Trabalho de Sokets: Jogo: Batalha Naval

Alunos: Viviani Andrade e Carol Vieira

Regras do Jogo:

- Há um tabuleiro de 5x5, ou seja, 25 blocos. Há 3 navios escondidos (um em cada bloco).
- O objetivo do jogar é descobrir onde estão estes navios e acertá-los.
- A cada tiro dado é dito se você acertou algum navio. Caso tenha errado, é dito quantos navios existem naquela linha e naquela coluna.
- O jogo só acaba quando você descobrir e afundar os 3 navios.

- Legenda pro usuário:
- ~ : água no bloco. Ainda não foi dado tiro.
- o : tiro dado, não há nada ali.
- X : tiro dado, havia um navio ali.

Código do Cliente

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>

#define SERVER "192.168.1.14"
#define PORT 8888

//variáveis do socket
int s_socket;
struct sockaddr_in destino;
int conexao, len;
```

```c
main(){
    int tabuleiro[5][5];
    int navios[3][2];
    int tiro[2];
    int tentativas=0,acertos;

        //criação do socket.
        s_socket = socket(AF_INET, SOCK_STREAM, 0);
    if(s_socket <0){
                perror("Erro na criação de socket!");
        }

    len = sizeof(destino);
    destino.sin_family = AF_INET;
    destino.sin_port = htons(PORT);
    destino.sin_addr.s_addr = inet_addr(SERVER);
    bzero(&(destino.sin_zero),0);

    conexao = connect(s_socket, (struct sockaddr * ) &destino, len);
    if(conexao < 0){
                perror("Erro na conexão");
        close(s_socket);
        }

        //chama o método para inicializar o tabuleiro.
    inicializaTabuleiro(tabuleiro);
        printf("\n");
        printf("**********************************************************************\n");
        printf("********************************* BATALHA NAVAL ************************\n");
        printf("*************************************************Por Carol e Viviani****\n");
        printf("\n");
```

```c
//Método que inicializa todo o tabuleiro 5x5 com o valor -1.
void inicializaTabuleiro(int tabuleiro[][5]){
    int linha, coluna;
        for(linha=0 ; linha < 5 ; linha++ )
            for(coluna=0 ; coluna < 5 ; coluna++ )
                tabuleiro[linha][coluna]=-1;
}
```

```
do{
        //chama o método mostraTabule
    mostraTabuleiro(tabuleiro);
        printf("\n");
        //chama o método darTiro
    darTiro(tiro);
    printf("\n");
    //envia as informações de tiro[
    send(s_socket, &tiro[0], sizeof
    send(s_socket, &tiro[1], sizeof
tentativas++;

        //variável que corresponde
        int resp;
        //recebe do servidor o "ace
        recv(s_socket, &resp, sizeo
        recv(s_socket, &navios, siz
        //printf("Resp: %d\n",resp

        //se o método "acertou" do
        if(resp==1){
                printf("VOCÊ ACERT
    dica(tiro,navios,tentativas);
    printf("\n");
    acertos++;
    //envia para o servidor os ace
                send(s_socket, &ac
}
//se o método "acertou" do servido
else{
                printf("VOCÊ ERROU
                dica(tiro,navios,t
                printf("\n");
                //envia para o ser
                send(s_socket, &ac
        }
        //chama o método que atualiza o tabuleiro d
        alteraTabuleiro(tiro,navios,tabuleiro,resp)

}while(acertos!=3);

printf("\nJOGO TERMINADO! Você acertou os 3 navios em
//mostra o tabuleiro final, já atualizado.
mostraTabuleiro(tabuleiro);
```

```
//Método que imprime a matriz (tabuleiro).
void mostraTabuleiro(int tabuleiro[][5]){

    int linha, coluna;
```

```
//Método que pede para o cliente a linha e a coluna que ele deseja atirar.
//Guarda-se essas informações em um array de 2 posições.
void darTiro(int tiro[2]){

        printf("Linha: ");
        scanf("%d",&tiro[0]);
        tiro[0]--;

        printf("Coluna: ");
        scanf("%d",&tiro[1]);
        tiro[1]--;
```

posição com "~"

