UNIVERSIDADE DE SÃO PAULO

INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO

**INTRODUÇÃO À CIÊNCIA DA COMPUTAÇÃO - TRABALHO 04: “REDE SOCIAL”**

Gustavo Romanini Gois Barco

10749202

Isabella Ferreira Lopes

10872281

Lucas Alves Roris

11913771

SÃO CARLOS

Junho de 2020

1. **INTRODUÇÃO À ATIVIDADE**

Nesta tarefa, foi proposto o desenvolvimento de uma rede social com funcionalidades similares às do “Twitter”. Os principais tópicos da linguagem C utilizados foram arquivos, ponteiros, alocação dinâmica, Makefile e estruturas de dados. Os requisitos deste exercício foram:

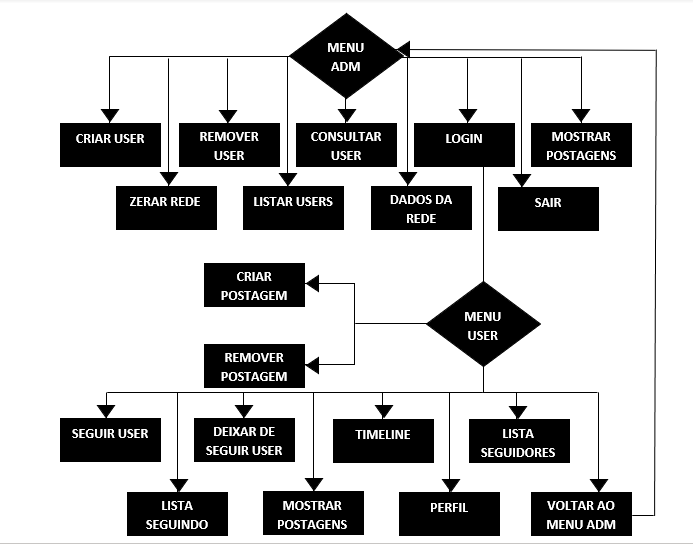
* Menu de Administrador com possibilidade de consultar e listar informações gerais sobre a rede e postagens, inserir e remover usuários, excluir todos os dados da rede, fazer login como usuário e sair do programa;
* Menu de Usuário com possibilidade de seguir e deixar de seguir outros, fazer postagens e removê-las, consultar seu perfil e timeline e retornar ao menu de administrador;
* Armazenamento das informações entre as execuções do programa.

1. **LÓGICA**

A primeira importante decisão tomada pelo grupo foi relativa à manipulação dos dados: foi decidido que, ao invés de tratar as informações diretamente dos arquivos, seria melhor criar structs, que, ao início da execução do programa, receberiam as informações dos arquivos. Durante o uso da rede social, todas as alterações seriam feitas diretamente nas structs, e salvas nos arquivos através da função **Salvar**.

Foram criados seis arquivos do tipo “.C” para separar o código em blocos de funcionalidade similar, para facilitar a visualização, o entendimento e, por fim, a concretização do Makefile.

O fluxo planejado é apresentado a seguir.



1. **DESENVOLVIMENTO**

As funções do programa estão dispostas da seguinte forma dos arquivos.C:

* **ALOCACAO.C**

Armazena as funções que verificam, a cada alocação ou realocação de ponteiros, se o retorno foi igual a NULL. Nesse caso, o programa é abordado e uma mensagem é apresentada ao usuário. São cinco funções, e cada uma analisa ponteiros de um dos tipos: ponteiro de int, ponteiro de char, ponteiro de ponteiro de char, ponteiro de dados e ponteiro de dadosPost.

* **FUNCOESBD.C**

Aqui foram colocadas as funções que tratam, de maneira direta, os dados e as traduções entre struct e arquivos. O nome se refere a “funções de banco de dados”, e elas são:

1. **Inicializar**: traduz (pela função **fread**)e transfere (por laços de repetição com os elementos das structs) os dados dos arquivos para as structs de usuários e dados;
2. **Salvar**: utiliza a função **fwrite** para escrever nos arquivos todos os elementos das structs, seguindo um padrão para que, futuramente, possam ser acessados na devida ordem;
3. **Terminar**: usa a função **free** para liberar os ponteiros do programa;
4. **VerificarExistencia**: recebe um nickname como parâmetro e percorre a struct de usuários para checar a validade daquele dado. Caso exista, retorna o número correspondente àquele usuário, do contrário, retorna -1.
5. **MostrarInformacoes**: printa os números totais de usuários e postagens da rede social e apresenta o(s) usuário(s) com maior número de seguidores.

* **INTERFACE.C**

Neste arquivo ficam as funções **MenuSLogin** e **MenuCLogin**, que printam as interfaces dos menus anterior e posterior ao login, respectivamente. Essas funções são chamadas dentro de cada um dos dois laços principais da **Main**.

* **POST.C**

Nesse arquivo foram desenvolvidas todas as funções que manipulam, diretamente, as postagens na rede social, seja essa manipulação feita pelo menu de administrador ou pelo de usuário. São elas:

1. **AdicionarPost**: antes de essa função ser chamada, é feita a realocação e a verificação do ponteiro

Na função principal, os arquivos **Usuario** e **Posts** são abertos pela função **fopen**. Em Seguida, a função **ftell** é utilizada para buscar dois dos parâmetros primordiais da inicialização do programa: o número de usuários existentes e o número de postagem. Esses números são armazenados em ponteiros inteiros. O motivo da escolha delas como ponteiros é que serão manipuladas pela maioria das funções do programa.

Os ponteiros **usuarios** e **posts**, que serão dinamizados, são inicializados como NULL, e depois realocados de acordo com os números anteriormente buscados. A cada realocação, e chamada a função **VerificarAlocacao** correspondente. Nesse ponto, é chamada a função **Inicializar**.

São criadas as variáveis **x** e **y**, que serão lidas do usuário enquanto ele não fechar o programa. **x** lê a opção referente ao menu de administrador, e y referente ao de usuário. Há tratamento para opções inválidas, e cada opção corresponde a um número. Quando esse número é inserido, é chamada a função correspondente.