**Eric T. Zancanaro**

**Trabalho 3 - Algoritmos e Programação**

**Sistema de Diários**

Implementar um sistema capaz de manter diários de diversos usuários através de arquivos. O programa deve iniciar perguntando o nome e a senha de um usuário, procurá-lo no arquivo de logins e, se encontrado, exibir um menu com as opções abaixo. Caso o usuário não exista, crie a entrada do usuário e a adicione no arquivo de logins.

* Mostrar todo o diário na tela;
* Mostrar uma página específica do diário utilizando um índice;
* Escrever nova página no diário;
* Remover uma página específica do diário, utilizando um índice;
* Salvar o diário.

O sistema deve manter um arquivo “**logins.safe**” contendo as informações sobre os usuários e, para cada usuário, manter um arquivo contendo seu diário (Pode ser um arquivo texto ou binário, a discrição de vocês.).

Para implementar o trabalho, será disponibilizado no moodle um arquivo **funcoes.h** contendo a definição das *structs* que devem ser utilizadas para representar os usuários e o diário, bem como os protótipos das funções que devem ser implementadas. **As structs e as funções presentes no arquivo funcs.h NÃO podem ser modificadas.** Caso achem necessário, podem criar novas funções nos arquivos de implementação.

As funções que devem ser implementadas são descritas abaixo:

**data \*criaData(int diaMesAno[3]);**

Cria um ponteiro para a estrutura data e aloca dinamicamente a memória. Usando o ponteiro, atribui os valores para os campos da estrutura (dia,mês,ano) sabendo que **diaMesAno[0]** é o dia, **diaMesAno[1]** é o mês e **diaMesAno[2]** é o ano. Retorna o ponteiro para a data.

**pagina \*criaPagina(int indice,int diaMesAno[3],char \*txt);**

Aloca memória para um valor do tipo página e faz a atribuição dos campos de acordo com os valores recebidos na chamada da função. (Dica: lembre da função criaData);

Retorna o ponteiro para a página criada.

**diario \*criaDiario();**

Aloca espaço na memória para um novo diário com o conteúdo de seus campos vazio. Retorna o ponteiro.

**int \*getData(data \*dia);**

Retorna um vetor contendo a data no formato vet[0] é o dia,[1] é o mês, [2] é o ano. (Dica: crie o vetor com malloc());

**diario \*carregaDiario(FILE \*arqDiario);**

Lê o arquivo contendo o diário do usuário, cria uma estrutura para armazenar o diário (malloc) e carrega o conteúdo do arquivo(criando as páginas para armazenar os valores lidos). Caso o arquivo não exista, cria um novo diário vazio.

Retorna um ponteiro para o diário criado.

**void imprimePag(pagina \*pag,FILE \*arquivo);**

Imprime uma página do diário no arquivo. Se o ponteiro para o arquivo for NULL, imprime a página na tela (printf).

**pagina \*buscaPagina(int indice,diario \*d);**

Procura no diário **d** uma página contendo o índice passado como parâmetro da função. Retorna o ponteiro para a página.

**char \*fazLogin(char nome[256],char senha[256],FILE \*logins);**

Carrega o arquivo de logins e procura o nome do usuário (**DICA: carregue várias estruturas do tipo usuário**). Caso encontre, testa se a senha fornecida é igual a senha lida do arquivo (**DICA: use a biblioteca string.h para testar a igualdade usando a função streq()**)..

**void novaPagina(diario \*d);**

Lê do usuário as informações necessárias para criar uma nova página (data e texto). Cria uma página para armazenar as informações e a adiciona ao diário do usuário. O índice da página deve ser controlado automaticamente. (**se a última página tem o índice 2, a nova página tem índice 3 e assim por diante.**)

**diario \*rasgaPagina(int indice,diario \*d);**

Exclui do diário a página que contém o índice passado como parâmetro.

**O trabalho pode ser realizado em trios e deverá ser apresentado na última semana de aulas. Cópias serão desconsideradas.**

**Em caso de dúvidas, estarei presente na sala 387, Anexo B (acima do GMICRO) nos seguintes horários:**

**Segunda-Feira: 15:00-18:00**

**Terça-Feira: 09:00 - 15:30**

**Quinta-Feira: 14:30 - 18:00**

**Sexta-Feira: 09: 13:00**