Execution Exercises – Date : 12.09.2025

1. Implement basic inter-process communication using System V message queues.

    • Write two separate C programs: sender.c and receiver.c.

    • The sender should send a string message to the receiver.

    • The receiver should print the message received.

    • Use msgget(), msgsnd(), and msgrcv().

    • Define a message structure with long mtype

2. *Implement a Producer-Consumer using Message Queues*

    Write two programs: producer.c and consumer.c.

    • The producer generates numbers from 1 to 10 and sends them as messages.

    • The consumer receives the numbers and prints their square.

    Use a common message queue.

    • Delete the message queue after processing

3. *Multiple Message Types Handling*

    Write a single program that sends 3 messages with different mtype values (e.g., 1 for INFO, 2 for WARNING, 3 for ERROR).

    • The receiver should be able to receive only messages of a specific type (e.g., ERROR). Allow the user to specify which message type they want to receive.

4. *POSIX Message Queue Implementation*

    • Write a program that uses POSIX message queues (mq_open, mq_send, mq_receive).

    • Create a sender and receiver program that exchange messages containing temperature sensor values.

    • Set message queue attributes (max msg, msg size).

    • Handle queue creation, usage, and cleanup.

5.*Send and Receive Structured Data*

    • Define a structure with fields: int id, char name[20], float marks.

    • Sender should send 3 student records via the message queue.

    • Receiver should read and print each student record.

6. *Message Queue with Timeout (POSIX)*

    • Modify a receiver program to wait for messages only for a fixed duration (e.g., 5 seconds).

    • If no message is received, print a timeout warning. Hint: Use mq_timedreceive() with struct timespec.

7. *Logging System via Message Queues*

    • Create a logger.c program that runs in the background and listens on a message queue.

    • Write another program log_client.c that sends messages (e.g., “User logged in”, “Error occurred”) to the logger.

    • Logger writes these messages to a file.

8. *Chat Between Two Processes*

    • Implement two processes (user1.c and user2.c) that can send and receive messages from each other using one or two queues.

    • Include sender name, timestamp, and message body.

9. *Delete and Recreate Message Queue Safely*

    • Write a script or C code that:

    • Checks if a message queue with a given key exists.

    • Deletes it if found.

    • Creates a fresh queue and starts communication.

10. *Error Handling and Edge Case Testing*

    • Intentionally try to send a message larger than the allowed size.

    • Try receiving from an empty queue in non-blocking mode.

    • Handle all errors with descriptive messages using perror().