# Semaphore in POSIX C

Note: Please use text highlight color to highlight your answer.

1. Read the below source code, semaphore\_thread.c and compile it. Note: please use the appropriate library in compiling.

// semaphore\_thread.c

#include <pthread.h>

#include <semaphore.h>

#include <unistd.h>

sem\_t mylock;

void\* thread( void\* arg)

{

sem\_wait(&mylock);

printf("\nEntering the critical section..\n");

sleep(4);

printf("\nExiting the critical section...\n");

sem\_post(&mylock);

}

int main()

{

sem\_init(&mylock, 0, 1);

pthread\_t t1,t2;

pthread\_create(&t1,NULL,thread,NULL);

sleep(2);

pthread\_create(&t2,NULL,thread,NULL);

pthread\_join(t1,NULL);

pthread\_join(t2,NULL);

sem\_destroy(&mylock);

return 0;

}

1. Run the program and describe what you have observed? Pastes the screenshot.
2. Find the location of the file semaphore.h using either find or locate. Paste the screenshot of the results.
3. What is the function of sem\_wait()? Explain it including the purpose, parameters and return values in your own words.
4. What is the function of sem\_post()? Explain it including the purpose, parameters and return values in your own words.
5. What is the function of sem\_init()? Explain it including the purpose, parameters and return values in your own words.
6. What is the function of sem\_destroy()? Explain it including the purpose, parameters and return values in your own words.
7. What is the function of pthread\_join()? Explain its functions in your own words.
8. Read the source code and consider if this binary semaphore in this code is equivalent to the mutex? Why do we use semaphore rather than the mutex?
9. Write a program named ABCD\_Writer.c based on the source code semaphore\_thread.c.
10. The program should be able to fill an array of 100,000 characters via four threads. Threads one, two, three, four fill up the array with the letters ‘A’,’B’,’C’,’D’ respectively.
11. The array should be shared between the four threads. Each thread has to compete with each other for modifying the array. Use a binary semaphore rather than mutex lock to synchronize their access to the critical section.
12. Each thread is allowed to fill up one letter only when it enters the critical section. That is, if the thread wants to write one more letter, it has to re-enter the critical section and it is only allowed to write its own letter (e.g. thread three is only allowed to write a letter ‘C’).
13. The four threads should keep the counter of how many characters they have written. Print out the corresponding counters.
14. It is only after the four threads finish their job that the parent thread can leave. The parent should count how many ‘As’,’Bs’,’Cs’,’Ds’ in the array independent from each thread’s counter. Print out the statistics to see if they match thread’s own counters.

Submission Requirement: (1) the ABCD\_Writer.c file (pasted or attached) and (2) the printout screenshot to show the thread’s counters and the main thread’s statistics.