Q1. Write a C program using semaphores to synchronize access between two threads incrementing a shared counter.

Objective:

- To prevent race conditions when multiple processes access shared resources.
- To demonstrate the use of semaphore operations wait() and signal().

Q2. Write a C program using semaphores to simulate the **Producer–Consumer** problem with a bounded buffer.

Objective:

- To synchronize producer and consumer processes using semaphores.
- To ensure that the producer does not overwrite a full buffer and the consumer does not read from an empty buffer.

Q3. Write a C program using semaphores to simulate five philosophers sitting around a table, each alternating between thinking and eating, while sharing five forks.

Objective:

- To demonstrate process synchronization and resource sharing.
- To ensure that no two adjacent philosophers eat at the same time.