Name	Anurag Nair
UID no.	2022600035
Experiment No.	8

AIM:	Chat bot using Shared Memory
Program	

PROGRAM:	Chat1.c
	
	#include <stdio.h>
	#include <stdlib.h>
	#include <unistd.h>
	#include <string.h>
	#include <semaphore.h>
	#include <fcntl.h>
	#include <sys/types.h>
	#include <sys/ipc.h>
	#include <sys/shm.h>
	#include <sys/wait.h>
	
	#define SHM_SIZE 1024
	
	int main()
	{
	  char *shared_memory;
	  int shmid;
	
	  shmid = shmget(20, SHM_SIZE, IPC_CREAT | 0666);
	  if (shmid == -1)
	  {
	  perror("Error in creating shared memory");
	  exit(1);
	  }
	  printf("Memory segment successfully created\n");
	
	  shared_memory = shmat(shmid, NULL, 0);
	  if (shared_memory == (char *)-1)

```
    {
        perror("Error in attaching memory");
        exit(1);
    }

    sem_t *sem_user1, *sem_user2;
    int flag = 0;

    sem_user1 = sem_open("/user1_semaphore", O_CREAT | O_EXCL,
0644, 0);
    if (sem_user1 == SEM_FAILED)
    {
        sem_unlink("/user1_semaphore");
        sem_user1 = sem_open("/user1_semaphore", O_CREAT | O_EXCL,
0644, 0);
        if (sem_user1 == SEM_FAILED)
        {
            perror("Semaphore creation failed");
            exit(EXIT_FAILURE);
        }
    }

    sem_user2 = sem_open("/user2_semaphore", O_CREAT | O_EXCL,
0644, 1);
    if (sem_user2 == SEM_FAILED)
    {
        sem_unlink("/user2_semaphore");
        sem_user2 = sem_open("/user2_semaphore", O_CREAT | O_EXCL,
0644, 1);
        if (sem_user2 == SEM_FAILED)
        {
            perror("Semaphore creation failed");
            exit(EXIT_FAILURE);
        }
    }

    while (1)
    {
        sem_wait(sem_user2);
        if (flag == 1)
```

```
            printf("User 2 says: %s", shared_memory);
        flag = 1;
        printf("Reply to User 2: ");
        fgets(shared_memory, SHM_SIZE, stdin);
        sem_post(sem_user1);
    }
    return 0;
}
```

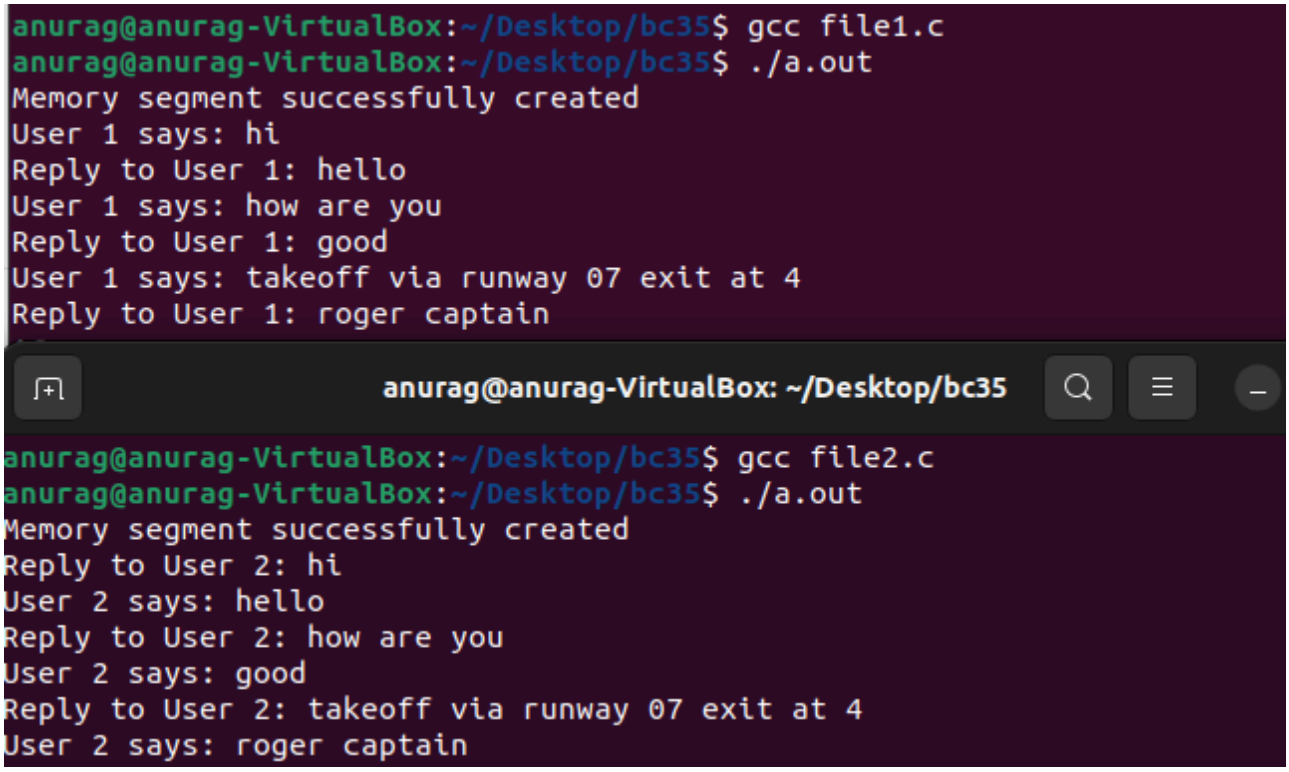
Chat2.c

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <semaphore.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/wait.h>

#define SHM_SIZE 1024

int main()
{
    char *shared_memory;
    int shmid;

    shmid = shmget(20, SHM_SIZE, 0666);
    if (shmid == -1)
    {
        perror("Error in creating shared memory");
        exit(1);
    }
    printf("Memory segment successfully created\n");

    shared_memory = shmat(shmid, NULL, 0);
    if (shared_memory == (char *)-1)
    {
```

```
    perror("Error in attaching memory");
    exit(1);
  }

  sem_t *sem_user1, *sem_user2;

  sem_user1 = sem_open("/user1_semaphore", 0);
  if (sem_user1 == SEM_FAILED)
  {
    sem_unlink("/user1_semaphore");
    sem_user1 = sem_open("/user1_semaphore", O_CREAT | O_EXCL,
0644, 0);
    if (sem_user1 == SEM_FAILED)
    {
      perror("Semaphore creation failed");
      exit(EXIT_FAILURE);
    }
  }

  sem_user2 = sem_open("/user2_semaphore", 1);
  if (sem_user2 == SEM_FAILED)
  {
    sem_unlink("/user2_semaphore");
    sem_user2 = sem_open("/user2_semaphore", O_CREAT | O_EXCL,
0644, 1);
    if (sem_user2 == SEM_FAILED)
    {
      perror("Semaphore creation failed");
      exit(EXIT_FAILURE);
    }
  }

  while (1)
  {
    sem_wait(sem_user1);
    printf("User 1 says: %s", shared_memory);
    printf("Reply to User 1: ");
    fgets(shared_memory, SHM_SIZE, stdin);
    sem_post(sem_user2);
  }
```

	return 0; }

OUTPUT:



CONCLUSION:	From this experiment, I learned how two programs can talk to each other using shared memory and semaphores in C. Shared memory lets them share data, while semaphores help them take turns and avoid messing up the shared data. It showed me the importance of making sure processes don't interfere with each other when sharing data.