/* Automated Biryani Serving */

- There are **n** robot chefs, **m** serving tables and **k** students. Each of them is a thread. There is a **global mutex lock** on total number of students and **one mutex lock for each table**.
- n (**No. Of Chefs**), m (**No. Of tables**) and k (**No. Of Students**)are user inputs and rest of the variables are randomly generated using `***rand()***` function in C with seed current time.

Robot Chef's :
Implemented using struct `ROBOT` and for each of the **n** chefs, a new thread is created.

- Preparation time of robot is stimulated by `sleep(<random int between 2 and 5>)`
- It then invokes biryani_ready function.
- Inside biryani_ready function it continously checks for all tables and whenever it finds a table with status 0, it locks the table and refill that table (makes its status 1). this is done for all its vessels.

Serving Tables :
Implemented using struct `tabl` and for each of the **m** tables, a new thread is created.

- It waits for it to be filled (status to change from 0 to 1)
- It then creates random number of slots between 1 to 10.
- It then invokes ready_to_serve_table function
- Inside the function it loops until its capacity(size of the vessel loaded by robot chef) becomes zero.
- It also regularly updates the **maxslots** variable based on its capacity. For eg: if capacity becomes 5 from 6 then slots_available will also become 5(note: it happens only when **maxslots** > capacity of the table).
- It returns from the function when capacity and thus slots_available becomes zero.
- After returning it status is again changed to 0 and it can now be rediscovered by robot chefs.

Students :
Implemented using struct `studen` and for each of the **k** students, a new thread is created.

- Each student arrives at a random time between 5 and 15 seconds.
- It then invokes wait_for_slot() function.
- Inside the function it searches for a table with available slot till it finds one.
- It then occupies that slot and decreases the capacity and available number of slots of that table.
- It then sleeps for 3 seconds which denotes time taken to eat biryani.
- It then invokes student_in_slot function. Inside the function it releases the slot and decrease the global variable **bookhe_bache** which keeps the count of remaining students.
- Whenever **bookhe_bache** reaches zero, all the functions return and all threads are killed implying simulation is over.

NOTE : Only one mutex lock is used per table and one global mutex lock is used to count the number of students remaining.