

## Signature Posix

```
//creazione thread
int pthread_create(pthread_t * thread,
                  const pthread_attr_t * attr,
                  void * (*start_routine)(void *),
                  void *arg);

// join
int pthread_join( pthread_t thread,void** value_ptr );

//mutex
int pthread_mutex_init(pthread_mutex_t *mutex,
                      pthread_mutex_attr *attr);
int pthread_mutex_lock(pthread_mutex_t* mutex );
int pthread_mutex_unlock(pthread_mutex_t* mutex );
int pthread_mutex_destroy(pthread_mutex_t *mutex);

//condition
int pthread_cond_init( pthread_cond_t *cond,
                      pthread_condattr_t *cond_attr )
int pthread_cond_destroy( pthread_cond_t *cond )
pthread_cond_wait(&a_c_v,&a_mutex);
pthread_cond_signal (pthread_cond_t *cond)
pthread_cond_broadcast (pthread_cond_t *cond)
```

## Signature MPI

```
MPI_Init (&argc,&argv);
MPI_Comm_size (comm,&size);
MPI_Comm_rank (comm,&rank);
MPI_Finalize ();
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 );
MPI_Get_count(MPI_Status *status, MPI_Datatype datatype, int *count
);
int MPI_Isend( void *buf, int count, MPI_Datatype datatype, int dest,
int tag, MPI_Comm comm, MPI_Request *request );
int MPI_Wait (MPI_Request *request, MPI_Status *status);
int MPI_Test (MPI_Request *request, int *flag, MPI_Status *status)
int MPI_Type_vector(int block_count, int block_length, int stride,
MPI_Datatype old_datatype, MPI_Datatype* new_datatype);
int MPI_Type_commit(MPI_Datatype* datatype);
int MPI_Type_free(MPI_Datatype* datatype);
```