Q1:- Semaphore is one of the concurrency mechanisms available. Find out about more concurrency mechanisms. How do these mechanisms protect critical sections? Compare their implementations with *wait ()* and *signal ()* operations of semaphores.

***Explanation:***

**Concurrency** is the computation of processes within a time frame to give an impression of simultaneous execution. Here, it's important to note that this is not the same thing as actually running simultaneously. True parallelism would allow for the simultaneous execution of the processes, but concurrent processes do not exhibit parallelism.

**Semaphores** are integer variables that are used to solve the critical section problem by using two atomic operations, wait and signal that are used for process synchronization.

The definitions of wait and signal are as follows −

**Wait**

The wait operation decrements the value of its argument S, if it is positive. If S is negative or zero, then no operation is performed.

wait(S)

{

while (S<=0);

S--;

}

**Signal**

The signal operation increments the value of its argument S.

signal(S)

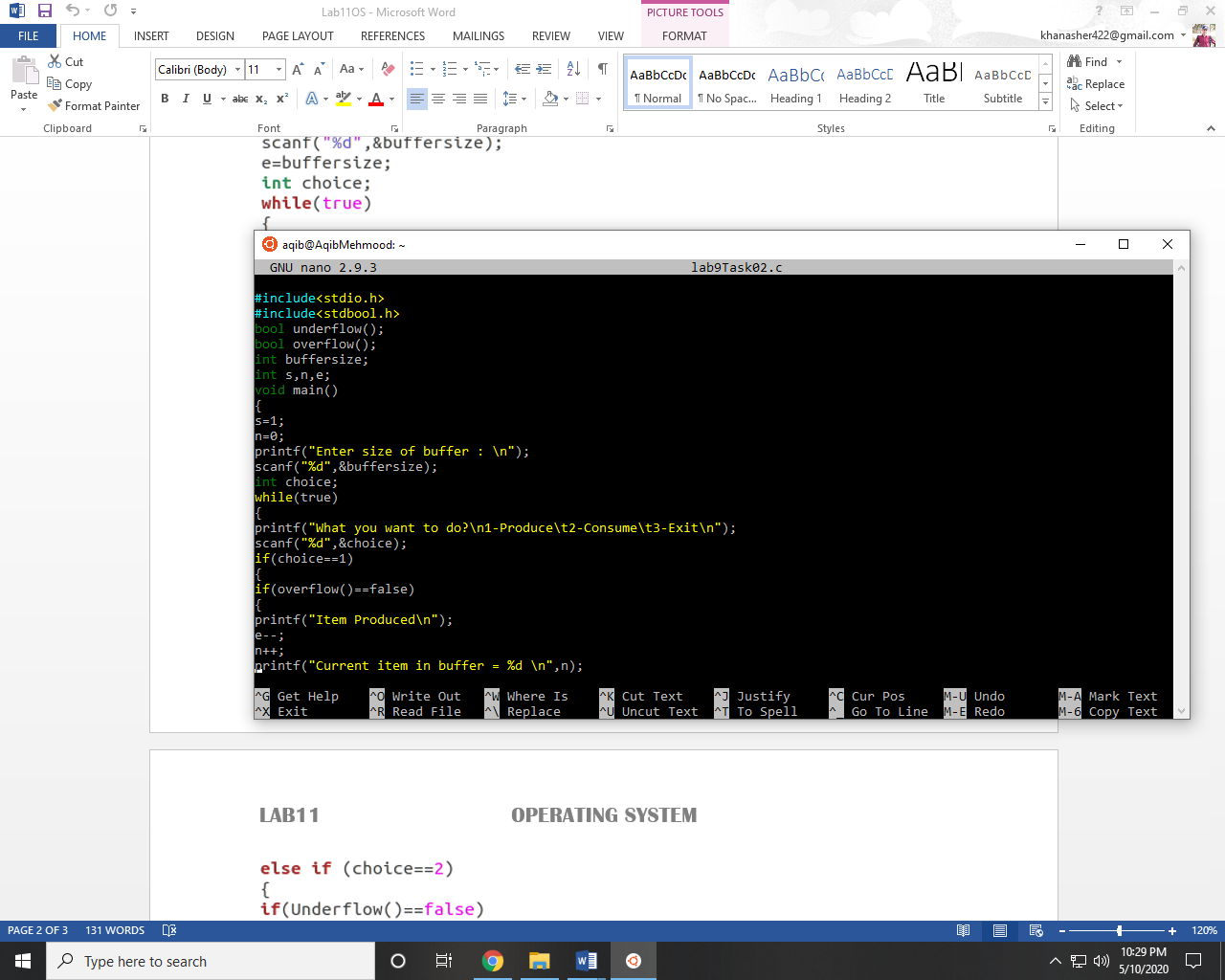
{

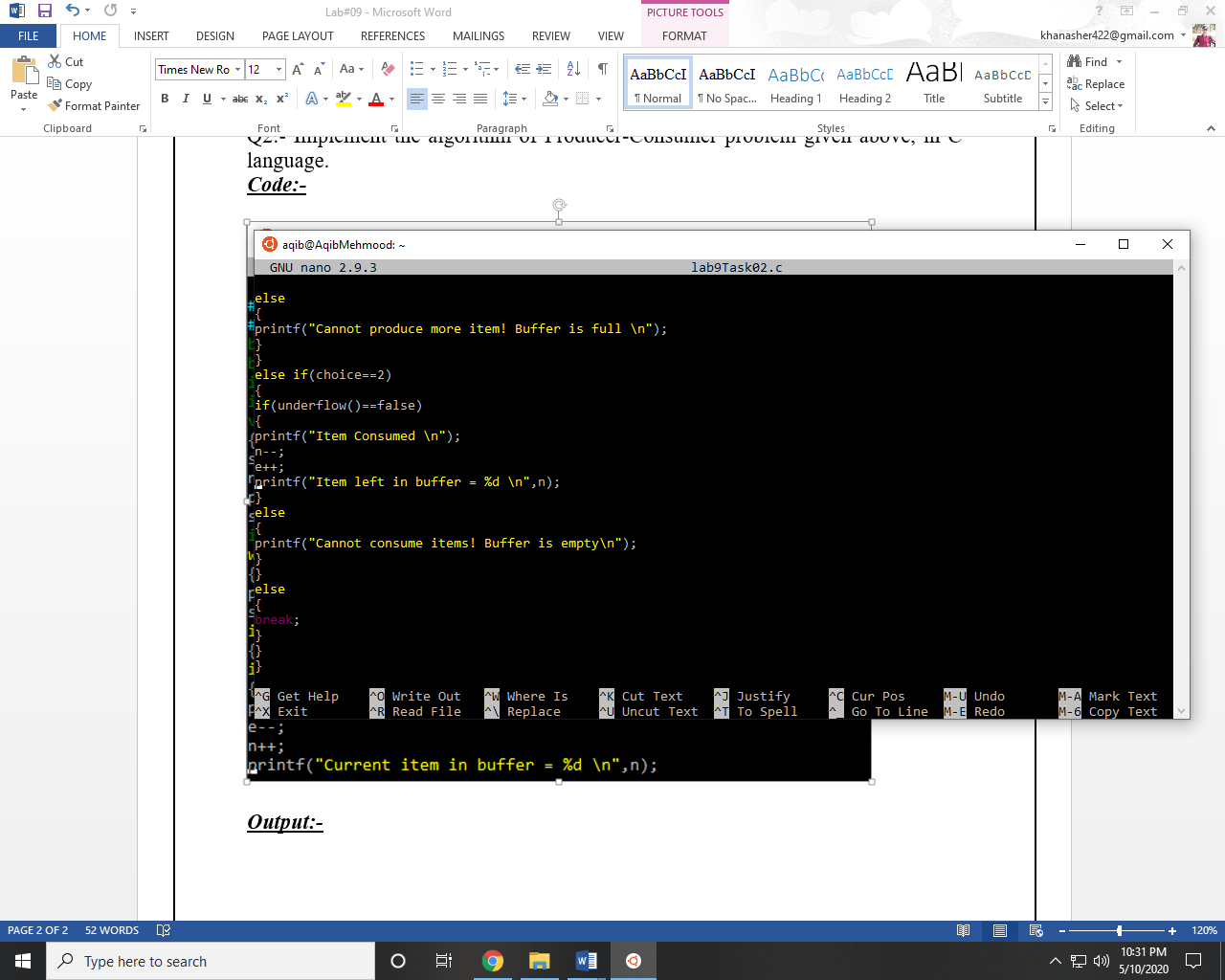
S++;

