```
#include <pthread.h>

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

DESCRIPTION

The pthread_create() function is used to create a new thread, with attributes specified by attr, within a process. If attr is NULL, the default attributes are used. If the attributes specified by attr are modified later, the thread's attributes are not affected. Upon successful completion, pthread_create() stores the ID of the created thread in the location referenced by thread. The thread is created executing start_routine with arg as its sole argument. If the start_routine returns, the effect is as if there was an implicit call to pthread_exit() using the return value of start_routine as the exit status. Note that the thread in which main() was originally invoked differs from this. When it returns from main(), the effect is as if there was an implicit call to exit() using the return value of main() as the exit status.

returns 0 if successful

```
#include <pthread.h>

int pthread_join(pthread_t thread, void **value_ptr);
```

DESCRIPTION

The pthread_join() function suspends execution of the calling thread until the target thread terminates, unless the target thread has already terminated. On return from a successful pthread_join() call with a non-NULL value_ptr argument, the value passed to pthread_exit() by the terminating thread is made available in the location referenced by value_ptr. When a pthread_join() returns successfully, the target thread has been terminated. The results of multiple simultaneous calls to pthread_join() specifying the same target thread are undefined. If the thread calling pthread_join() is canceled, then the target thread will not be detached.
It is unspecified whether a thread that has exited but remains unjoined counts against _POSIX_THREAD_THREADS_MAX.

RETURN VALUE

If successful, the pthread_join() function returns zero. Otherwise, an error number is returned to indicate the error.

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *print_message_function( void *ptr );

main()
{
     pthread_t thread1, thread2;
     char *message1 = "Thread 1";
     char *message2 = "Thread 2";
     int  iret1, iret2;

    /* Create independent threads each of which will execute function */

     iret1 = pthread_create( &thread1, NULL, print_message_function, (void*)
message1);
     iret2 = pthread_create( &thread2, NULL, print_message_function, (void*)
message2);

     /* Wait till threads are complete before main continues. Unless we  */
     /* wait we run the risk of executing an exit which will terminate   */
     /* the process and all threads before the threads have completed.   */

     pthread_join( thread1, NULL);
     pthread_join( thread2, NULL);

     printf("Thread 1 returns: %d\n",iret1);
     printf("Thread 2 returns: %d\n",iret2);
     exit(0);
}

void *print_message_function( void *ptr )
{
     char *message;
     message = (char *) ptr;
```

```
     printf("%s \n", message);
}
```