UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Controle de Registros em Arquivo

Projeto parte 2

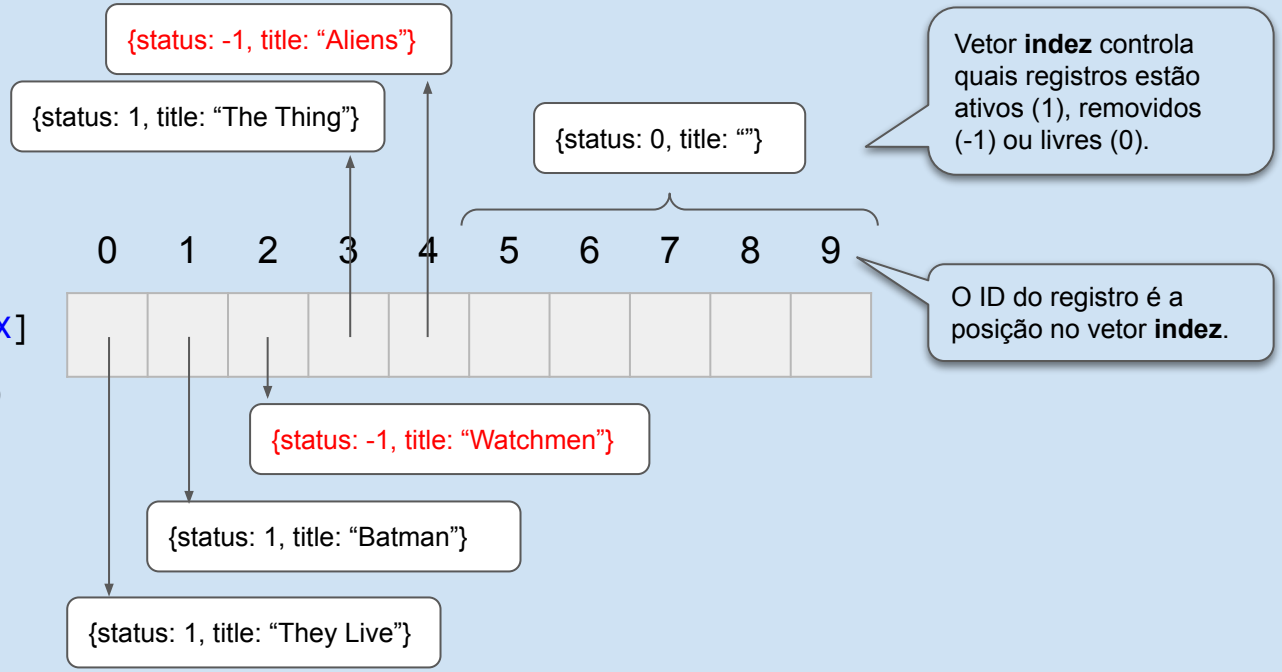
Marcos Silvano Almeida
Departamento de Computação
UTFPR Campo Mourão

Projeto parte 2: Controle de Registros

- Objetivo:
  - Implementar app gerenciamento de registros de dados com persistência em arquivo

- Características
  - Trabalho em **dupla** OU **sozinho**.
  - Interface do usuário (UI) em modo texto com menus de navegação
  - Escolher um tema para os registros
    - Ao menos 4 campos + id
  - Dados organizados em arquivos binários de structs
  - Como no Projeto 1, vetor de índice é utilizado para buscar e controlar registros
  - Permitido utilizar lib <string.h>
  - Trabalho será recuperação do primeiro
    - T1 = MAX(T1, (T1+T2)/2);

