**Sample Process Scheduling Program using single thread:**

#include <iostream>

#include <cstdlib>

#include <unistd.h>

#include <pthread.h>

using namespace std;

void \*myThreadFun(void \*vargp)

{

sleep(1);

cout << "Printing Hello from Thread" << endl;

return NULL;

}

int main()

{

pthread\_t thread\_id;

cout << "Before Thread" << endl;

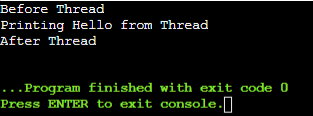
pthread\_create(&thread\_id, NULL, myThreadFun, NULL);

pthread\_join(thread\_id, NULL);

cout << "After Thread" << endl;

exit(0);

}



**Sample Process Scheduling Program Using multiple threads:**

#include <iostream>

#include <cstdlib>

#include <pthread.h>

using namespace std;

#define NUM\_THREADS 5

void \*PrintHello(void \*threadid) {

long tid;

tid = (long)threadid;

cout << "Hello World! Thread ID, " << tid << endl;

cout << "Before termination of Thread ID, " << tid << endl;

pthread\_exit(NULL);

cout << "After termination of Thread ID, " << tid << endl;

}

int main () {

pthread\_t threads[NUM\_THREADS];

int rc;

int i;

for(i = 0; i < NUM\_THREADS; i++) {

cout << "main() : creating thread, " << i << endl;

rc = pthread\_create(&threads[i], NULL, PrintHello, (void \*) i);

if (rc) {

cout << "Error: unable to create thread, " << rc << endl;

exit(-1);

}

}

cout << "Before termination of Main Thread" << endl;

pthread\_exit(NULL);

cout << "After termination of Main Thread" << endl;

}

