1. *Create a thread in either a C or a Java program and run it. The thread should read a keyboard value "x" and exit from the code, or it can also use an "ESC" to exit. That is, the thread should run until the keyboard exit character is entered. Print your source code and output to a single file and generate a pdf and submit it in Blackboard. Put your compile and run options if any in the code as comments.*

   a. **thread_create.c**

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>

// Compile: gcc -o thread_create thread_create.c -lpthread

void *read_keys(void * ptr);

int main()
{
    pthread_t thread1;
    int iret1 = pthread_create(&thread1, NULL, read_keys, NULL);
    if(0 != iret1)
    {
        printf("Error - pthread_create() return code: %d\n", iret1);
        return (-1);
    }
    pthread_join(thread1, NULL);
    printf("All threads complete\n");
    return 0;
}

void *read_keys(void *ptr)
{
    printf("Enter x or X or ESC to exit\n");
    int cur_char;
    do
    {
        cur_char = getchar();
    }
    while(('x' != (char)(cur_char)) && ('X' != (char)(cur_char)) && (27 != cur_char));
    return 0;
}
```

   b. **output**

      i. **Exit using 'x':**

```
[mjs@localhost Homework4]$ ./thread_create
Enter x or X or ESC to exit
i
P
4
x
All threads complete
```

ii. **Exit using 'X':**

```
[mjs@localhost Homework4]$ ./thread_create
Enter x or X or ESC to exit
8
d
S
X
All threads complete
```

iii. **Exit using ESC:**

```
[mjs@localhost Homework4]$ ./thread_create
Enter x or X or ESC to exit
l
F
3
*
^[                    (ESC)
All threads complete
```

2. *Using the above thread program, now create 20 threads in your program and run them. No need to wait for the keyboard in each thread, just print "Hello" and its thread id in each thread on a single line and exit. Append your source code and output to the same file as listed above. Add enough comments in your program to make it readable.*

a. **multi_thread_create.c**

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>

// Compile: gcc -o multi_thread_create multi_thread_create.c -lpthread

void *print_hello(void * ptr);

int main()
{
    printf("Main thread: %u\n", pthread_self());
    pthread_t tID[20];

    // Create 20 threads
    unsigned int i = 0;
    for(i = 0; i < 20; ++i)
    {
        // Create thread
        int iret1 = pthread_create(&tID[i], NULL, print_hello, NULL);
        if(0 != iret1)
        {
            printf("Error - pthread_create() return code: %d\n", iret1);
            return (-1);
```

```
            }
        }

        // Wait for all threads to complete
        for(i = 0; i < 20; ++i)
        {
            pthread_join(tID[i], NULL);
        }
        printf("All threads complete\n");
        return 0;
}

void *print_hello(void *ptr)
{
        printf("Hello from thread %u\n", pthread_self());
        return 0;
}
```

b. output

```
[mjs@localhost Homework4]$ ./multi_thread_create
Main thread: 2243671872
Hello from thread 2185111296
Hello from thread 2193504000
Hello from thread 2176718592
Hello from thread 2168325888
Hello from thread 2201896704
Hello from thread 2143147776
Hello from thread 2227074816
Hello from thread 2159933184
Hello from thread 2235467520
Hello from thread 2151540480
Hello from thread 2218682112
Hello from thread 2210289408
Hello from thread 2134755072
Hello from thread 2126362368
Hello from thread 2092791552
Hello from thread 2109576960
Hello from thread 2117969664
Hello from thread 2076006144
Hello from thread 2101184256
Hello from thread 2084398848
All threads complete
```