DCC023 Redes de Computadores– Turma TE (METATURMA)
Universidade Federal de Minas Gerais – UFMG
Marcelo Costa Monteiro – 2014139487 - Belo Horizonte - 04/09/2020

# TRABALHO PRÁTICO 1 - JOGO DA FORCA

**Resumo:** O trabalho em duas aplicações socket: um servidor que recebe palpites (caracteres) de um cliente e retorna se o palpite está presente ou não em uma palavra pré-determinada, bem como quantas vezes aparece e a posição. O cliente recebe os caracteres como entradas fornecidas pelo usuário e exibe no terminal a palavra encontrada até o momento. A aplicação foi desenvolvida na linguagem C, usando padrão POSIX, protocolo TCP e compatível com IPv4 e IPv6.

**Executando**

Para executar o projeto:

1. Abra o diretório raiz, onde há o Makefile e execute o comando: make
2. Execute o servidor com o comando: ./servidor <porta> (ex: ./servidor 51511)
3. Em um terminal separado, execute o cliente: ./cliente <ip-servidor> <porta-servidor> (./cliente 127.0.0.1 5151)

**Implementação**

Para iniciar as conexões socket, desenvolvi três funções:

- *initSocketServer:* iniciam o socket no servidor através dos comandos socket(), bind() e listen().
- *acceptConnection:* aguarda e recebe, quando houver, uma conexão com algum cliente.
- *connectToServer*: tenta se conectar via socket ao servidor.

A implementação das funções acima está presente no arquivo ***libs/socketHandler.h***. Todas as três retornam um inteiro corresponte ao socket a ser utilizado. Foi utilizada a estrutura *sockaddr_storage* visto que a mesma comporta implementações IPv4 e IPv6.

O arquivo ***servidor.c*** implementa o servidor. Você pode definir a palavra a ser descoberta e o protocolo IP a ser utilizado alterando os valores de IPv e WORD nas linhas iniciais deste arquivo. Assim que alguma conexão é aceita, o servidor envia o tamanho da palavra, através da função send1(). Essa e as demais funções send2(), send3() e send4() correspondem aos tipos de mensagens do protocolo de comunicação, definidos na documentação do trabalho.

Há também as funções recvByte(), usada para receber as mensagens do tipo 1 e 2, e a função recvAnswer(), para mensagens do tipo 3 e 4. Por fim, a função charFind() é responsável por calcular quantas ocorrências do caracter há na palavra e informar as posições. A implementação dessas funções está no arquivo ***libs/common.h***. Optei por implementar as funções separadamente a fim de organizar melhor os arquivos servidor.c e cliente.c, bem como facilitar uma possível reutilização.

**Detalhes adicionais**

O cliente.c usa o array *word* para armazenar os caracteres que já foram encontrados. Há um '_' no lugar dos caracteres ainda não descobertos. Ao descobrir todas as palavras o cliente exibe uma mensagem final para o usuário congratulando-o e exibindo a palavra completa. O servidor, por sua vez, usa o arrau *charsTried* para registrar os palpites já realizados *triesCounter* e *charsFound* para contar as tentativas e acertos, e usando para verificar quando o jogo termina.

Também foi criada a função _upper(), que realiza uma operação simples com o valor ASCII, e assim sendo possível trabalhar tanto caracteres caixa alta quanto caixa baixa.