**Zach Vance**

**Lab 4**

**A screenshot of a computer program

Description automatically generated**

**A number grid with numbers

Description automatically generated with medium confidence**

**1. Is there any shared variable in the programs? If any, list all shared variables? (Note that in thread synchronization, a shared variable refers to a variable that is shared by multiple threads.)**

Yes, the variable int n is a shared variable in both task1.1 and task1.2.

**2. What are the output of lab4 task1.1.c? Are they the same in different rounds of execution? Why or why not?** The output of lab4task1.1.c has the same output during every round of execution. Every round of execution this program outputs 10 10 10 10 10 10 10 10 10 10. In this case the producer increments n to 10 before the consumer thread begins printing the value of n.

**3. What are the output of lab4 task1.2.c? Are they the same in different rounds of execution? Simply speaking, the second program involves much heavier workload (i.e., much larger number of iterations) than the first program. To reduce the IO and save our time, we print the value of n with a large sampling step. How may the workload have an influence on the output? Justify your answer.**

Each round of execution for lab4task1.2.c has a different output. The first 5 or 6 numbers printed are different and the last 4-5 numbers seem to be the same but this is likely due to the heavy workload. The larger workload influences the output because it affects the timing of thread operations making the context switching unpredictable. The large sampling step is used to reduce i/o operations but the workload still influenced the output because of lack of synchronization.

A screenshot of a computer screen

Description automatically generated

**4. In producer() function, which statements are entry section, critical section, and exit section?**

void \*producer(void \*param)

{

int i;

for (i=0; i<10; i++)

{

sem\_wait(&psem); //entry section

n++; //critical section

sem\_post(&csem); //exit section

}

pthread\_exit(0);

}

**5. How many context switches between the producer thread and the consumer thread in the execution of the program? Justify your answer.**Twenty. There two context switches in each iteration of the for loop – one in the consumer function and one in the producer function.

**6. What is the first value printed? Will it be changed if you exchange the two pthread create statements? Explain why?**

The first value printed is ‘1’ regardless of if the ‘pthread-create’ statements are exchanged. Thread creation order does not determine the initial value of n because csem (consumer semaphore) is set to 0 which tells the consumer to wait, regardless of whether it was created first or not. So the producer thread is executed first and increments n to 1 which is then printed by the consumer thread.

**A close-up of a number

Description automatically generated  
  
7. What will be the first value printed if you exchange the initialization values for psem and csem? Explain why?**

**A number and symbols on a black background

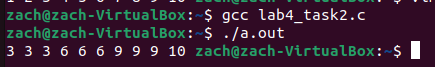
Description automatically generated**

The psem (producer semaphore) is set to 0, which indicates for the producer to wait and the csem (consumer semaphore) is set to 1, which tells the consumer to begin. Since the consumer begins immediately, the values start being printed before n is incremented so 0 is printed.

**8. In either cases, are values of n printed consecutively? Justify your answer.**

Yes in both cases the values of n are printed consecutively. In both cases the producers will increase n in increments of 1 and immediately after the consumer will print the value of n.

**9. If psem is set to 3, and csem is set to 0, do you observe any pattern in the output? If yes, explain why?**

****

The observed pattern is 3 3 3 6 6 6 9 9 9 10. This is caused by psem being set to 3 which causes the producer to execute 3 times before the consumer executes causing n to be incremented to a value of 3. The consumer thread then executes and prints the current value of n three times. The producer thread then executes once more incrementing the value of n to 6 after which the consumer thread is executed printing 6 three times. This process is completed once more with the value of 9. 10 is the final value because it is the upper limit included in both loops.

**10 What is median elapse time under the first setting? What is the median elapse time under the second setting? Which one is smaller? Explain why,**

The median elapsed time under the first setting is **6.627817.** The median elapsed time for the second setting is **0.410987**.

The second setting has psem set to 1e3 which allows for the producer thread to run for larger times without having to switch back and forth. In this case the producer will increment n 1000 times before giving access to the consumer. This greatly reduces the frequency of thread switching and results in a much quicker execution compared to setting 1 in which the consumer and producer switch back and forth constantly.

**A screenshot of a computer program

Description automatically generated   
A screenshot of a computer program

Description automatically generated**