MEMÓRIA+ARQUIVO
Vetor de Índices




```
struct Entry indez[MAX]
```

Persistido em arquivo binário após cada operação:
index.idx

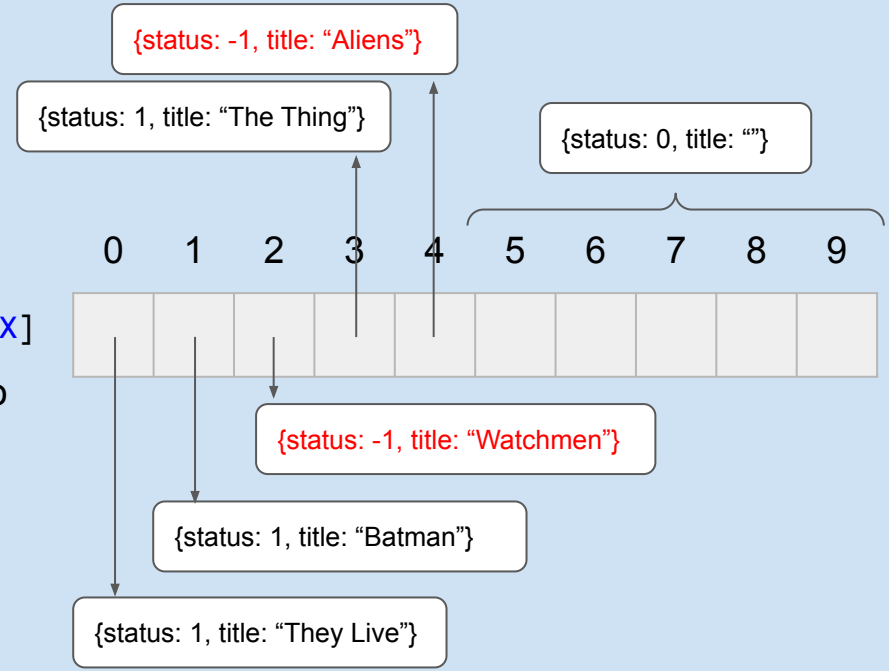
ARQUIVO
Registros

Arquivo **binário** de registros:
records.data

```
{0, “They Live”, 1988, “John Carpenter”, “Sci-Fi”}
{1, “Batman”, 1989, “Tim Burton”, “Super Hero”}
{2, “Watchmen”, 2009, “Zack Snider”, “”Super Hero”}
{3, “The Thing”, 1982, “John Carpenter”, “Horror”}
{4, “Aliens”, 1986, “James Cameron”, “Sci-Fi”}
```

O arquivo **record.data** armazena os dados de todos os registros cadastrados.

MEMÓRIA+ARQUIVO
Vetor de Índices




```
struct Entry {
    int status;
    char title[31];
};
```

Campo **title** replicado no **indez** permite busca por **título do filme**.
Campo status informa:
1 - registro ativo
0 - posição livre
-1 - registro removido

`struct Entry indez[MAX]`

Persistido em arquivo binário após cada operação:
index.idx

ARQUIVO
Registros

Arquivo **binário** de registros:
records.data

```
{0, “They Live”, 1988, “John Carpenter”, “Sci-Fi”}
{1, “Batman”, 1989, “Tim Burton”, “Super Hero”}
{2, “Watchmen”, 2009, “Zack Snider”, “”Super Hero”}
{3, “The Thing”, 1982, “John Carpenter”, “Horror”}
{4, “Aliens”, 1986, “James Cameron”, “Sci-Fi”}
```

```
struct Movie {
    int id;
    char title[31];
    int year;
    char director[21];
    char genrer[21];
};
```

Questões gerais

1. O registro de filme (Movie) é apenas um exemplo
   a. Você deve fazer o seu próprio registro, com tema e campos diferentes
2. Operações do app: adicionar, consultar, remover, editar, mapa de registros e exibir lixeira & restaurar registros.
3. O vetor **indez** deve permanecer na memória
   a. Deve ser persistido por completo em arquivo binário (**index.idx**), após cada operação (**item 2**) que realize modificações nos registros
4. Os registros devem ser armazenados em arquivo binário de struct (**records.data**)
   a. Cada operação (**item 2**) deve abrir o arquivo de registros, realizar as operações necessárias e fechá-lo.
5. **OBS:** Você **não deve**, em momento algum, trazer todos os registros do arquivo record.data para a memória, isto é, não deve existir um vetor de registros. Cada registro deve ser lido/escrito separadamente no arquivo.
   a. Os registros devem ser acessados da seguinte maneira: abrir o arquivo (**fopen**), posicionar o cursor (**fseek**), ler/escrever dados (**fread/fwrite**) e fechar (**fclose**).
6. Cada operação deve realizar validações, informando sucesso ou erros.