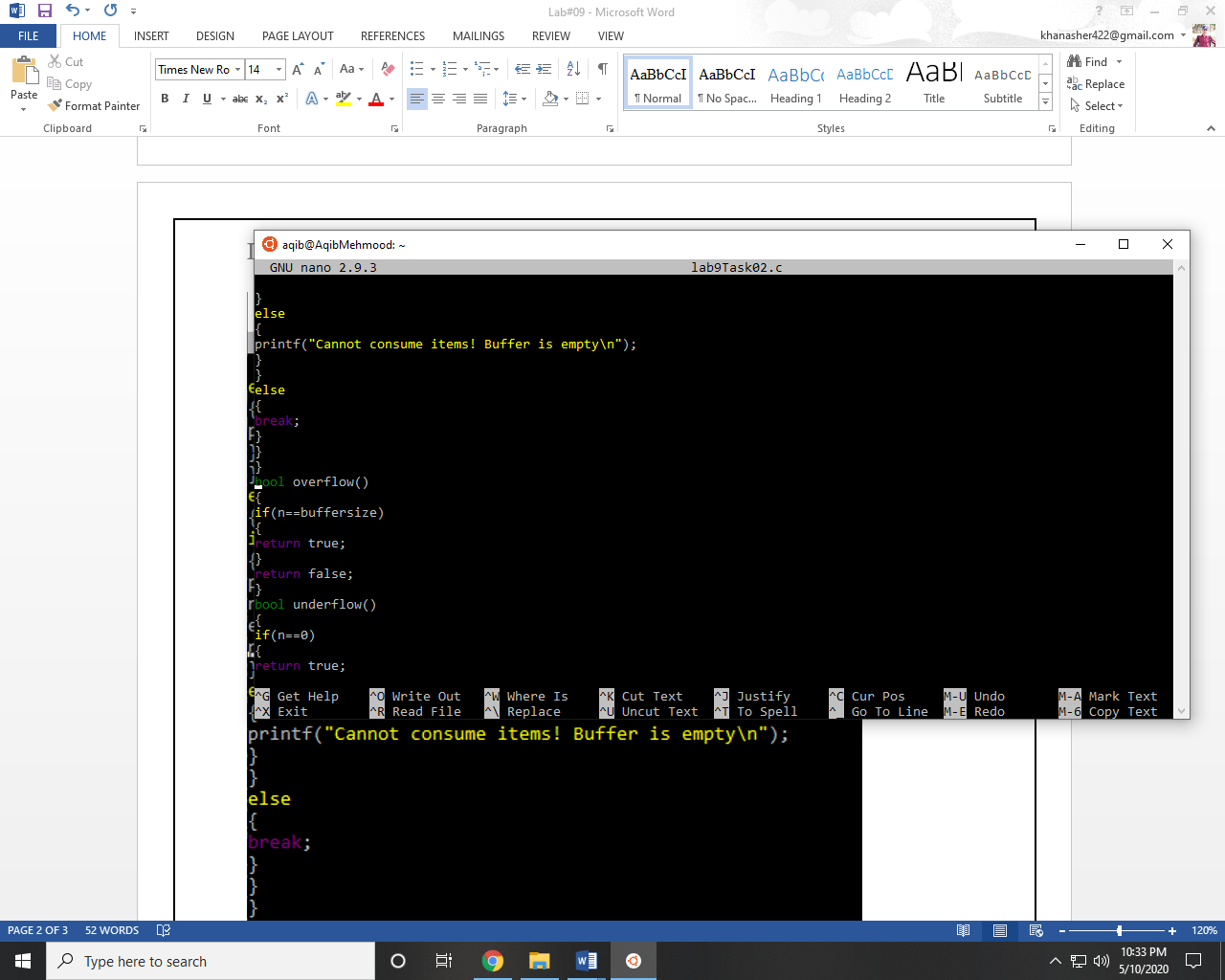
