EXPERIMENT - 5
Inter Process Communication using Shared Memory

AIM:

C program Writer, attaches itself to the shared memory segment created in Reader Process and it reads the content of the shared memory.

ALGORITHM

1) Start the program
2) Declare the variables, shmid
3) shmat() and shmdt() are used to attach and detach shared memory segments.
   void *shmat(int shmid, const void *shmaddr, int shmflg);
   int shmdt(const void *shmaddr);
4) shmat() returns a pointer, shmaddr, to the head of the shared segment associated with a validshmid.
5) shmdt() detaches the shared memory segment located at the address indicated byshmaddr
6) SharedMemory_Writer.c creates the string and shared memory portion.
7) SharedMemory_Reader.c attaches itself to the created shared memory portion and uses the string (printf)
8) Stop the program.

PROGRAM:

/* sharedMemoryWriter.c */

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
int main()
{
/*ftok function is used to generate unique key which is for System V IPC functions */
key_t key = ftok("shmfile",65);
/*shmget functionreturns the shared memory identifier associated with key in shmid */
int shmid = shmget(key,1024,0666|IPC_CREAT);
/*shmat function is used to attach to the shared memorysegment associated with the shared memory identifier, shmid, to the address space of the calling process. */
char *str = (char*) shmat(shmid,(void*)0,0);
printf("Write Data: ");
//scanf("%s", str);
//gets(str);
fgets(str, 25, stdin);
printf("\n Data written in memory: %s\n",str);
/* shmdtfunction detaches the shared memory segment located at the specified address from the address space of the calling process */
shmdt(str);
```

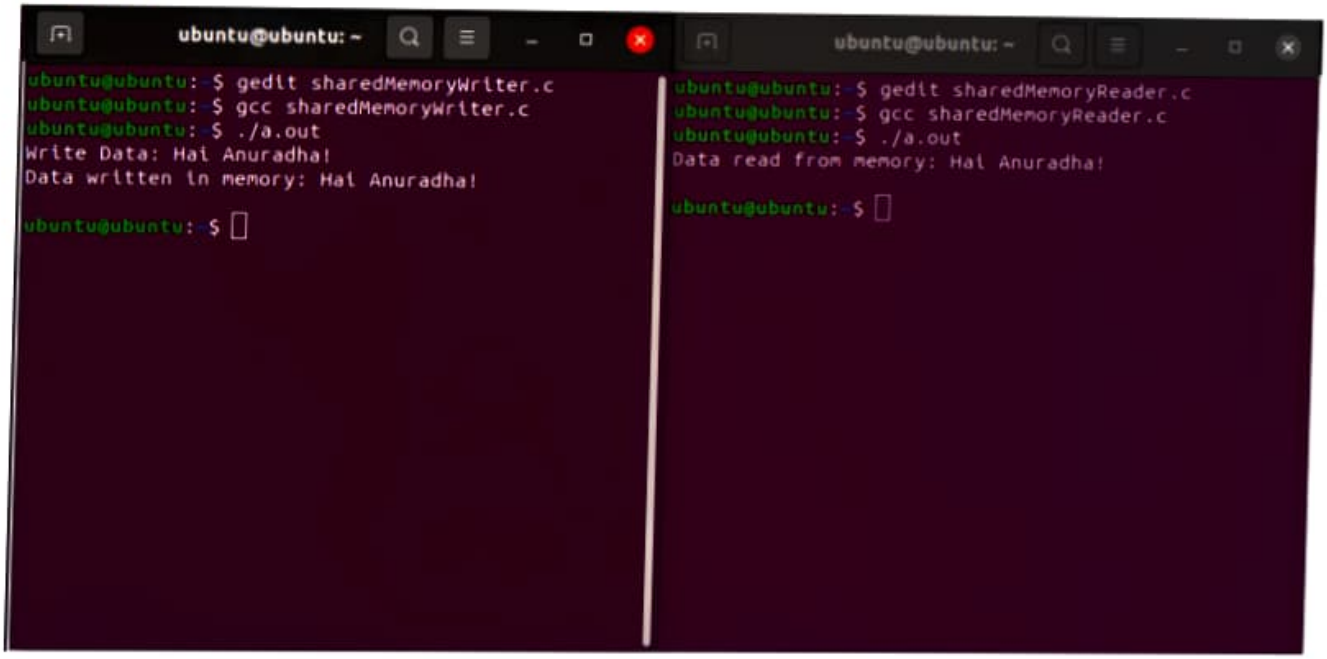
```
return 0;
}
```

/* sharedMemoryReader.c */

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

int main()
{
/*ftok to generate unique key */
key_t key = ftok("shmfile",65);
// shmget returns an identifier in shmid
int shmid = shmget(key,1024,0666|IPC_CREAT);
// shmat to attach to shared memory
char *str = (char*) shmat(shmid,(void*)0,0);
printf("Data read from memory: %s\n",str);
//detach from shared memory
shmdt(str);
// destroy the shared memory
shmctl(shmid,IPC_RMID,NULL);
return 0;
}
```

OUTPUT:



RESULT:

The program is compiled, executed and the output is verified.

Date: