

```

/* Svolgimento Compito 1, 3, 5, 7 - 21/12/05 */

#include <stdio.h>
#include <rpc/rpc.h>
#include "opfile.h"
#include <dirent.h>

static ListaStanze lista;

int *aggiungi_stanza_l_svc (AggStanza *richiesta, struct svc_req *rp )
{
    static int result;
    int indicePrimaLibera, i;

    result=-1;
    indicePrimaLibera = -1;

    printf("Ricevuta richiesta aggiunta stanza %s, tipo %c\n", richiesta->nomeStanza,
    richiesta->tipo);

    // controllo che non esista un'altra stanza con lo stesso nome
    for (i = 0; i < N; i++)
    {
        if ( strcmp(richiesta->nomeStanza, lista.elenco[i].nomeStanza)==0 )
            return &result;
        // e mantengo la prima posizione libera
        if ( (strcmp(lista.elenco[i].nomeStanza, "L")==0) && (indicePrimaLibera == -1) )
            indicePrimaLibera = i;
    }
    if (indicePrimaLibera == -1)
        // non ci sono posizioni libere, insuccesso
        result=-1;
    else
    {
        strcpy(lista.elenco[indicePrimaLibera].nomeStanza, richiesta->nomeStanza);
        lista.elenco[indicePrimaLibera].tipo=richiesta->tipo;
        result=0;
    }
    return &result;
}

ListaStanze *elimina_utente_l_svc (char **utente, struct svc_req *rp )
{
    static ListaStanze risultato;
    int cont, i, j, isUtente;

    risultato.ris=-1;
    cont = 0;

    printf("Ricevuta richiesta eliminazione utente %s\n", *utente);

    for (i = 0; i<N; i++) {
        isUtente = -1;
        for (j = 0; j < K; j++) {
            if ( strcmp(*utente, lista.elenco[i].utente[j].nome)==0 ) {
                strcpy(lista.elenco[i].utente[j].nome, "L");
                risultato.ris++;
                strcpy(risultato.elenco[risultato.ris].nomeStanza, lista.elenco[i].nomeStanza);
                risultato.elenco[risultato.ris].tipo=lista.elenco[i].tipo;
                for(j=0; j<K; j++)

```

```

        strcpy(risultato.elenco[risultato.ris].utente[j].nome,
lista.elenco[i].utente[j].nome);
        break;
    } //if
} //for
}
return &risultato;
}

```

```

int *aggiungi_utente_1_svc (AggUtente *utente, struct svc_req *rp )
{
    static int result;
    int indiceStanza, indiceUtente, i;
    result=-1;
    indiceStanza = -1;

    printf("Ricevuta richiesta aggiunta stanza %s, utente %s\n", utente->nomeStanza,
utente->nomeUtente);

    // controllo che esista la stanza alla quale si vuole aggiungere l'utente
    for (i = 0; i < N; i++){
        if ( strcmp(utente->nomeStanza, lista.elenco[i].nomeStanza)==0 ) {
            indiceStanza = i;
            break;
        }
    }
    if (indiceStanza >= 0){
        int indiceUtente = -1;
        // controllo esistenza di un utente con lo stesso nome
        for (i = 0; i < K; i++) {
            if ( strcmp(utente->nomeUtente, lista.elenco[indiceStanza].utente[i].nome)==0 )
                return &result;
            if ( (strcmp(lista.elenco[indiceStanza].utente[i].nome, "L")==0) && (indiceUtente
== -1) )
                indiceUtente = i;
        }
        // non ci sono posizioni libere, restituisco false
        if (indiceUtente == -1)
            return &result;
        else {
            strcpy(lista.elenco[indiceStanza].utente[indiceUtente].nome, utente->nomeUtente);
            result=0;
            return &result;
        }
    }
    else
        return &result; // la stanza non e' presente
}

```

```

int *elimina_stanza_1_svc (char **nomeStanza, struct svc_req *rp )
{
    static int result;
    int i;
    result=-1;

    printf("Ricevuta richiesta eliminazione stanza %s\n", *nomeStanza);

    int indiceStanza = -1;
    // controllo che esista la stanza che si vuole cancellare
    for (i = 0; i < N; i++)
        if ( strcmp(*nomeStanza, lista.elenco[i].nomeStanza)==0 ) {

```

```

        indiceStanza = i;
        break;
    }
    if (indiceStanza >= 0) {
        strcpy(lista.elenco[indiceStanza].nomeStanza, "L");
        lista.elenco[indiceStanza].tipo='L';
        for(i=0; i<K; i++)
            strcpy(lista.elenco[indiceStanza].utente[i].nome, "L");
        result=0;
        return &result;
    } else
        return &result; // la stanza non e' presente
}

```

```

ListaStanze *visualizza_utente_1_svc (char **nomeUtente, struct svc_req *rp )
{
    static ListaStanze risultato;
    int cont, i, j, isUtente;
    risultato.ris=-1;
    cont = 0;

    printf("Ricevuta richiesta visualizzazione utente %s\n", *nomeUtente);

    for (i = 0; i<N; i++) {
        isUtente = -1;
        for (j = 0; j < K; j++) {
            if ( strcmp(*nomeUtente, lista.elenco[i].utente[j].nome)==0 ) {
                isUtente=0;
                break;
            }
        }
        if( isUtente==0 ){
            risultato.ris++;
            strcpy(risultato.elenco[risultato.ris].nomeStanza, lista.elenco[i].nomeStanza);
            risultato.elenco[risultato.ris].tipo=lista.elenco[i].tipo;
            for(j=0; j<K; j++)
                strcpy(risultato.elenco[risultato.ris].utente[j].nome,
lista.elenco[i].utente[j].nome);
        }
    }
    return &risultato;
}

```

// NOTA!!! Da invocare nello stub del server (file opfile\_svc.c

```

void initStanze()
{
    int i,j;
    for(i=0; i<N; i++){
        strcpy(lista.elenco[i].nomeStanza, "L");
        lista.elenco[i].tipo='L';
        for(j=0; j<K; j++)
            strcpy(lista.elenco[i].utente[j].nome, "L");
    }
    printf("Inizializzazione terminata\n");
}

```