UNIVERSIDADE TECNOLÓGICA DO PARANÁ
ENGENHARIA DE SOFTWARE

DOUGLAS WENDER
LUCAS TOMASI
NISIO OLIVEIRA

PROJETO EM C: GERAR FILAS
GERENCIADOR DE FILAS PARA BANCOS

2018

O nome da aplicação

Como nosso sistema é para um gerenciamento de filas em consultas ou bancos, colocamos o nome de "*Gerar Filas"*.

Os integrantes

O projeto foi realizado com 3 alunos do terceiro período do curso de Engenharia de Software. Sendo eles, Douglas Wender, Lucas Tomasi, Nisio Oliveira

Propósito e Modo de uso

O propósito da aplicação é intuitivo, gerar a senha e chamar o próximo da fila, através de arquivos usando a linguagem de programação C. O seu uso é básico, sendo que em uma aplicação o usuário que deseja entrar na fila escolhe qual fila deseja e isso gera um registro em um arquivo de dados que proporciona para a parte da aplicação que chama o próximo da fila que está no arquivo de texto. E o usuário responsável por chamar essa fila usa a segunda aplicação, que contém o mesmo arquivo como "base de dados" chama o primeiro da fila.

Para conseguir utilizar o software é necessário criar o arquivo em um diretório especificado no código, pois no início da função main, ele pega o arquivo e conta qual é a próxima senha a ser inscrita.

Divisão das Tarefas

Nisio Oliveira: Idealizador e supervisor, testador e gerente do projeto.

Lucas Tomasi: Programador da parte de chamar as senhas.

Douglas Wender: Programador da parte de gerar senhas e analista dos códigos.

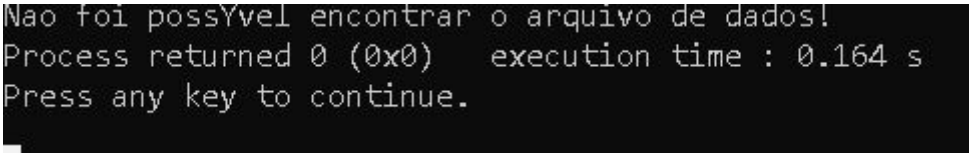
Conteúdos da Disciplina:

- TAD;
- Arquivos;
- Listas;
- Filas;

A implementação

Foi iniciada no meio do semestre, no fim de abril com o aprendizado do conteúdo de listas e posteriormente de filas, então pensamos na ideia de um gerenciador de filas para bancos. Usando o conteúdo aprendido em Algoritmos 2 de arquivos, conseguimos passar essa informação para o aplicativo. Segue abaixo exemplos de códigos que usamos para trabalhar com arquivos para salvar a inserção de novas senhas a fila.

Observação: Durante o desenvolvimento da primeira parte que é a de gerar o arquivo contendo os dados com as senhas, foi necessário que os arquivos já existissem para conseguir ler e descobrir de onde continuar as senhas, caso contrário ele exibirá uma mensagem de erro logo ao executar o programa.



Exemplo de execução do programa caso não tenha algum dos arquivos de texto.

Quanto a parte de arquivos não fizemos nada de novo, apenas criamos um TAD, e o chamamos de FileManager para fácil utilização de arquivos juntamente com a fila dinâmica.

```
FILE* AbreArquivoNormal(char modo){

    FILE *file;

    switch(modo){
        case 'w':
            file = fopen("C:\\Users\\dougl\\Documents\\senhasNormal.txt", "wt"); //ATENÇÃO, RENOMEAR O USUÁRIO ALI DE ACORDO COM O SEU NOME DE USUÁRIO DO WINDOWS.
            break;
        case 'r':
            file = fopen("C:\\Users\\dougl\\Documents\\senhasNormal.txt", "rt");
            break;
        case 'a':
            file = fopen("C:\\Users\\dougl\\Documents\\senhasNormal.txt", "a");
            break;
    }
    if(file == NULL){
        printf("Nao foi possivel encontrar o arquivo de dados Normal!");

    }
    return file;
}
```

Exemplo de código para abrir um arquivo. txt em C

```c
void CadastraNormal(int senha, char canal){

    FILE *file;

    file = AbreArquivoNormal('a');

    fprintf(file, "%d\t%c\n", senha, canal);

    FechaArquivo(file);
}
```

