
**Técnicas de Programação**

*[AN32F-IF62C]*

**PROFESSOR**

*Diogo Cezar Teixeira Batista*

# TRABALHO - 02

# SÉRIES ENCADEADAS

Cornélio Procópio

2017

# 1 INSTRUÇÕES

- Este trabalho deverá ser entregue em um arquivo único compactado no formato .zip ou .rar;
- Dever ser entregue em um único arquivo: LINKED_SERIES.c;
- O trabalho é individual;
- O arquivo fonte deve ter como comentário o nome do aluno, da turma e o que o exercício faz;
- Cada aluno deverá explicar o exercício resolvido para o professor;
- Trabalho iguais serão anulados;
- Não se esqueça de organizar, e identar o seu código;
- Utilize nomes significativos para as suas variáveis;
- Escreva as instruções para o usuário do programa que está criando;

# 2 EXERCÍCIOS

*E ai? Você sabe quais séries e filmes você já assistiu*?

Bom... se você possui uma conta ou perfil no netflix, isso é possível de ser visto.

Basta acessar o link: https://www.netflix.com/viewingactivity

Inclusive foi dessa lista que capturei o banco de dados que utilizei como exemplo para este exercício. Fiquem a vontade para gerar o seu próprio banco de dados e testar com o trabalho.

Mas vamos ao que interessa!

Neste exercício, vocês vão utilizar os conhecimentos adquiridos com:

- Alocação Dinâmica;
- Manipulação de Arquivos;
- Listas Encadeadas;
- Recursividade;

# Prontos?

Crie uma aplicação para gerenciar tudo o que você assiste. A estrutura para este exemplo é simples. Siga a seguinte definição:

```
typedef struct Item{
      char[10] date;
      char[100] name;
      struct Item* next;
} Item;
```

Nós queremos saber apenas a data e o título do que foi assistido.

Note que a estrutura tem um ponteiro, que aponta para um próximo elemento.

Deve-se criar uma estrutura de manipulação de lista encadeada para fazer todo o gerenciamento. Isso incluir as seguintes operações:

1. `start()` → é a função que inicia uma nova lista, retorna apenas `NULL` para um ponteiro;
2. `insert(Item * list, char * date, char * name)` → é a função que insere um novo item. Nessa função, deve-se criar um novo ítem (com `malloc` ou `calloc`), atribuir os valores (`date` e `name`), fazer o novo elemento apontar para a lista (exemplo: `new->next = list`) e retornar o novo elemento (`return new`);
3. `print(Item * list)` → é a função que vai percorrer toda a lista e imprimir cada um dos elementos, neste caso deve-se implementar uma versão iterativa, sem a utilização de recursividade;
4. `printRecursive(Item * list)` → é a função que vai percorrer toda a lista, mas de forma recursiva, ou seja, em determinado momento desta função, a própria função `printRecursive` deverá ser chamada. Não se esqueça de uma condição de parada;
5. `printRecursiveInverted(Item * list)` → é a função que vai percorrer toda a lista, de forma recursiva, mas dessa vez imprimindo a lista de trás para frente.
6. `search(Item * list, char * date)` → é a função que exibirá todos os itens em uma data específica;
7. `destroy(Item* list)` → é a função que irá limpar a lista em questão;
8. `load(Item * list)` → é a função que irá carregar os items a partir de um arquivo;

9. `save(Item * list)` → é a função que irá salvar os itens em um arquivo.

OBSERVAÇÃO → fica a seu critério a utilização da lista como uma variável global ou um parâmetro entre as funções.

A inserção de elementos nesta lista poderá ocorrer de duas formas:

1. Manualmente, no qual informa-se ítem por ítem a ser cadastrado;
2. Por arquivo, no qual se informa o nome do arquivo fonte, e todas as informações serão carregadas para a memória do programa.

Crie um menu com as seguintes opções para o usuário:

```
Olá, bem-vindo ao sistema de gerência de séries e filmes!

O que você deseja fazer?

[01] - Inserir um item manualmente;
[02] - Carregar um arquivo com os meus itens;
[03] - Listar os itens de forma iterativa;
[04] - Listar as itens de forma recursiva;
[05] - Listar as itens de forma recursiva invertida;
[06] - Limpar a lista em memória;
[07] - Salvar lista em arquivo texto;
[08] - Contar items em memória;
[09] - Procurar item por data;
[10] - Sair
```

Ao escolher a opção `[01] - Inserir um item manualmente` deve-se perguntar ao usuário uma nova data e nome e então criar uma lista (se não houver) e utilizar a função `insert()` para inserir um novo ítem;

Ao escolher a opção `[02] - Carregar um arquivo com os meus itens` o programa deverá perguntar ao usuário o nome do arquivo a ser carregado. Se o arquivo existir, deve-se então, para cada ocorrência do arquivo, utilizar a função de `insert()` para cadastrar um novo filme na lista em memória.

IMPORTANTE → sempre que um arquivo for carregado, deve-se EXCLUIR a lista anterior. Então, se existiam ítens cadastrados manualmente, eles serão perdidos. Mas, depois de carregar a lista, é possível inserir novos itens manualmente.

Existe um formato definido para a leitura de dados, segue um exemplo do arquivo que deverá ser lido:

```
18/09/2017;Rick and Morty: Temporada 2: Olha quem está expurgando
18/09/2017;Rick and Morty: Temporada 2: TV interdimensional
17/09/2017;Bates Motel: Temporada 2: Presunção de inocência
16/09/2017;Bates Motel: Temporada 2: Mergulho
```

Note que a separação entre a data e o título é feita com um `;` e note também que no final de cada linha temos um `\n`

Esse formato foi escolhido por se tratar de um formato de arquivo .csv que pode ser aberto em qualquer editor de planilhas;

Ao escolher a opção `[03] - Listar os itens de forma iterativa` deve-se percorrer a lista e exibir na tela os ítens cadastrados em memória mas de forma iterativa (sem a utilização da recursividade).

Ao escolher a opção `[04] - Listar as itens de forma recursiva` deve-se percorrer a lista e exibir na tela os ítens cadastrados em memória, mas de forma recursiva. Note que, como a lista sempre aponta para o próximo, a lista por definição ficará com a ordem invertida, e por esse motivos temos a opção `[05] - Listar as itens de forma recursiva invertida.`

Ao escolher a opção `[06] - Limpar a lista em memória` deve-se limpar a lista em memória. Neste ponto, podemos assumir apenas que o ponteiro para o primeiro ítem receberá `NULL`, mas como um desafio, você pode percorrer a lista e liberar a memória alocada para cada um dos elementos.

A opção `[07] - Salvar lista em arquivo texto` deverá salvar a lista que está em memória em um arquivo exatamente igual ao padrão que foi definido anteriormente. Mas antes, deve-se perguntar o nome do arquivo a ser salvo.

A opção `[08] - Contar items em memória` deve-se percorrer a lista de ítens e incrementar um contador, para verificar quantos itens estão em memória.

A opção `[09] - Procurar item por data` deverá perguntar uma data para o usuário. Com essa data, deve-se fazer uma busca em todos os ítens e imprimir os correspondentes.