

EX 7 IPC USING SHARED MEMORY

AJAY SRINIVAS R

230701017

```
liveuser@localhost:live-~$ cat > sender.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>
#include <unistd.h>

#define SHM_SIZE 1024 // Size of the shared memory segment

int main() {
    key_t key = ftok("shafila", 85); // Generate a unique key
    int shmid = shmget(key, SHM_SIZE, 0666 | IPC_CREAT); // Allocate shared memory segment

    if (shmid == -1) {
        perror("shmget failed");
        exit(1);
    }

    char *str = (char *)shmat(shmid, NULL, 0); // Attach to shared memory

    if (str == (char *){-1}) {
        perror("shmat failed");
        exit(1);
    }

    printf("Writing to shared memory...\n");
    sprintf(str, "Welcome to Shared Memory"); // Write message to shared memory
    sleep(2); // Sleep to give time for receiver to read

    shmdt(str); // Detach from shared memory
    return 0;
}

liveuser@localhost:live-~$ cat > receiver.c
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <string.h>

#define SHM_SIZE 1024 // Size of the shared memory segment

int main() {
    key_t key = ftok("shafila", 85); // Generate a unique key
    int shmid = shmget(key, SHM_SIZE, 0666 | IPC_CREAT); // Allocate shared memory segment

    if (shmid == -1) {
        perror("shmget failed");
        exit(1);
    }

    char *str = (char *)shmat(shmid, NULL, 0); // Attach to shared memory

    if (str == (char *){-1}) {
        perror("shmat failed");
        exit(1);
    }

    printf("Message Received: %s\n", str); // Read and print the message from shared memory

    shmdt(str); // Detach from shared memory
    shmctl(shmid, IPC_RMID, NULL); // Remove shared memory segment
    return 0;
}

printf("Message Received: %s\n", str); // Read and print the message from shared memory

shmdt(str); // Detach from shared memory
shmctl(shmid, IPC_RMID, NULL); // Remove shared memory segment
return 0;
}

liveuser@localhost:live-~$ gcc sender.c -o sender # Compile sender.c
liveuser@localhost:live-~$ gcc receiver.c -o receiver # Compile receiver.c
liveuser@localhost:live-~$ ./sender # Run sender process
Writing to shared memory...
liveuser@localhost:live-~$ ./receiver # Run receiver process
Message Received: Welcome to Shared Memory
```