Complete o código apresentado respeitando as seguintes regras:

- Não apague os comentários;
- Substitua apenas as sequências "/*...*/" com o código esperado;
- Mantenha a formatação "Bold+ Fundo amarela" nas suas resposta

/*_____ Cliente.C _____
Este cliente destina-se a enviar mensagens passadas na linha de comando, sob
a forma de um argumento, para um servidor especifico cuja locacao e' dada
na linha de comando também.

O cliente, depois de enviar uma mensagem ao servidor, aguarda pela recepcao de uma
mensagem. Verifica se a mensagem foi enviada pelo servidor. Se a mensagem recebida
tiver sido enviado pelo servidor, este cliente deve "contar" a toda gente a mensagem do servidor.

O protocolo usado e' o UDP.
_____*/

```
#include <stdio.h>
#include <winsock.h>
#define BUFFERSIZE     4096

/*QUESTÃO 1: Defina o endereço IP para broadcast (mensagem por difusão ou para "toda gente")*/

#define TODA_GENTE_ADDR /*...*/ 255.255.255.255
#define TODA_GENTE_PORT  6000

void Abort(char* msg);
```

```
/*QUESTÃO 2: Inicialize os winsocks*/

    iResult = /*...*/(MAKEWORD(2, 2), /*...*/);

    if (iResult != 0) {

        printf("WSAStartup failed: %d\n", iResult);

        getchar();

        exit(EXIT_FAILURE);

    }


    if (argc != 4) { /*Testa sintaxe*/

        fprintf(stderr, "Sintaxe: %s frase_a_enviar ip_destino_serv porto_destino_serv \n", argv[0]);

        exit(EXIT_FAILURE);

    }
```

Caminho: p » strong » span

```
/*QUESTÃO 3: Defina a função e seus parametros para abrir o socket*/

    if ((sockfd = /*...*/( /*...*/, /*...*/ , /*...*/) ) == INVALID_SOCKET)

        Abort("Impossibilidade de criar socket");


/*================ ACTIVA POSSIBILIDADE DE ENVIO POR DIFUSAO ===============*/
```

```
/*QUESTÃO 4: Active a possibilidade de envio por difusão , ou seja, para "toda a gente" */

    opt = /*...*/;

    setsockopt(sockfd, /*...*/ , /*...*/, /*...*/, sizeof(opt));


/*================== PREENCHE ENDERECO DO SERVIDOR ====================*/

    memset((char*)&serv_addr, 0, sizeof(serv_addr));

    serv_addr.sin_family = AF_INET;

    serv_addr.sin_addr.s_addr = inet_addr(argv[2]);

    serv_addr.sin_port = htons(atoi(argv[3]));


/*------------------ PREENCHE ENDERECO PARA O ENVIO DE MENSAGENS POR DIFUSÃO ----------
```

*QUESTÃO 5: Preencha os dados para o envio de mensagens por difusão , ou seja, para "toda a gente" */

```
memset((char*)&todos_addr, 0, sizeof(todos_addr));
todos_addr.sin_family = /*...*/;
todos_addr.sin_addr.s_addr = /*...*/;
todos_addr.sin_port = htons( /*...*/);
```

AF_INET

inet_addr(TODA_GENTE_ADDR)

TODA_GENTE_PORT

====================== ENVIA MENSAGEM AO SERVIDOR ==================/

```
msg_len = strlen(argv[1]);
```

/*QUESTÃO 6: Preencha os parametros para enviar a mensagem ao servidor*/

sendto argv[1] (struct sockaddr *) &serv_addr

```
if ( /*...*/(sockfd, /*...*/, /*...*/, 0, /*...*/, /*...*/) == SOCKET_ERROR)
    Abort("SO nao conseguiu aceitar o datagram");
```

sizeof(serv_addr)

```
printf("<CLI1> Mensagem enviada ao servidor.\n");
tam_addr = sizeof(addr);
```

/*QUESTÃO 7: Preencha os parametros para aguardar uma mensagem*/

(struct sockaddr *) &addr

recvfrom

```
nbytes = /*...*/(sockfd, buffer, sizeof(buffer), 0, /*...*/, /*...*/);
```

&tam_addr

```
if (nbytes == SOCKET_ERROR)
    Abort("Erro ao receber mensagem");
```

/*VERIFICA SE A MENSAGEM RECEBIDA FOI ENVIADA PELO SERVIDOR*/

```
/*QUESTÃO 8: Verifique se a mensagem recebida foi enviada pelo servidor*/
    if (strcmp( /*...*/) == 0 && atoi( /*...*/)) {

/*ENVIA A MENSAGEM RECEBIDA DO SERVIDOR A TODA A GENTE*/
```

inet_ntoa(addr.sin_addr), inet_ntoa(serv_addr.sin_addr)

↳ itoa(addr.sin_port == serv_addr.sin_port)

```
/*QUESTÃO 9: Envia a mensagem recebida do servidor a "toda a gente."*/
        nbytes = /*...*/(sockfd, buffer, nbytes, 0, /*...*/, /*...*/);

    if (nbytes == SOCKET_ERROR)
        Abort("Erro ao enviar a mensagem a todos");
    printf("<CLI1> Mensagem enviada a toda gente\n");
}
```

sendto

(struct sockaddr *) &todos_addr

↳ sizeof(todos_addr)

```
/*QUESTÃO 10: Feche o socket*/
    /*...*/
    exit(EXIT_SUCCESS);
}

/*_____ Abort _____
Mostra a mensagem de erro associada ao ultimo erro no SO e abando com
"exit status" a 1
_____
*/
void Abort(char* msg) {
```

closesocket(socketfd);