```
//Método que informa para o cliente dicas para a próxima jodada.
void dica(int tiro[2], int navios[][2], int tentativa){
        int linha=0,
            coluna=0,
            fila;

        //conta quantos navios há na linha tiro[0]
        for(fila=0 ; fila < 3 ; fila++){
            if(navios[fila][0]==tiro[0])
                linha++;
            if(navios[fila][1]==tiro[1])
                coluna++;
        }

        printf("
}
```

```
//Método que atualiza o tabuleiro com base nas tentativas de tiro do cliente.
void alteraTabuleiro(int tiro[2], int navios[][2], int tabuleiro[][5], int resp){
                //variável "resp" corresponde ao método "acertou" do servidor.
        if(resp==1)
            tabuleiro[tiro[0]][tiro[1]]=1;
        else
            tabuleiro[tiro[0]][tiro[1]]=0;
    }
```

Código do Servidor

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <arpa/inet.h>
#include <sys/types.h>
#include <sys/socket.h>

#define PORT 8888
#define CLIENTE "192.168.1.14"

//variáveis do socket
struct sockaddr_in s_destino, s_origem;
int s_socket, sockfd_cliente, len, lenRemoto, retornoBind;
```

Código do Servidor

```
main(){
        int tabuleiro[5][5];
    int navios[3][
        int tiro[2
    int tentativas

        //crição d
        s_socket =
    if(s_socket <
                pe
    }
    len = sizeof(s
        bzero(&(s_
    s_destino.sin_
    s_destino.sin_
    s_destino.sin_
        retornoBin
    if(retornoBind
                pr
    }
    listen(s_socke
    printf("\n");
        printf("**
        printf("**
        printf("**

    //chama o método para iniciar os navios.
    iniciaNavios(navios);
```

```
//Método  que inicializa os navios em posições aleatórias entre 1 e 5.
void iniciaNavios(int navios[][2]){
        srand(time(NULL));
        int navio, anterior;

        for(navio=0 ; navio < 3 ; navio++){
                //array navios que guarda as posições aleatórias a qual os navios irão estar no tabuleiro.
            navios[navio][0]= rand()%5;
            navios[navio][1]= rand()%5;
           // printf("L: %d, C: %d\n",navios[navio][0],navios[navio][1]);

            //agora vamos checar se esse par não foi sorteado
            //se foi, só sai do "do...while" quando sortear um diferente
            for(anterior=0 ; anterior < navio ; anterior++){
                if( (navios[navio][0] == navios[anterior][0])&&(navios[navio][1] == navios[anterior][1]) )
                    do{
                        navios[navio][0]= rand()%5;
                        navios[navio][1]= rand()%5;
                    }while( (navios[navio][0] == navios[anterior][0])&&(navios[navio][1] == navios[anterior][1]) );
            }

        }
    }
```

```
while(1){
        sockfd_cliente = accept( s_socket, (struct sockaddr *) &s_destino, &lenRemoto);

        int tiro[2];
        int resp;
        do{
                //recebe do cliente
                recv(sockfd_cliente,
                recv(sockfd_cliente,
                //printf("Linha: %d
                //printf("Coluna: %

                //variável "resp" r
                resp=acertou(tiro,n
                //printf("Resp: %d\

                //envia para o clie
                send(sockfd_cliente,
                send(sockfd_cliente,
                //recebe do cliente a quantidade de acertos.
                recv(sockfd_cliente, &acertos, sizeof(acertos), 0);
        }while(acertos!=3);
        //close(sockfd_cliente);
```

```
//Método que verifica se o tiro foi dado exatamente na posição onde se encontra um navio.
//Se acertou retorna 1, se não retorna 0.
int acertou(int tiro[2], int navios[][2]){
    int navio;
        for(navio=0 ; navio < 3 ; navio++){
            if(tiro[0]==navios[navio][0] && tiro[1]==navios[navio][1]){
                //printf("Você acertou o tiro (%d,%d)\n",tiro[0]+1,tiro[1]+1);
                return 1;
            }
        }
        return 0;
    }
```