In this homework, we simulate a simple Airline Reservation System which is involve accessing shared resources. Two travel agencies are used, which will be assigned to two different threads. Shared resource is a 2*50 global matrix, that is filled with all zeros initially, this shared resource represents seats in the plane and is used by travel agencies. Since there are two threads, that use same resource, shared resource should be prevented from simultaneous usages or race conditions. Therefore, strict alternation method is used for overcome this situation. In this method, while one thread (agency) is in busy-waiting, other one does its tasks (means it is in critical region), that based on shared resource, when thread, that is in critical region, finishes its tasks, turn is changed and, thread, which was in critical region, is now in busy-waiting and the other thread is enter critical region. This method continues until plane fulls.

Flow of the program:

- Firstly, two threads are created with connecting them different routines, that threads will execute, after creating them successfully, pthread_join comment is used for synchronization (main thread should wait other threads).
- For each thread there are functions called reserverFirstAgency and reserverSecondAgency. (these are the routines that threads do). These two function together form strict alternation method that explained above. And they run concurrently.
- Firstly, they run while plane is not full and "contin" variable is true. (this variable is explained very detail in program).
- In non critical region threads create a random seat number.
- Then one of the thread is in busy-waiting other one firstly converts seat number as matrix coordinates via "converter" function.
- checks if the seat is reserved or not, if it is not reserved it reserves that seat and marks it (if thread1 reserves, marks it as 1; if thread2 reserves, marks it as 2) and decrement the capacity.
- Then, there is an if statement that prevent a situation explained as following; if one thread takes last seat while other thread is in busy waiting, after turn changes, thread, which was in the busy-waiting before, tries to find empty seat, which does not exist anymore.
- If plane is full (there isn't any seat) and "contin" is true (as I mentioned before, it is explained detail in program) is entered to inside that if statement, inside the statement appropriate messages is shown and "contin" is assigned as false, which means that their tasks finish and functions will not continue anymore.
- If plane is not full, at the end of the main while loop "turn" is changed, which means that, thread, which is in critical region, will get in to busy-waiting and vice versa for other thread.
- When tasks are completed by threads, we return to main function and, the last state of the plane is shown by "showPlane" function.
- Then the program is ended up with return 0;