**KASIRALA DURGA-192324073**

#### 10.Illustrate the concept of inter-process communication using message queue with a C program.

### Aim:

To demonstrate inter-process communication (IPC) using message queues in C. This allows processes to communicate with each other by sending and receiving messages.

### Algorithm:

1. **Create a message queue**: Use msgget() to create a message queue.
2. **Send a message**: Use msgsnd() to send messages to the queue.
3. **Receive a message**: Use msgrcv() to receive messages from the queue.
4. **Remove the queue**: Use msgctl() to remove the message queue after use.

### Procedure:

1. Initialize the message queue.
2. Send a message from the sender process.
3. Receive the message in the receiver process.
4. Clean up the message queue.

### Code:

### #include <stdio.h>

### #include <sys/ipc.h>

### #include <sys/msg.h>

### #include <string.h>

### struct message {

### long msg\_type;

### char msg\_text[100];

### };

### int main() {

### key\_t key = ftok("progfile", 65); // Generate unique key

### int msgid = msgget(key, 0666 | IPC\_CREAT); // Create message queue

### struct message msg;

### msg.msg\_type = 1; // Message type (should be positive)

### strcpy(msg.msg\_text, "Hello from sender!"); // Message content

### msgsnd(msgid, &msg, sizeof(msg) - sizeof(long), 0); // Send message

### printf("Message sent: %s\n", msg.msg\_text); // Confirm message sent

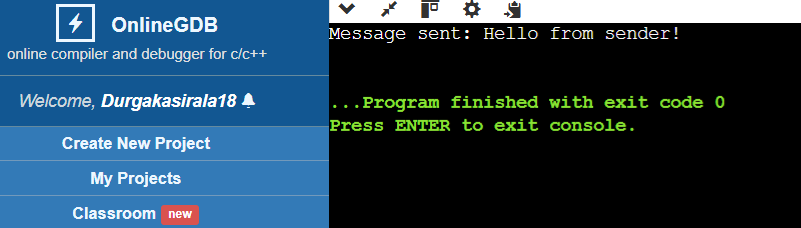
### return 0;

### }

### Result:

* The sender sends the message "Hello from sender!" to the message queue.
* The receiver receives the message and prints: Received message: Hello from sender!.

**Output:**

****