Exercise 11

Create a 2-dimensional cartesian grid topology to communicate between processes.

In each task is initialized a variable with the local rank of the cartesian communicator. The exercise is divided in three steps:

1) Compare the local rank with the global MPI_COMM_WORLD rank. Are they the same number?

2) Calculate on each task the average between its local rank and the local rank from each of its neighbours (north, east, south, west). Notice that in order to do this the cartesian communicator has to be periodic (the bottom rank is neighbour of the top)

3) Calculate the average of the local ranks on each row and column. Create a family of sub-cartesian communicators to allow the communications between rows and columns

---

HINTS:

C

MPI_DIMS_CREATE

```
int MPI_Dims_create(int nnodes, int ndims, int *dims)
```

MPI_CART_CREATE

```
int MPI_Cart_create(MPI_Comm comm_old, int ndims, int *dims, int *periods, int reorder, MPI_Comm *comm_cart)
```

MPI_CART_COORDS

```
int MPI_Cart_coords(MPI_Comm comm, int rank, int maxdims, int *coords)
```

MPI_CART_SHIFT

```
int MPI_Cart_shift(MPI_Comm comm, int direction, int disp, int *rank_source, int *rank_dest)
```

MPI_CART_SUB

```
int MPI_Cart_sub(MPI_Comm comm, int *remain_dims, MPI_Comm *newcomm)
```

MPI_COMM_FREE

```
int MPI_Comm_free(MPI_Comm *comm)
```

MPI_SENDRECV

```
int MPI_Sendrecv(void *sendbuf, int sendcount, MPI_Datatype sendtype, int dest, int sendtag, void *recvbuf, int recvcount, MPI_Datatype recvtype, int source, int recvtag, MPI_Comm comm, MPI_Status *status)
```

MPI_REDUCE

```
int MPI_Reduce(void* sendbuf, void* recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int root, MPI_Comm comm)
```

MPI_INIT

```
int MPI_Init(int *argc, char ***argv)
```

MPI_COMM_SIZE

```
int MPI_Comm_size(MPI_Comm comm, int *size)
```

MPI_COMM_RANK

```
int MPI_Comm_rank(MPI_Comm comm, int *rank)
```

MPI_FINALIZE

```
int MPI_Finalize(void)
```

FORTRAN

MPI_DIMS_CREATE

```
MPI_DIMS_CREATE(NNODES, NDIMS, DIMS, IERROR)
INTEGER NNODES,NDIMS,DIMS(*),IERROR
```

MPI_CART_CREATE

```
MPI_CART_CREATE(COMM_OLD, NDIMS, DIMS, PERIODS, REORDER, COMM_CART,
IERROR)
INTEGER COMM_OLD, NDIMS, DIMS(*), COMM_CART, IERROR
LOGICAL PERIODS(*), REORDER
```

MPI_CART_COORDS

```
MPI_CART_COORDS(COMM, RANK, MAXDIMS, COORDS, IERROR)
INTEGER COMM, RANK, MAXDIMS, COORDS(*), IERROR
```

MPI_CART_SHIFT

```
MPI_CART_SHIFT(COMM, DIRECTION, DISP, RANK_SOURCE, RANK_DEST, IERROR)
INTEGER COMM, DIRECTION, DISP, RANK_SOURCE, RANK_DEST, IERROR
```

MPI_CART_SUB

```
MPI_CART_SUB(COMM, REMAIN_DIMS, NEWCOMM, IERROR)
INTEGER COMM, NEWCOMM, IERROR
LOGICAL REMAIN_DIMS(*)
```

MPI_COMM_FREE

```
MPI_COMM_FREE(COMM, IERROR)
INTEGER COMM, IERROR
```

MPI_SENDRECV

```
MPI_SENDRECV(SENDBUF, SENDCOUNT, SENDTYPE, DEST, SENDTAG, RECVBUF,
RECVCOUNT, RECVTYPE, SOURCE, RECVTAG, COMM, STATUS, IERROR)
<type> SENDBUF(*), RECVBUF(*)
INTEGER SENDCOUNT, SENDTYPE, DEST, SENDTAG, RECVCOUNT, RECVTYPE,
SOURCE, RECV TAG, COMM, STATUS(MPI_STATUS_SIZE), IERROR
```

MPI_REDUCE

```
MPI_REDUCE(SENDBUF, RECVBUF, COUNT, DATATYPE, OP, ROOT, COMM, IERROR)
<type> SENDBUF(*), RECVBUF(*)
INTEGER COUNT, DATATYPE, OP, ROOT, COMM, IERROR
```

MPI_INIT

```
MPI_INIT(IERROR)
INTEGER IERROR
```

MPI_COMM_SIZE

```
MPI_COMM_SIZE(COMM, SIZE, IERROR)
INTEGER COMM, SIZE, IERRO
```

MPI_COMM_RANK

```
MPI_COMM_SIZE(COMM, SIZE, IERROR)
INTEGER COMM, SIZE, IERROR
```

MPI_FINALIZE

```
MPI_FINALIZE(IERROR)
INTEGER IERROR
```