```c
#include <sys/mman.h>

//****structure d echange client/ serveur*****
struct requete {
int type;// consultation=0 , reservation=1
int spectacles;//numéro de spectacle de 0 a ...
int nb_place_dispo;//place disponibles pour le spectacle donné
int nb_place_resa;//nombre de place a reserver sur spectacle donné
int confirm_resa;//confirmation reservation, 0 ou 1
};

//***** Création du sémaphore;*****
sem_t semaphore;

//***tableau des spectacles et nbr places***
int spect[3]={10,15,20};

//initialisation de la structure
void init_struct(struct requete *stru){
        stru->type=99;
        stru->spectacles=99;
        stru->nb_place_dispo=99;
        stru->nb_place_resa=99;
        stru->confirm_resa=99;

}

//structure pour threads
struct dfonc
{
        int *sock_fd;
        struct requete *structure;
        int *srvc_fd;
};

//*****declaration pour serveur leger*****
struct dfonc socs;

// Création d'un tableau de thread
pthread_t threads[3];



//**************************************
//*****menu client*****
int menu_client(struct requete *stru){

        int choix;

        printf("********* *********** *************\n");
        printf("********* MENU CLIENT *************\n");
        printf("********* *********** *************\n");
        printf("Pour consulter, tapez 0 (zéro)\n");
        printf("Pour réserver, tapez 1 (un)\n");
        printf("Pour fermer, tapez 2 (deux)\n");
        printf("**********************************\n");

        scanf("%d",&choix);

        if( choix == 0 || choix == 1){
                stru->type=choix;

                printf("**************************************\n");
                printf("Veuillez choisir votre spectacle\n");
                printf("tapez 0 pour le concert\n");
```

```
                printf("tapez 1 pour le theatre\n");
                printf("tapez 2 pour l' opéra\n");
                printf("tapez 3 pour fermer\n");
                printf("**************************************\n");
                scanf("%d",&choix);
                printf("**************************************\n");

                if(choix==0||choix==1||choix==2){
                        stru->spectacles=choix;
                        if(stru->type==1){
                                printf("Combien de places voulez vous réserver?\n");
                                printf("**************************************\n");
                                stru->nb_place_resa=scanf("%d",&choix);
                                printf("**************************************\n");
                        }

                }else{
                        return 0;
                }


        }else{
                return 0;
        }

return 1;
}


//*****reponse serveur*****
//*************************
void reponse_serv(struct requete *stru){

        //si consultation
        if(stru->type==0){
                printf("Le nombre de places disponibles est: %d\n", stru-
>nb_place_dispo );
        }
        //si reservation
        if (stru->type==1){
                if(stru->confirm_resa==1){
                        printf("La reservation de vos %d places a été faite, il reste %d
places disponibles\n",stru->nb_place_resa,stru->nb_place_dispo );
                }else{
                        printf("Erreur! il n' y a plus assez de places disponibles\n");
                }
        }
}


//*****definition de la memoire partagée*****
void* shm_creation(size_t size) {
  //creation des flags
  int protection = PROT_READ | PROT_WRITE;
  int visibility = MAP_SHARED | MAP_ANONYMOUS;

  return mmap(NULL, size, protection, visibility, -1, 0);
}

//*****fonction 2ieme thread consultation*****
void * thread_resa(){

        // On attend la disponibilité du sémaphore
        sem_wait(&semaphore);

        if((spect[socs.structure->spectacles]-socs.structure->nb_place_resa) >= 0){
```

```
                spect[socs.structure->spectacles] = spect[socs.structure->spectacles]-
socs.structure->nb_place_resa;
                socs.structure->nb_place_dispo = spect[socs.structure->spectacles];
                socs.structure->confirm_resa = 1;
        }else{
                socs.structure->nb_place_dispo = spect[socs.structure->spectacles];
                socs.structure->confirm_resa = 0;
        }

        // On relache le sémaphore
        sem_post(&semaphore);

    send(*socs.srvc_fd, socs.structure, sizeof(struct requete), 0);
        close(*socs.srvc_fd);
}

//*****fonction 3ieme thread reservation*****
void * thread_consult(){

        //renseignement du nombre de places
        socs.structure->nb_place_dispo = spect[socs.structure->spectacles];

    send(*socs.srvc_fd, socs.structure, sizeof(struct requete), 0);
        close(*socs.srvc_fd);
}

//*****fonction 1ier thread reception*****
void * thread_reception(){

        recv(*socs.srvc_fd, socs.structure, sizeof(struct requete), 0);

        //si reservation
        if(socs.structure->type == 1){
                int err2;
                if ((err2 = pthread_create(&threads[1], NULL, &thread_resa, NULL)) != 0) {
                        printf("Echec de la création du thread: [%s]", strerror(err2));
                }
        printf("Création du thread reservation numéro 1\n");
        pthread_join(threads[1], NULL);

        //si consultation
        }else{
                int err3;
                if ((err3 = pthread_create(&threads[2], NULL, &thread_consult, NULL)) !=
0) {
                        printf("Echec de la création du thread: [%s]", strerror(err3));
                }
        printf("Création du thread consultation numéro 2\n");
        pthread_join(threads[2], NULL);
        }

}
```