Trabalho Prático 1

Redes de Computadores

Henrique Rotsen Santos Ferreira

2020100945

Documentação do Código:

Inclusões de Bibliotecas:

O código inclui diversas bibliotecas padrão do C, bem como um arquivo de cabeçalho personalizado chamado common.h. Este arquivo contém definições e declarações necessárias para o programa.

Definições de Constantes:

O arquivo common.h define constantes que representam estados das células no campo minado, assim como comandos que podem ser enviados pelo cliente.

Struct Action:

A struct action armazena informações sobre a ação realizada pelo jogador ou enviada ao jogador. Ela contém o tipo de ação, as coordenadas e o estado do tabuleiro.

Funções Auxiliares (declaradas em common.h):

*logexit(const char *msg)*: Imprime uma mensagem de erro e encerra o programa.

*addrparse(const char *addrstr, const char *portstr, struct sockaddr_storage *storage)*: Converte uma representação de endereço e porta para uma estrutura sockaddr_storage.

*addrtostr(const struct sockaddr *addr, char *str, size_t strsize)*: Converte um endereço de sockaddr para uma string legível.

*server_sockaddr_init(const char *proto, const char *portstr, struct sockaddr_storage *storage)*: Inicializa a estrutura sockaddr_storage com as informações de endereço do servidor.

convert(int n): Converte um número em um caractere correspondente, provavelmente para representar um estado em um jogo.

Função main:

A função principal do programa. Ela lida com a configuração inicial, inicializações de sockets, leitura do gabarito do campo minado a partir de um arquivo, e estabelece a comunicação com os clientes.

Inicialização e Configuração do Servidor:

O servidor é inicializado para aceitar conexões tanto via IPv4 quanto IPv6.

Loop de Aceitação de Clientes:

O servidor entra em um loop infinito para aceitar conexões de clientes.

Processamento de Mensagens dos Clientes:

O código processa as mensagens dos clientes, atualizando o estado do jogo conforme as ações realizadas.

Envio de Respostas aos Clientes:

O servidor envia respostas de volta aos clientes, informando o estado atual do jogo.

Fechamento de Conexões:

As conexões são fechadas apenas após o cliente dar EXIT. Ou seja, mesmo que ocorra o “WIN” ou “GAME OVER”, é possível dar um "reset’ e começar de novo sem que a conexão encerre.

Fechamento de Arquivos e Sockets:

Arquivos e sockets são devidamente fechados ao final do programa.

Desafios:

Para fazer o trabalho, tive que aprender toda a parte de programação em redes, para isso vi a playlist disponibilizada no Moodle. Após vê-la, tive que testar o código e pensar em como seria implementada a troca de mensagens seguindo o padrão exigido na documentação do TP.

De modo geral, o TP pareceu bem explicado, com apenas certas ambiguidades que vi na documentação, mas nada que dificultasse a realização dele. O principal que era fazer o cliente/servidor era o mais complicado, porém as aulas disponibilizadas são muito boas, com até mais detalhes de implementação que o necessário para esse trabalho.

Problemas e Dificuldades:

Durante o trabalho, tive alguns problemas no que tange às mensagens, percebi ao realizar testes com outros colegas que o “board” ficava com lixo de memória dependendo do colega, para resolver isso, implementei uma cópia do estado atual do jogo no servidor. Ou seja, fiz uma *“struct action userboard”*, que é nada mais nada menos que uma mensagem que fica apenas no servidor, para evitar de caso o cliente gere uma nova mensagem a cada iteração eu pegar lixo de memória.

Na minha implementação eu trabalhei com a ideia de que temos uma “*struct action msg”*, e essa mensagem deveria ser alterada tanto pelo servidor, quanto pelo cliente, porém como não foi especificado como isto era para ser feito, ocorreu o problema do lixo de memória, caso alguém instanciasse uma nova mensagem toda vez.

No mais tive alguns problemas com tratamento de erro, pois foquei muito em resolver erros que faziam sentido para mim, porém não foi pedido para ser resolvido. Um exemplo é o que acontece se eu desse um *"remove_flag"* em uma célula já revelada? Isto não é um erro descrito na documentação, e inicialmente eu tratei todos esses, porém vi que dessa forma tinha um problema ao usar o meu cliente com o servidor de outra pessoa.

Testes:

Fiz inúmeros testes no meu servidor e no cliente, a fim de testar todas as ocasiões descritas no problema. Os testes testaram tanto as conexões do meu TP em IPv4 e IPv6, quanto estas conexões em clientes e servidores de outros colegas.