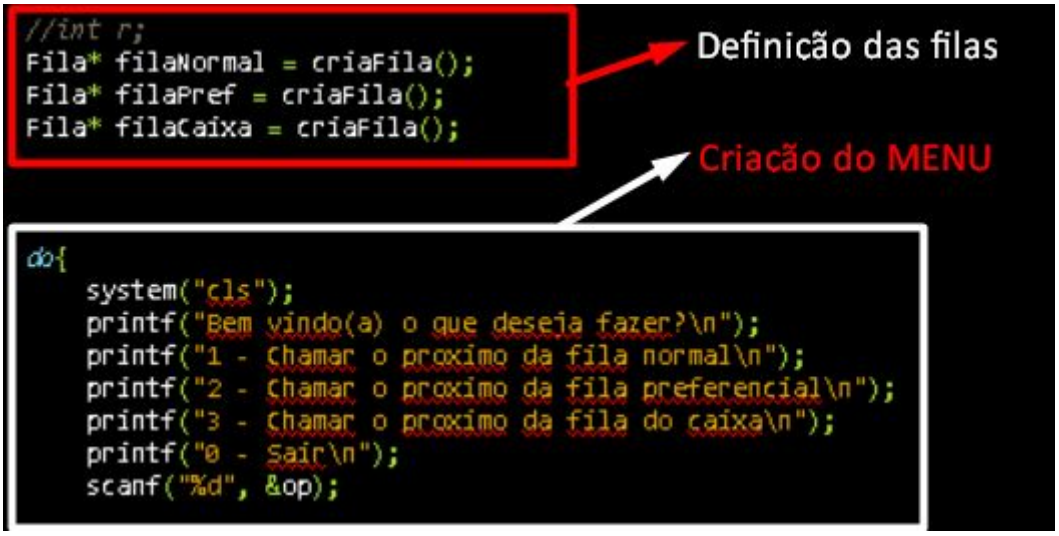
Exemplo de código para cadastrar nova senha no arquivo .txt

Depois na parte de chamar essas senhas, carregamos o arquivo de texto em outro programa que percorre esse arquivo contendo as senhas e os carrega em uma fila dinamicamente alocada.

```c
struct senha{
    int numero;
    char canal;
};


struct elemento{
    struct elemento *ant;
    struct elemento *prox;
    struct senha dados;
};

typedef struct elemento Elem;

struct fila{
    //char nome[30];
    struct elemento* inicio;
    struct elemento* fim;
    int qtd;
};
```

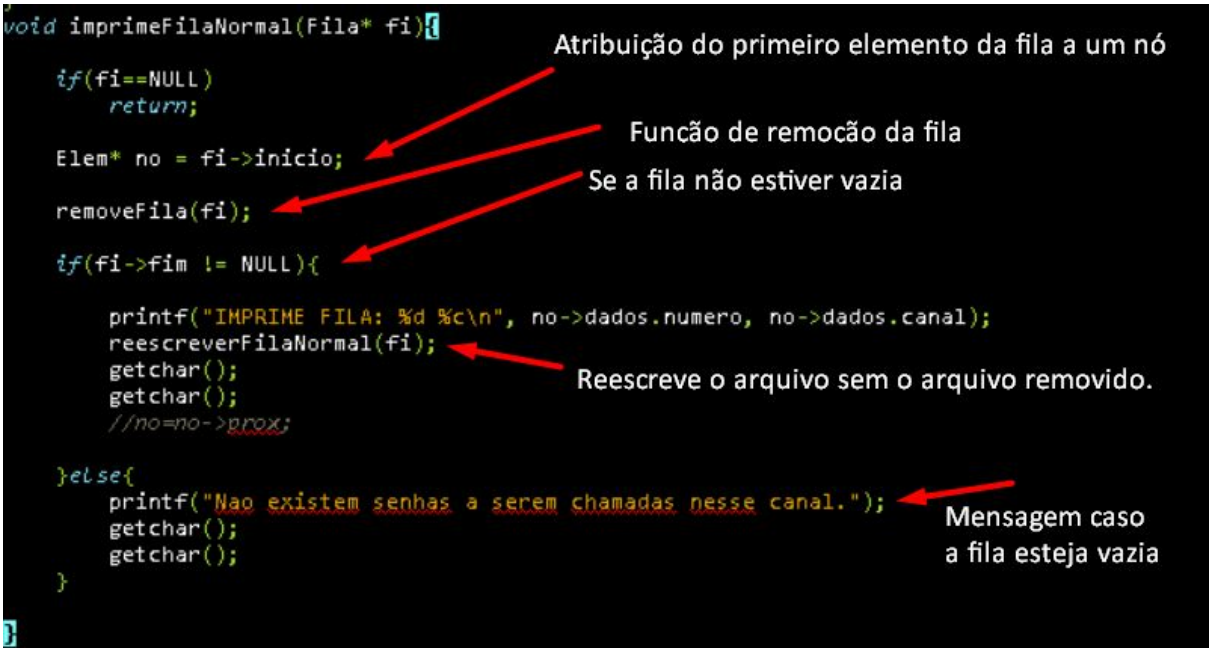
Estruturas criadas para formar uma fila.

