Variables

char movies_name[100][100];	Save movies’ name.
int movies_tickets[100];	Save the amount of tickets of movies.
int movie_to_watch[100];	Save which movie each customer want to watch.
int sell_result[100];	Save whether there is ticket available for each customer, written by box office agents and read by customers.
int ticket_taker_serving;	Used to pass customer number between customer and ticket taker.
int concession_worker_serving;	Used to pass customer number between customer and concession worker.
int current_customer_order;	Used to pass what the current customer want to order between customer and concession worker.

Semaphore

sem box_office = 0;	● Indicates how many agents are available in the box office. ● Customer has to wait for this semaphore before being enqueued. ● Inited to 0 because when an agent is ready to serve a customer, the agent will signal this semaphore.
sem box_office_queue_mutex = 1;	● The queue of customers in the box office may only be accessed by one thread at a time. ● When a customer has to be enqueued or an agent has to dequeue a customer, they have to wait for this semaphore. ● When they finish enqueueing of dequeueing, they will signal this semaphore.
sem customer_ready = 0;	● Indicates how many customers are already enqueued into box office queue and ready for buying ticket. ● Once a customer is enqueued into the queue of box office, the customer would signal this semaphore. ● Box office will wait for this semaphore before dequeue a customer from queue and serve the customer.

sem tickets_mutex = 1;	● The amount of available tickets of movies may only be accessed by one thread at a time. ● When an agent tries to sell a ticket, he has to wait for this semaphore, making sure there is always one thread modify the available tickets info. ● Once an agent finish selling ticket, he would signal this semaphore, letting other agents be able to sell ticket.
sem sell_ticket_finished[100] = {0};	● Indicates whether an agent complete selling ticket to a certain customer. ● Once a customer pass his customer number and which movie he want to watch to an agent, the customer will wait for his semaphore of this semaphore array. This simulates the customer waits for an agent selling a ticket. ● Once an agent complete selling a ticket, he would signal the semaphore of the customer he is serving of this semaphore array.
sem ticket_taker_ready = 0;	● When the ticket taker is ready for serving a customer, he signals this semaphore. ● After a customer is in the line seeing ticket taker, he waits for this semaphore.

sem ready_for_tearing_ticket = 0;	● Before ticket taker tears a ticket, he would wait for this semaphore to get the customer number of the customer he is serving. ● After the customer pass his customer number in to variable 'ticket_taker_serving', he signals this semaphore. Then ticket taker fetch the customer number and tear the customer’s ticket.
sem ticket_torn[100] = {0};	● Once ticket torn a customer’s ticket, he would signals the customer’s semaphore of this semaphore array. ● Customers wait for their semaphores of this semaphore array before go into theater or go to concession. This simulates customers waiting for ticket taker tearing their tickets.
sem concession_worker_ready = 0;	● Customers wait for this semaphore when they are in the line buying either popcorn or soda. ● When the concession worker is ready for serving the next customer, he signals this semaphore.

sem ready_for_ordering = 0;	● Before concession worker start filling a customer’s order, he has to wait for this semaphore. ● Once a customer pass his customer number and his order to variable 'concession_worker_serving' and 'current_customer_order', he signals this semaphore. Then concession worker fetch the customer's number and order and start filling order.
sem order_filled[100] = {0};	● Once concession worker filled a customer’s order, he would signals the customer’s semaphore of this semaphore array. ● Customers wait for their semaphores of this semaphore array before go into theater. This simulates customers waiting for concession worker filling their orders.

Pseudocode

CUSTOMER

```
void customer(i) {

    int customer_number = i;
    int movie = decide_movie();
    enter_line_at_box_office();

    /* Assign customer to agent. */
    wait(box_office);
    wait(box_office_queue_mutex);
    enqueue1(customer_number);
    signal(customer_ready);
    signal(box_office_queue_mutex);

    /* Wait for agent sell ticket. */
    wait(sell_ticket_finished[customer_number]);
    if (sell_result[customer_number] == 1) {
        /* If failed (sold out). */
        leave_theater();
        return;
    }

    /* If successful, go on to see ticket taker. */
    goto_see_ticket_taker();
    wait(ticket_taker_ready);
    ticket_taker_serving = customer_number;
    signal(ready_for_tearing_ticket);
    wait(ticket_torn[customer_number]);
    enter_main_lobby();

    /* Decide if go to concession. */
    int goto_concession = decide_concession();
    if (goto_concession) {
        goto_concession();
        int customer_order = devide_order();
        wait(concession_worker_ready);
        concession_worker_serving = customer_number;
        current_customer_order = customer_order;
        signal(ready_for_ordering);
        wait(order_filler[customer_number]);
        recieve_foods();
    }
    enjoy_movie();
    return;
}
```

BOX OFFICE AGENT

```
void box_office_agent(i) {

    int agent_number = i;
    int serve_customer;

    while (true) {
        signal(box_office);
        wait(customer_ready);

        /* Get the current serving customer’s number. */
        wait(box_office_queue_mutex);
        dequeue1(serve_customer);
        signal(box_office_queue_mutex);

        /* Get access to the ticket info. */
        wait(tickets_mutex);
        if (movies_tickets[movie_to_watch[serve_customer]] > 0) {
            movies_tickets[movie_to_watch[serve_customer]]--;
            signal(tickets_mutex);
            sell_ticket();
            sell_result[serve_customer] = 1;
        } else {
            signal(tickets_mutex);
            sell_result[serve_customer] = 0;
        }
        signal(sell_ticket_finished[serve_customer]);
    }
}
```

TICKET TAKER

```
void ticket_taker() {

    int serve_customer;
    while (true) {
        signal(ticket_taker_ready);

        /* Get the current serving customer’s number. */
        wait(ready_for_tearing_ticket);
        serve_customer = ticket_taker_serving;
        tear_ticket();
        signal(ticket_torn[serve_customer]);
    }
}
```

CONCESSION WORKER

```
void concession_worker () {
    int serve_customer;
    int fill_order;
    while (true) {
        signal(concession_worker_ready);

        /* Get the current serving customer’s number and order. */
        wait(ready_for_order);
        serve_customer = concession_worker_serving;
        fill_order = current_customer_order;
        fill_order(fill_order);
        signal(order_filled[serve_customer]);
    }
}
```