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

int MPI_Finalize ()

int MPI_Comm_size (MPI_Comm comm, int *size)

int MPI_Comm_rank (MPI_Comm comm, int *rank)

int MPI_Send (void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)

int MPI_Recv (void *buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

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

int MPI_Sendrecv_replace (void *buff, int count, MPI_Datatype datatype, int dest, int sendtag, int source, int recvtag, MPI_Comm comm, MPI_Status *status)

---

int MPI_Barrier (MPI_Comm comm)

int MPI_Allreduce (void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, MPI_Comm comm)

1.) **int MPI_Bcast** (void *buf, int count, MPI_Datatype datatype, int source, MPI_Comm comm)
2.) **int MPI_Reduce** (void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int target, MPI_Comm comm)

1.) **int MPI_Scan** (void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, MPI_Comm comm)
2.) **int MPI_Exscan** (void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, MPI_Comm comm)

1.) **int MPI_Gather** (void *sendbuf, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int recvcount, MPI_Datatype recvdatatype, int target, MPI_Comm comm)
2.) **int MPI_Gatherv** (void *sendbuf, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int *recvcounts, int *displs, MPI_Datatype recvdatatype, int target, MPI_Comm comm)

1.) **int MPI_Allgather** (void *sendbuf, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int recvcount, MPI_Datatype recvdatatype, MPI_Comm comm)
2.) **int MPI_Allgatherv** (void *sendbuf, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int *recvcounts, int *displs, MPI_Datatype recvdatatype, MPI_Comm comm)

1.) **int MPI_Scatter** (void *s, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int recvcount, MPI_Datatype recvdatatype, int source, MPI_Comm comm)
2.) **int MPI_Scatterv** (void *sendbuf, int *sendcounts, int *displs, MPI_Datatype senddatatype, void *recvbuf, int recvcount, MPI_Datatype recvdatatype, int source, MPI_Comm comm)

int MPI_Alltoall (void *sendbuf, int sendcount, MPI_Datatype senddatatype, void *recvbuf, int recvcount, MPI_Datatype recvdatatype, MPI_Comm comm)

int MPI_Comm_split (MPI_Comm comm, int color, int key, MPI_Comm *newcomm)

---

int MPI_Init_thread (int *argc, char ***argv, int required, int *provided)

-> Required is the aimed support level
-> Provided is the support level provided by the MPI implementation

MPI_THREAD_SINGLE: only one thread will execute (the same as initializing the environment with MPI_Init ())

MPI_THREAD_FUNNELED: only the master thread can make MPI calls

MPI_THREAD_SERIALIZED: all threads can make MPI calls, but only one thread at a time can be in such state

MPI_THREAD_MULTIPLE: all threads can make simultaneous MPI calls without any constraints