Operating System – CS23431

Ex 7	**IPC Shared Memory**
Name: B M Madhumitha	
Reg No: 230701168	

Aim:

To write a C program to do Inter Process Communication (IPC) using shared memory between sender process and receiver process.

Algorithm:

sender

1. Set the size of the shared memory segment
2. Allocate the shared memory segment using shmget
3. Attach the shared memory segment using shmat
4. Write a string to the shared memory segment using sprintf
5. Set delay using sleep
6. Detach shared memory segment using shmdt

receiver

1. Set the size of the shared memory segment
2. Allocate the shared memory segment using shmget
3. Attach the shared memory segment using shmat
4. Print the shared memory contents sent by the sender process.
5. Detach shared memory segment using shmdt

Program Code:

sender.c

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/ipc.h>
#include<sys/shm.h>

#define size 1024
int main(){
   key_t key = ftok("shmfile",65); //get key from file to key

   int shmid = shmget(key,size,IPC_CREAT | 0666);//get id of size with permission 0666
   if(shmid==-1){
      perror("shmget");
      exit(1);
   }
```

```
    char *shmadr =(char*) shmat(shmid,NULL,0);//attach id to address

    printf("Enter the message to send:");
    fgets(shmadr, size, stdin);

    shmdt(shmadr);

    return 0;
}
```

receiver.c

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/ipc.h>
#include<sys/shm.h>
#define size 1024
int main(){
    key_t key = ftok("shmfile",65);

    int shmid = shmget(key, size, 0666);
    if(shmid == -1){
        perror("shmid");
        exit(1);
    }
    char *shmadr = (char*) shmat(shmid,NULL,0);
    if(shmadr == (char*)-1){
        perror("shamdr");
        exit(1);
    }

    printf("Receiver: Data read from shared memory: %s\n", shmadr);

    shmdt(shmadr);
    shmctl(shmid, IPC_RMID, NULL); // Remove shared memory
    return 0;
}
```

Sample Output

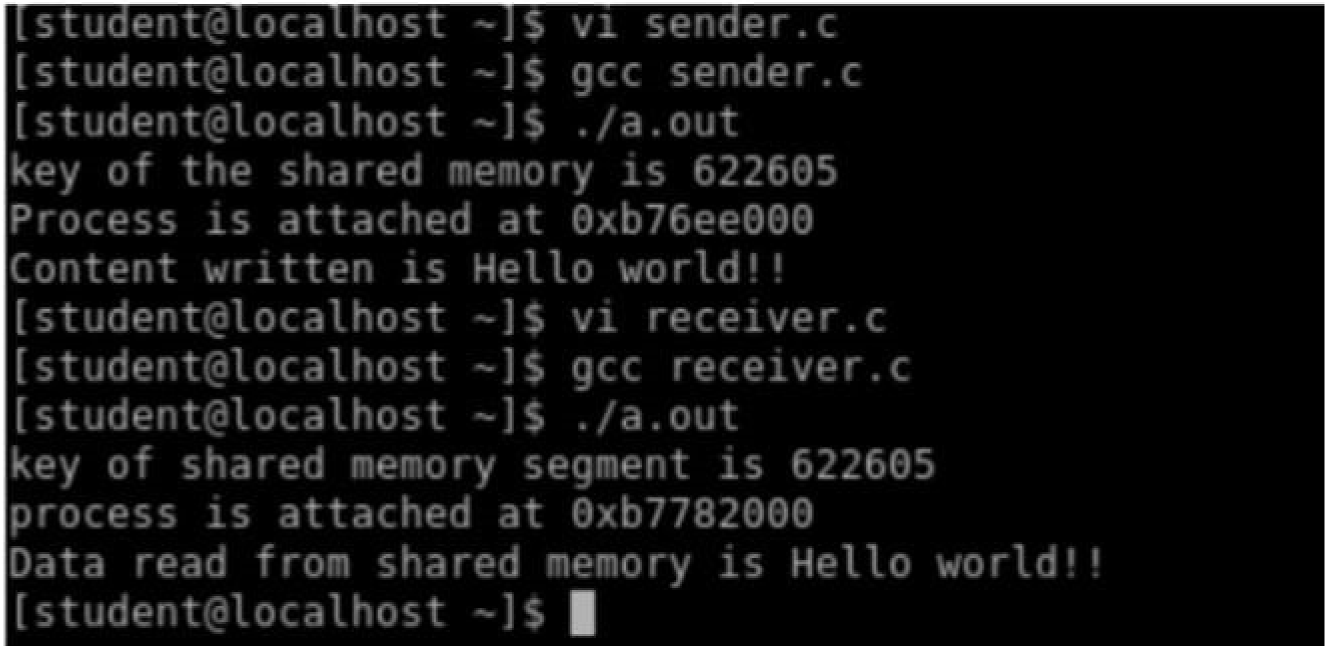
Terminal 1

```
[root@localhost student]# gcc sender.c -o sender
[root@localhost student]# ./sender
```

Terminal 2

```
[root@localhost student]# gcc receiver.c -o receiver
[root@localhost student]# ./receiver
Message Received: Welcome to Shared Memory
[root@localhost student]#
```

Output:



```
[student@localhost ~]$ vi sender.c
[student@localhost ~]$ gcc sender.c
[student@localhost ~]$ ./a.out
key of the shared memory is 622605
Process is attached at 0xb76ee000
Content written is Hello world!!
[student@localhost ~]$ vi receiver.c
[student@localhost ~]$ gcc receiver.c
[student@localhost ~]$ ./a.out
key of shared memory segment is 622605
process is attached at 0xb7782000
Data read from shared memory is Hello world!!
[student@localhost ~]$
```

Result: Thus, the program was executed successfully.