```
//user1
#include<iostream>
#include <signal.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <unistd.h>
using namespace std;

#define FILLED 0
#define Ready 1
#define NotReady -1

struct memory {
  char buff[100];
  int status, pid1, pid2;
};

struct memory* shmptr;

// handler function to print message received from user2

void handler(int signum)
{
  // if signum is SIGUSR1, then user 1 is receiving a message
from user2

  if (signum == SIGUSR1) {
    cout<<"User2: "<<shmptr->buff<<endl<<endl;
  }
}

int main()
{
  // process id of user1

  int pid = getpid();

  int shmid;

  // key value of shared memory
  int key = 12345;

  // shared memory create
  shmid = shmget(key, sizeof(struct memory), IPC_CREAT |
0666);

  // attaching the shared memory
```

```
  shmptr = (struct memory*)shmat(shmid, NULL, 0);

  // store the process id of user1 in shared memory
  shmptr->pid1 = pid;
  shmptr->status = NotReady;

  // calling the signal function using signal type SIGUSER1
  signal(SIGUSR1, handler);

  while (1) {
    while (shmptr->status != Ready)
    continue;
    sleep(1);

    // taking input from user1

    cout<<"User1: ";
    cin.getline(shmptr->buff, 100);

    shmptr->status = FILLED;

    // sending the message to user2 using kill function

    kill(shmptr->pid2, SIGUSR2);
  }

  shmdt((void*)shmptr);
  shmctl(shmid, IPC_RMID, NULL);
  return 0;
}
```

```
// user 2
#include<iostream>
#include <signal.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <sys/types.h>
#include <unistd.h>
using namespace std;

#define FILLED 0
#define Ready 1
#define NotReady -1

struct memory {
  char buff[100];
  int status, pid1, pid2;
};

struct memory* shmptr;

// handler function to print message received from user1

void handler(int signum) {
  // if signum is SIGUSR2, then user 2 is receiving a message
from user1

  if (signum == SIGUSR2) {
    cout<<"User1: "<<shmptr->buff<<endl<<endl;
  }
}

// main function

int main()
{
  // process id of user2
  int pid = getpid();

  int shmid;

  // key value of shared memory
  int key = 12345;

  // shared memory create

  shmid = shmget(key, sizeof(struct memory), IPC_CREAT |
0666);
```

```
  // attaching the shared memory

  shmptr = (struct memory*)shmat(shmid, NULL, 0);

  // store the process id of user2 in shared memory
  shmptr->pid2 = pid;

  shmptr->status = NotReady;

  // calling the signal function using signal type SIGUSR2
  signal(SIGUSR2, handler);

  while (1) {
    sleep(1);

    // taking input from user2

    cout<<"User2: ";
    cin.getline(shmptr->buff, 100);

    shmptr->status = Ready;

    // sending the message to user1 using kill function

    kill(shmptr->pid1, SIGUSR1);

    while (shmptr->status == Ready) {
      continue;
    }
  }

  shmdt((void*)shmptr);
  return 0;
}
```

```
Someshwars-MacBook-Pro:Assignment8 someshwargaikwad$ ./user1
User2: Hey, Whats up?

User1: Nothing Much.
User2: What happened to your girl?

User1: We broke up..
```

```
Someshwars-MacBook-Pro:Assignment8 someshwargaikwad$ ./user2
User2: Hey, Whats up?
User1: Nothing Much.

User2: What happened to your girl?
User1: We broke up..

User2: Sed
```

```
Someshwars-MacBook-Pro:Assignment8 someshwargaikwad$ ./user1
User2: Hey, Whats up?

User1: Nothing Much.
User2: What happened to your girl?

User1: We broke up..
User2: Sed

User1:
```

```
Someshwars-MacBook-Pro:Assignment8 someshwargaikwad$ ./user2
User2: Hey, Whats up?
User1: Nothing Much.

User2: What happened to your girl?
User1: We broke up..

User2: Sed
```