**Calculating the average of 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 first 4 elements:

Sum1=1+10+9+100= 120

Count=4

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

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

Count=6

Then 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**

Suppose there are two child 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 parent 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 the 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 of 8 bytes on which both processes will place their (sum, count) pair and the server will read from this shared memory segment. Because there is only one location you would have to synchronize the three processes using semaphores.
2. A shared memory portion for placing the average by server. (Only one portion required which will be shared by both worker processes)
3. Shared memory portions for the required semaphores and mutexes.