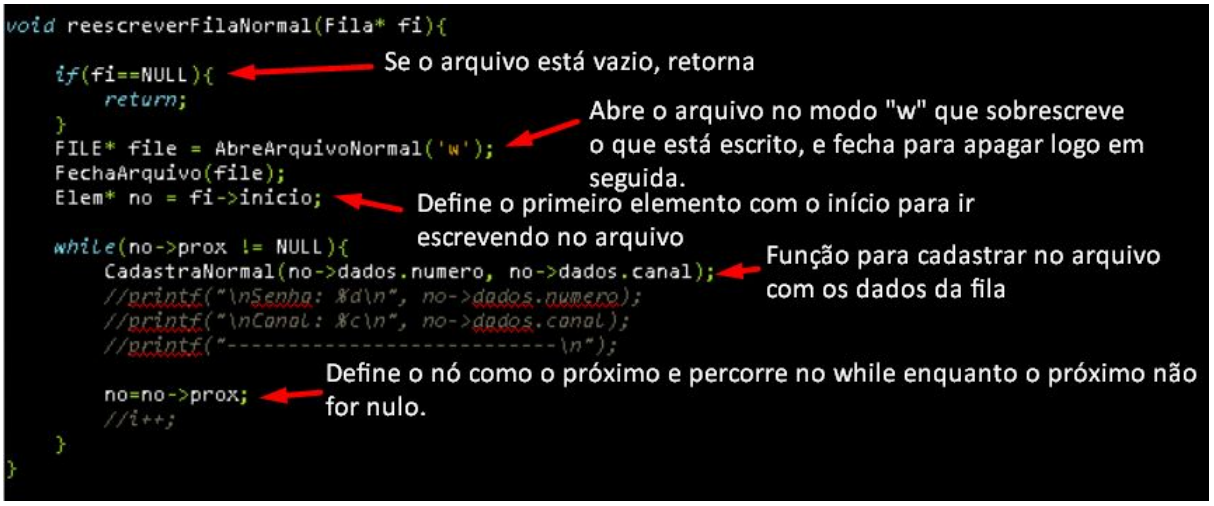
Começamos a parte de “Pegar” essas senhas armazenadas em um arquivo de texto.



Começo do programa de chamar novas senhas.

Com esse programa, precisamos carregar as senhas em filas alocando dinamicamente. A seguir, vemos uma função que imprime a fila e reescrever o arquivo logo em seguida.





Análise de complexidade

Conforme proposto em sala de aula, para que fosse feito o cálculo de análise de complexidade de uma função, podemos visualizar o resultado na imagem a seguir:

Análise de complexidade calculado.

Referências

- Conteúdo ministrado em aulas e slides;
- https://www.scriptbrasil.com.br/forum/topic/179408-banco-de-dados-em-txt-usando-a-linguagem-c/
- https://www.inf.pucrs.br/~pinho/LaproI/Arquivos/Arquivos.htm
- http://academyprogramming.blogspot.com.br/2015/06/crud-em-c-usando-arquivo.html
- http://web.archive.org/web/20071012105210/http://equipe.nce.ufrj.br/adriano/c/apostila/lista.htm#listaenc
- https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.bpxbd00/fopen.htm