UTFPR

Protótipo de Interface

+

Simulação de Funcionamento

Menu Principal

```
========================================
 GERENCIAMENTO DE FILMES
========================================

MENU PRINCIPAL
1 - Adicionar filme          5 - Lixeira & Restaurar
2 - Consultar filmes         6 - Mapa de registros
3 - Remover filme            0 - Sair
4 - Alterar filme

Entre com uma opção: 1
```

Quando programa é iniciado, deve verificar existência dos arquivos:
- index.idx
- records.data

Se não existirem, cria ambos. Caso contrário, carrega o vetor em **index.idx** para a memória (**indez**)

Após cada operação ser concluída, programa deve retornar ao menu principal.

Validar opção (garantir que é válida, 0 à 6, e informar erros, caso ocorram)

1 - Adicionar filme

```
========================================
 MENU PRINCIPAL / Adicionar filme
========================================
Novo ID: [3]
Título: Batman
Ano: 1989
Diretor: Tim Burton
Gênero: Super Hero

Confirmar (S/N)? S

Novo registro adicionado.
Pressione uma tecla >
```

ID é automático (índice do vetor **indez**): Neste caso, mostra a próxima posição disponível no vetor **indez.**
Primeiro procura por posição nunca usada (0). Caso não encontre, procura por posição removida (-1).

Confirma operação.

ADICIONAR REGISTRO

1. Obter próxima posição (se) disponível no vetor **indez** e exibe na tela.
2. Obter os dados do registro do usuário (scanf);
3. Confirmar operação;
4. Continuar se confirmação positiva;
5. Preencher os campos de variável struct **Movie** com os dados informados;
6. Atualizar **Entry** no vetor **indez** para posição ocupada (**status** → 1);
7. Escrever (fwrite) registro em **records.data**, na posição correta.
8. Escrever **indez** inteiro em **index.idx**.
9. Informar status e aguardar tecla.
10. Retornar ao Menu Principal.

2 - Consultar filme

A busca só poderá ser aplicada ao campo que estiver em **Entry**, no vetor **indez** (em memória).

CONSULTAR REGISTRO

1. Obter a string para busca (scanf);
2. Se string igual a * (asterisco), exibir todos os registros de **status** 1 em **indez**.
3. Caso contrário, realizar busca da string digitada no campo title de cada **Entry** de **indez** que possua **status** 1.
4. Cada **Entry** com resultado positivo para a busca deve ter o registro lido (fread) de **records.data** e exibido na tela.
5. Informar status e aguardar tecla.
6. Retornar ao Menu Principal.

```
==============================
 MENU PRINCIPAL / Consultar filme
==============================
Busca por título: man

Filmes encontrados com 'man':

ID      Título              Ano     Diretor             Gênero
01      Batman              1989    Tim Burton          Super Hero
03      Demolition Man      1993    Marco Brambilla     Action
14      Hollow Man          2000    Paul Verhoeven      Thriller

Pressione uma tecla >
```

3 - Remover filme

O processo para remover um filme existente utiliza o processo de:
- Consultar filme

REMOVER REGISTRO

1. Realizar operação de consulta de filme;
2. Obter ID do registro a remover (scanf);
3. Confirmar operação;
4. Continuar se confirmação positiva;
5. Verificar se ID é válido (existe e contém **status** 1, “ativo”);
6. Modificar **status** para -1 (lixeira) da **Entry** em **indez** que possua o ID informado;
7. Escrever **indez** inteiro em **index.idx**.
8. Informar status e aguardar tecla.
9. Retornar ao Menu Principal.

```
=======================================
 MENU PRINCIPAL / Editar filme
=======================================
Busca por título: man

Filmes encontrados com 'man':

ID      Título              Ano     Diretor             Gênero
01      Batman              1989    Tim Burton          Super Hero
03      Demolition Man      1993    Marco Brambilla     Action
14      Hollow Man          2000    Paul Verhoeven      Thriller

ID a remover: 3

Confirmar (S/N)? S
Registro na lixeira.
```

A posição removida em **indez** é marcada como status “lixeira” (-1). O registro no arquivo **records.data** são mantidos, caso venha a ser restaurado.

