

```
/*
```

```
Es 3 Compito 16/04/09
```

```
Il servizio ruptime fornisce il nome degli host attivi sulla sottorete e il loro tempo totale di attività. Ecco la relativa pagina di MAN.
```

```
NAME
```

```
ruptime -- show host status of local machine
```

```
SYNOPSIS
```

```
ruptime [-alrtu]
```

```
DESCRIPTION
```

```
Ruptime gives a status line like uptime for each machine on the local network; these are formed for packets broadcast by each host on the network once a minute. Machines for which no status report has been received for 11 minutes are shown as being down.
```

Fornire lo pseudocodice per un client ruptime, facendo tutte le opportune ipotesi e mettendo in evidenza le chiamate alla socket library che utilizzate.

```
OPERAZIONI:
```

- 1) Creazione socket e configurazione per l'invio di pacchetti ogni minuto a tutti gli host.
- 2) Creazione socket ricezione (IP/porta host mittente richiesti da command line) + utilizzo select()+utilizzo recvfrom()

```
*/
```

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
#include <netinet/in.h>
```

```
main(char *argv[])
```

```
int servSock[];
```

```
int port;
```

```
int PortNo;
```

```
char uptime;
```

```
int sock_upt;
```

```
int on = 1;
```

```
struct sockaddr_in sin;
```

```
char add_client;
```

```
char info;
```

```
//creo il socket per la trasmissione agli host della rete con modalita connectionless
```

```
if ((sock_upt = socket(PF_INET, SOCK_DGRAM, 0)) < 0) {
    perror("socket");
    exit(1);
}
```

```
//attraverso la funzione si setsockopt abilito il broadcast
```

```
if (setsockopt(sock_upt, SOL_SOCKET, SO_BROADCAST,
    &on, sizeof(on)) < 0) {
    perror("setsockopt");
    exit(1);
}
```

```

//inizializzo il socket
sin.sin_family = PF_INET;
sin.sin_addr.s_addr = INADDR_BROADCAST;
sin.sin_addr.s_port = htons(5555);

while(1) {
    @adesso = localtime(time);
    // solo nei primi due secondi di ogni minuto
    if($adesso[0]<2) {
        sendto(sock_upt,&info,lenght_uptime,0,sockaddr*)&sin,add_lenght);
    }
}
// Creo l'array di descrittori di socket

for (port = 0; port < lenght; port++){

// Recupero la porta da utilizzare dalla lista di porte specificata a linea di comando

portNo = atoi(argv[port + 2]);
//richiedo indirizzo ip dell'host da cui voglio ricevere le informazioni sull'uptime
printf("Inserire indirizzo ip dell'host");
scanf(%c,add_client);

// Creo il socket relativo a quella porta e a quell'indirizzo e faccio il bind

servSock[port] = CreateUDPServerSocket(portNo,add_client);
}

while (running){

// Azzero l'insieme dei socket controllati che saranno controllati dalla select. Deve essere
chiamata prima di ogni select().

FD_ZERO(&rset);

// Inserisco i socket creati nella lista di descrittori che la select controlla

for (port = 0; port < noPorts; port++)
FD_SET(servSock[port], &rset);

// Imposto il timeout prima di ogni select()

selTimeout.tv_sec = 6600; // secondi

if (select(servSock + 1, &rset, NULL, NULL, &selTimeout) == 0)
printf("Timeout");//suppongo che se l'host non mi risp entro 11 min vuol dire che non è
collegato e quindi non avrò richieste in lettura
else {
    //ricevo l'uptime del client di cui ho fatto richiesta specificando l'indirizzo del
mittente
n=recvfrom(servSock + 1,&uptime,lenght,flags,sockaddr*) &add_client,add_lenght);
if(n<0)printf("errore");
else
printf(uptime);//visualizza le informazioni legate all'uptime del host richiesto

```

```
close(servSock+1);
```