EXP 30: Write C programs to demonstrate the following thread related concepts.
(i) create (ii) join (iii) equal (iv) exit

Create

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

// Function to be executed by the thread
void* thread_function(void* arg) {
    printf("Thread is executing\n");
    return NULL;
}

int main() {
    pthread_t thread_id;

    // Create the thread
    if (pthread_create(&thread_id, NULL, thread_function, NULL) != 0) {
        perror("Thread creation failed");
        return 1;
    }

    // Wait for the thread to complete
    pthread_join(thread_id, NULL);

    printf("Main thread exiting\n");
    return 0;
}
```

Join

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

// Function to be executed by the thread
void* thread_function(void* arg) {
    printf("Thread is executing\n");
    return NULL;
}

int main() {
    pthread_t thread_id;

    // Create the thread
    if (pthread_create(&thread_id, NULL, thread_function, NULL) != 0) {
        perror("Thread creation failed");
        return 1;
    }

    // Main thread waits for the created thread to finish
    pthread_join(thread_id, NULL);

    printf("Main thread joined the created thread\n");
    return 0;
}
```

Equal

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

```
// Function to be executed by the thread
void* thread_function(void* arg) {
    pthread_t current_thread = pthread_self();
    printf("Thread ID: %ld\n", (long)current_thread);
    return NULL;
}

int main() {
    pthread_t thread_id1, thread_id2;

    // Create two threads
    pthread_create(&thread_id1, NULL, thread_function, NULL);
    pthread_create(&thread_id2, NULL, thread_function, NULL);

    // Wait for both threads to finish
    pthread_join(thread_id1, NULL);
    pthread_join(thread_id2, NULL);

    // Check if thread IDs are equal
    if (pthread_equal(thread_id1, thread_id2)) {
        printf("The threads are equal\n");
    } else {
        printf("The threads are not equal\n");
    }

    return 0;
}
```

Exit

```
#include <stdio.h>

#include <pthread.h>


// Function to be executed by the thread

void* thread_function(void* arg) {

   printf("Thread is executing\n");


   // Exit from the thread

   pthread_exit(NULL);

}


int main() {

   pthread_t thread_id;


   // Create the thread

   if (pthread_create(&thread_id, NULL, thread_function, NULL) != 0) {

      perror("Thread creation failed");

      return 1;

   }


   // Main thread waits for the created thread to finish

   pthread_join(thread_id, NULL);
```

```
    printf("Main thread exiting\n");

    return 0;

}
```

Sample Output

For Create,

```
Thread is executing
Main thread exiting

--------------------------------
Process exited after 2.751 seconds with return value 0
Press any key to continue . . .
```

For Join,

```
Thread is executing
Main thread joined the created thread

--------------------------------
Process exited after 2.911 seconds with return value 0
Press any key to continue . . .
```

For Equal,

```
Thread ID: 1
Thread ID: 2
The threads are not equal

--------------------------------
Process exited after 2.651 seconds with return value 0
Press any key to continue . . .
```

For Exit,

```
Thread is executing
Main thread exiting

--------------------------------
Process exited after 2.583 seconds with return value 0
Press any key to continue . . .
```