National University of Computer and Emerging Sciences

Laboratory Manual

for

Operating Systems Lab

(CL-220)

Course Instructor	Ms. Namra Absar
Lab Instructor(s)	Rasaal Ahmad
Section	5B
Semester	Fall 2023

Department of Computer Science

FAST-NU, Lahore, Pakistan

Objectives

· In this lab, students will practice thread and process synchronization using semaphores

Lab Questions

Question 1:

Implement a program in C/C++ which is passed as parameter an integer number n. The program will create n threads. There will be a variable x shared among all these threads. These threads will do the following:

1. Increment the value of x by 1.
2. After incrementing the value of x by 1, print the value of x.

Before creating n threads, the program will also create a special thread that will not do the above task. Instead, when the n threads have finished their execution, it will print ("All threads have finished their execution"). Make use of semaphores to synchronize the access to x. Also use a semaphore to prevent the special thread from printing the required statement until the n threads have finished. The values of x will be printed in order. Do not use semaphores in the main function; use them in the threads created.

Calculating the maximum number from a list of integers by exploiting parallelism:

Suppose we have an array of integers: 1, 10, 9, 100, 23, 4, 11, 12, 3, and 1. The size of this array is 10. Now if we want to calculate the average, we can simply do:

Average = sum of elements/ total elements.

Now, if we have a large number of elements, then it will take a lot of time to calculate the average. We can utilize parallelism by dividing the task of finding the sum among different processes. Suppose a process p1 finds the sum of the first 4 elements:

Sum1=1+10+9+100= 120

Count=4

And another process p2 calculates the sums of the remaining 6 elements:

Sum2=23+4+11+12+3=53

Count=6

The average calculated as:

Total Sum = Sum1+Sum2= 120+53= 173

Total Count= Count1 + Count2= 4+6=10

Average= Total Sum/Total Count= 173/10= 17.3

Question 2:

Suppose there are two processes. Each process reads a different file having a list of integers. Both processes read the integers, calculate their sum, and send the sum and the count of integers to a server process via shared memory. The server then finds the total average by following formulae:

Total Sum= sum of integers sent by p1 + sum of integers sent by p2

Total Count= count sent by p1 + count sent by p2

Average= Total Sum/ Total Count

After calculating the average, the server sends the average to both processes. Both processes then print the sum on their respective terminal. (You need to synchronize the processes using semaphores on shared memory)

Shared Memory Portions Required:

1. The will be a single shared memory portion on which both processes will place their (sum, count) pair. One process will put the pair on 0th index and the other will put the pair on 1st index.
2. A shared memory portion for current available index number where a process can put the pair.