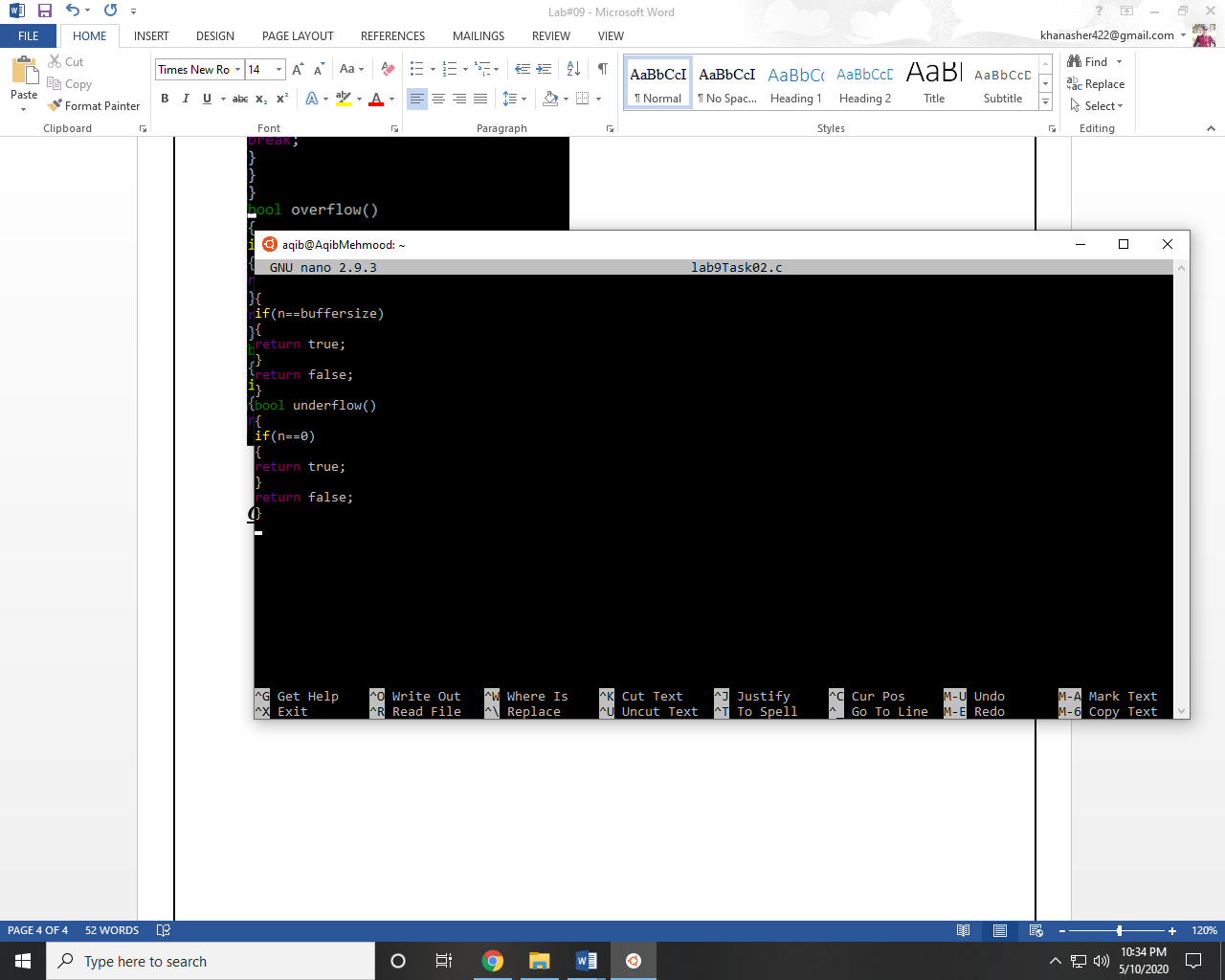
}

