Bonjour, je suis la tache n° 15 sur 16 taches
Bonjour, je suis la tache n° 0 sur 16 taches
Bonjour, je suis la tache n° 2 sur 16 taches
Bonjour, je suis la tache n° 3 sur 16 taches
Bonjour, je suis la tache n° 4 sur 16 taches
Bonjour, je suis la tache n° 6 sur 16 taches
Bonjour, je suis la tache n° 9 sur 16 taches
Bonjour, je suis la tache n° 13 sur 16 taches
Bonjour, je suis la tache n° 5 sur 16 taches

Q2:Le nom du fichier txt est nommé d'après le numéro de thread et correspond au thread correspondant. Le contenu est le résultat de l'exécution du thread. De cette façon, nous savons ce que chaque thread exécute.

Exercice2

Envoi bloquant et non bloquant

Q1:La première version est plus sûr, car elle attendra que la transmission réussie avant d'exécuter myvar = myvar + 2;

Par contre, la deuxième version exécutera myvar = myvar + 2 immédiatement après l'exécution de MPI_ISend.En d'autre termes, il n'attendra pas la transmission réussie de MPI_ISend. Cela entraînera une concurrence pour la valeur de myvar.

Modification (la deuxième version):

```
MPI_Request Request;
MPI_Status Status;
myvar = 0;
for ( i = 1; i < numtasks , ++i ) {
task = i;
MPI_ISend (& myvar ,... ,... , task ,...);
MPI_Wait(&Request,&Status);
myvar = myvar + 2;
// Do some works
...
MPI_Wait (...);
}
```

Exercice3

Q1: Quand je executez mpiexec -np 10 ./Circulation_jeton.exe, je constate que:

Processeur n°0 sur 10 et jeton = 32
Processeur n°1 sur 10 et jeton = 33
Processeur n°2 sur 10 et jeton = 34
Processeur n°3 sur 10 et jeton = 35
Processeur n°4 sur 10 et jeton = 36
Processeur n°5 sur 10 et jeton = 37

Processeur n°6 sur 10 et jeton = 38
Processeur n°7 sur 10 et jeton = 39
Processeur n°8 sur 10 et jeton = 40
Processeur n°9 sur 10 et jeton = 41

Parce que je utilise envoi bloquant, le processus suivant ne s'affichera qu'une fois que le processus précédent aura envoyé et reçu le jeton avec succès.

Exercice4

L'information de la premiere version est:

La premiere version
nbSamples = 100000,nbp = 5
The mean pi = 3.14976
Le temps de l'execution de premiere version = 1.402ms

L'information de la premiere version est:

La deuxieme version
nbSamples = 100000,nbp = 5
The mean pi = 3.1346
Le temps de l'execution de premiere version = 0.108ms

Le temps d'execusion en utilisant un seul processus = 1.939ms.
Donc, pour la premiere version, l'accélération obtenue = 1.939/1.402=1.38 fois;
pour la deuxieme version, l'accélération obtenue = 1.939/0.108=17.95 fois.

Exercice5

Quand je execute mpiexec -np 16 ./Diffusion_hypercube.exe, il affiche que:

La processus n°0est source et jeton=18
le temps d'execution0.031215ms
La processus n°1est source et jeton=18
La processus n°2est source et jeton=18
La processus n°3est source et jeton=18
La processus n°4est source et jeton=18
La processus n°5est source et jeton=18
La processus n°6est source et jeton=18
La processus n°7est source et jeton=18
La processus n°9 et jeton=18
La processus n°10 et jeton=18
La processus n°11 et jeton=18
La processus n°12 et jeton=18
La processus n°13 et jeton=18
La processus n°14 et jeton=18
La processus n°15 et jeton=18
La processus n°8 et jeton=18