Lab 6 Tasks

Q1. Design a tic tac toe game using an unnamed pipe and forks, where 1 process is the designated “server”. This process will handle the communication with the different clients and include game logic to judge the result of the game. 2 other processes will communicate with this server, with each message containing the “cell number” of the desired move. You need to validate whether the cell number is already occupied or not and whether it is in bounds (i.e. if it is a valid move). Note that tic tac toe game is typically played on 3x3 board, so cells could be numbered 1-9.

Q2. Simulate a restaurant table reservation system, assuming that tables are numbered from 1 till n. When a customer comes in, they speak to the receptionist and inform them of a specific desired table (for example table #3). The receptionist must communicate with the restaurant attendant to verify whether the table is available or occupied. According to the current state the customer is either seated at the table or told to wait until the table is free again. Map this scenario using named pipes, where 1 process acts as a receptionist, 1 process will act as the attendant and there will be any number of processes acting as the customers walking into the restaurant.

Q3. Write two programs that would implement the concept of shared memory, the requirements are as follows:

a. The first program would create a shared memory and put a number in it.
b. The second program would store the number (which would come as string) in an integer variable and then writes in the memory “ready”.
c. The term “ready” is then picked up by the first program, it prints this value onto the screen and puts ‘*’ in the memory.
d. The second program when read ‘*’ will put the table of the number which it stored in part b from 1 – 10. Such that:
    i. Assume num is the variable it stored the number came from program 1.
    ii. It will initiate an iterator say int i and assign 1 in it.
    iii. Whenever it sees ‘*’ in the shared memory it will put the value calculated from the equation: i*n in the shared memory. Which then picked up by the first program and again it will read the value, print it on the screen and put ‘*’ again.
    iv. This cycle continues till i > 10