**7b OS 33185**

#include <stdio.h>

#include <stdlib.h>

#include <sys/ipc.h>

#include <sys/shm.h>

#include <string.h>

#include <unistd.h>

#define SHM\_KEY 1234

#define SHM\_SIZE 1024

int main() {

int shmid = shmget(SHM\_KEY, SHM\_SIZE, 0666 | IPC\_CREAT);

if (shmid == -1)

{

perror("shmget failed");

exit(1);

}

char \*shm\_ptr = (char \*)shmat(shmid, NULL, 0);

if (shm\_ptr == (char \*)-1)

{

perror("shmat failed");

exit(1);

}

printf("Server: Enter a message to write to shared memory: ");

fgets(shm\_ptr, SHM\_SIZE, stdin);

shm\_ptr[strlen(shm\_ptr) - 1] = '\0';

printf("Server: Message written to shared memory: %s\n", shm\_ptr);

printf("Server: Waiting for client to read the message...\n");

while (\*shm\_ptr != '\*') {

sleep(1);

}

shmdt(shm\_ptr);

shmctl(shmid, IPC\_RMID, NULL);

printf("Server: Shared memory cleaned up. Exiting.\n");

return 0;

}

Test Case 1:

Server:

Server: Enter a message to write to shared memory: Hello, Client!

Server: Message written to shared memory: Hello, Client!

Server: Waiting for client to read the message...

Server: Shared memory cleaned up. Exiting.

Client:

Client: Message read from shared memory: Hello, Client!

Client: Memory detached. Exiting.

Test Case 2:

Server:

Server: Enter a message to write to shared memory: Shared memory is cool.

Server: Message written to shared memory: Shared memory is cool.

Server: Waiting for client to read the message...

Server: Shared memory cleaned up. Exiting.

Client:

Client: Message read from shared memory: Shared memory is cool.

Client: Memory detached. Exiting.

Test Case 3:

Server:

Server: Enter a message to write to shared memory: Have a great day!

Server: Message written to shared memory: Have a great day!

Server: Waiting for client to read the message...

Server: Shared memory cleaned up. Exiting.

Client:

Client: Message read from shared memory: Have a great day!

Client: Memory detached. Exiting.