Q2:- Implement the algorithm of Producer-Consumer problem given above, in C language.

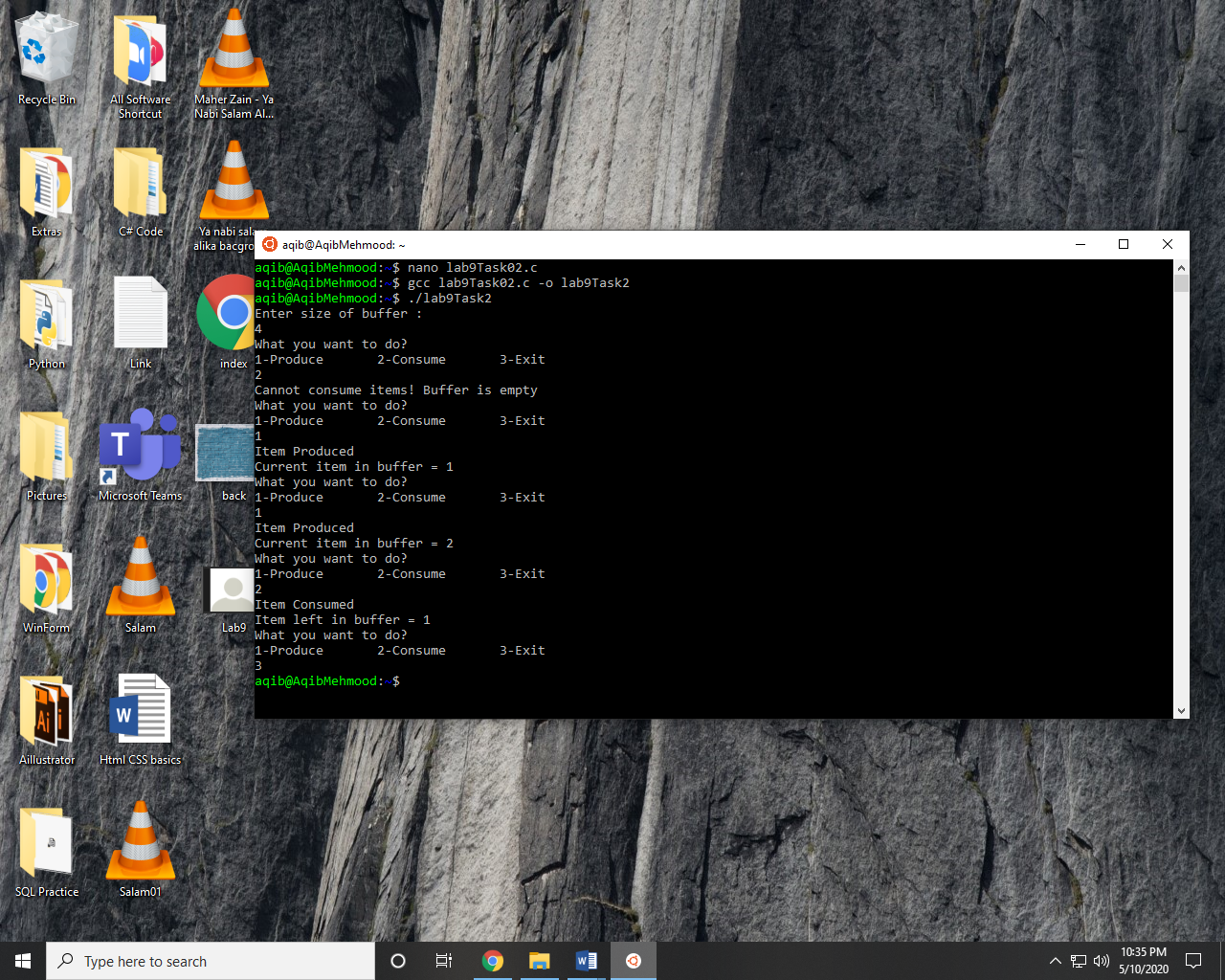
***Code:-***





***Output:-***



~~~~~~\*\*/**THE END**/\*\*~~~~~~