Aula 6

25 de novembro de 2020 11:40

Ex. 9:

Servidor:

char resposta[5];

```
if(sendto( sockfd , buffer , strlen(buffer) , 0 , (struct sockaddr*)&cli_addr , sizeof(cli_addr) ) == SOCKET_ERROR)
        Abort("SO nao conseguiu aceitar o datagram");
```

resposta → strlen(resposta)

resposta = Converte_int_asci(strlen(buffer));

itoa

sprintf → semelhante do printf, mas imprime numa variável

no ecrã → printf("%d", strlen(buffer));

numa variável → sprintf(variável_destino, "%d", strlen(buffer));

sprintf_s(variável_destino, sizeof(variável_destino), ...);

Ex 10:

Servidor

int resposta;

resposta=strlen(buffer);

```
if(sendto( sockfd , buffer , strlen(buffer) , 0 , (struct sockaddr*)&cli_addr , sizeof(cli_addr) ) == SOCKET_ERROR)
        Abort("SO nao conseguiu aceitar o datagram");
```

(char*)&resposta, sizeof(resposta);

Cliente:

int resposta;

```
nbytes=recvfrom(sockfd , buffer , sizeof(buffer) , 0 , (struct sockaddr*) &serv_check , &serv_check_len);
```

(char*) &resposta, sizeof(resposta)

```
printf("\n<CLI1>Mensagem recebida {%s}\n",buffer);
```

%d resposta

Ex. 11:

Servidor:

Variáveis:

```
int iResult, nbytes,cli1_len, cli2_len;
struct sockaddr_in serv_addr,cli1_addr, cli2_addr ;
```

Ciclo while:

```
cli1_len=sizeof(cli1_addr);
cli2_len=sizeof(cli2_addr);
    while(1){

        fprintf(stderr,"<SER1>Esperando datagram...\n");

        nbytes=recvfrom(sockfd , buffer , sizeof(buffer) , 0 , (struct sockaddr*) &cli1_addr , &cli1_len);

        if(nbytes == SOCKET_ERROR)
            Abort("Erro na recepcao de datagrams");

        nbytes=recvfrom(sockfd , buffer , sizeof(buffer) , 0 , (struct sockaddr*) &cli2_addr , &cli2_len);

        if(nbytes == SOCKET_ERROR)
            Abort("Erro na recepcao de datagrams");


        if(sendto( sockfd ,(char *) &cli1_addr ,sizeof(cli1_addr) , 0 , (struct sockaddr*)&cli2_addr , sizeof(cli2_addr) ) == SOCKET_ERROR)
            Abort("SO nao conseguiu aceitar o datagram");

       printf("<SER1>Par formado ...\n");

    }
```

Cliente:

Variáveis:

```
int msg_len, iResult, nbytes, tam_len;
struct sockaddr_in serv_addr, addr ,cli_addr;
```

Espera pela resposta:

```
if(sendto( sockfd , argv[msg_idx] , msg_len , 0 , (struct sockaddr*)&serv_addr , sizeof(serv_addr) ) == SOCKET_ERROR)
        Abort("SO nao conseguiu aceitar o datagram");


printf("<CLI1>Mensagem enviada ao servidor ...\n");

tam_len=sizeof(addr);
nbytes=recvfrom(sockfd , (char *) &cli_addr , sizeof(cli_addr) , 0 ,(struct sockaddr*) &addr   ,&tam_len );

if(nbytes == SOCKET_ERROR)
            Abort("Erro na recepcao de datagrams");

if(nbytes != sizeof(cli_addr))
   Abort("Mensagem recebida do tipo inesperado");

if(strcmp(argv[ip_idx], inet_ntoa(addr.sin_addr))==0 && atoi(argv[port_idx])==ntohs(addr.sin_port))
{
   // mensagem veio do servidor

  nbytes=sendto(sockfd, (char *) &cli_addr, sizeof(cli_addr), 0 , (struct sockaddr *) &cli_addr, sizeof(cli_addr));

  if(nbytes==SOCKET_ERROR)
     Abort("Erro ao enviar o endereço ao par remoto");

  printf("Sou o cliente 2\n");
  printf("O meu par remoto -> IP: %s, Porto: %d\n",inet_ntoa(cli_addr.sin_addr),ntohs(cli_addr.sin_port));
}
else
{
     printf("Sou o cliente 1\n");
     printf("Par remoto -> IP: %s, Porto: %d \n",   inet_ntoa(addr.sin_addr), ntohs(addr.sin_port));
}
```