4. Develop a C program which demonstrates inter-process communication between a reader process and a writer process. Use mkfifo, open, read, write and close APIs in your program.

Theory:
mkfifo (make FIFO) is a Linux command and system call used in Unix-like systems to create a named pipe, a special file that acts as a communication channel for processes.

PROGRAM: We'll create two separate programs: **writer.c** and **reader.c**

1. **writer.c** This program will create a FIFO (if it doesn’t already exist) and write a message to it.

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <string.h>

#define FIFO_NAME "/tmp/myfifo"
#define BUFFER_SIZE 256

int main() {
    int fd;
    const char *message = "Hello from writer process!";

    if (mkfifo(FIFO_NAME, 0666) == -1) {
        perror("mkfifo");
        exit(EXIT_FAILURE);
    }
    printf("Writer: FIFO created\n");

    fd = open(FIFO_NAME, O_WRONLY);
    if (fd == -1) {
        perror("open");
        exit(EXIT_FAILURE);
    }
    printf("Writer: FIFO opened for writing\n");

    size_t message_length = strlen(message);
    if (write(fd, message, message_length) == -1) {
        perror("write");
        close(fd);
        exit(EXIT_FAILURE);
    }
    printf("Writer: Message written to FIFO\n");

    close(fd);

    return 0;
}
```

2. **reader.c** This program will open the FIFO and read the message written by the writer process.

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>

#define FIFO_NAME "/tmp/myfifo"
#define BUFFER_SIZE 256

int main() {
    int fd;
    char buffer[BUFFER_SIZE];
    ssize_t bytesRead;
    fd = open(FIFO_NAME, O_RDONLY);
    if (fd == -1) {
        perror("open");
        exit(EXIT_FAILURE);
    }

    bytesRead = read(fd, buffer, BUFFER_SIZE - 1);
    if (bytesRead == -1) {
        perror("read");
        close(fd);
        exit(EXIT_FAILURE);
    }

    buffer[bytesRead] = '\0';
    printf("Reader: Message received: %s\n", buffer);
    close(fd);

    if (unlink(FIFO_NAME) == -1) {
        perror("unlink");
        exit(EXIT_FAILURE);
    }

    return 0;
}
```

Compilation and Execution:

1. Compile the writer and reader programs one by one using below command:

```
braham@braham:~/Desktop/program$ gcc writer.c -o writer
braham@braham:~/Desktop/program$ gcc reader.c -o reader
```

2. Open two terminals:

- In the first terminal, run the writer:./writer
- In the second terminal, run the reader:./reader

OUTPUT:

Check the both terminal to see the output...

```
braham@braham:~/Desktop/program$ ./writer
Writer: FIFO created
Writer: FIFO opened for writing
Writer: Message written to FIFO
braham@braham:~/Desktop/program$ ./reader
Reader: Message received: Hello from writer process!
```