4 - Alterar filme

O processo para alterar um filme existente utiliza o processo de:
- Consultar filme

```
=======================================
 MENU PRINCIPAL / Alterar filme
=======================================
Busca por título: bat
Filmes encontrados com 'bat':
ID      Título              Ano     Diretor
01      Batman              1989    Tim Burton

ID a editar: 1
Título: Batman Returns
Ano: 1992
Diretor: Tim Burton
Gênero: Super Hero

Confirmar (S/N)? S
Registro alterado. Pressione uma tecla >
```

ALTERAR REGISTRO

1. Realizar operação de consulta de filme;
2. Obter ID do registro a editar (scanf);
3. Verificar se ID é válido (existe e contém **status** 1, "ativo");
4. Obter dados dos campos (scanf).
5. Confirmar operação;
6. Continuar se confirmação positiva;
7. Substituir registro na posição do ID informado, no arquivo **records.data**.
8. Substituir campo **title** na **Entry** da posição correspondente em **indez**.
9. Escrever **indez** inteiro em **index.idx**.
10. Informar status e aguardar tecla.
11. Retornar ao Menu Principal.

5 - Lixeira & Restaurar

```
=========================================
 MENU PRINCIPAL / Lixeira & Restaurar
=========================================
Filmes encontrados na lixeira:

ID      Título      Ano     Diretor          Gên
02      Aliens      1986    James Cameron    Sci-Fi
03      Watchmen    1986    Zack Snider      Super Hero

ID a restaurar: 2

Confirmar (S/N)? S
Registro restaurado.
```

RESTAURAR REGISTRO

1. Exibir todos os registros de status -1 em indez.
2. Obter ID do registro a editar (scanf);
3. Verificar se ID é válido (existe e contém **status** -1, “lixeria”);
4. Confirmar operação;
5. Continuar se confirmação positiva;
6. Modificar **status** para 1 (“ativo”) da **Entry** em **indez** que possua o ID informado;
7. Escrever **indez** inteiro em **index.idx**.
8. Informar status e aguardar tecla.
9. Retornar ao Menu Principal.

Exibe os dados dos registros marcados com **satus** -1 em **indez**. Caso **restaurado**, retorna **status** do registro para 1 em **indez**..

6 - Mapa de Registros

Exibir o conteúdo de **indez** e o arquivo **records.data**.

```
=======================================
 MENU PRINCIPAL / Mapa de Registros
=======================================
INDEZ: 04/10 índices (arquivo INDEX.IDX)
0:{01, They Live} 1:{01, Batman   } 2:{-1, Watchmen} 3:{-1,Aliens   }
4:{01, The Thing} 5:{00, -------  } 6:{00, ------- } 7:{00, ------- }
8:{00, -------  } 9:{00, -------  }

Arquivo RECORDS.DATA
{0, They Live, 1988, John Carpenter, Sci-Fi}
{1, Batman   , 1989, Tim Burton    , Super Hero}
{2, Watchmen , 2009, Zack Snider   , Super Hero}     [LIXEIRA]
{3, Aliens   , 1986, James Cameron , Sci-Fi}         [LIXEIRA]
{4, The Thing, 1982, John Carpenter, Horror}
```

Cronograma de execução

Datas e atividades

Semana 1

20/10 PAluno: dúvidas gerais (todos)

21/10 Acompanhamento individual: 20” por grupo

Necessário informar grupos e definir horário na quarta-feira!

Semana 2

27/10 PAluno: dúvidas gerais (todos)

28/10 Acompanhamento individual: 20” por grupo

Semana 3

03/11 PAluno: dúvidas gerais (todos)

04/11 Acompanhamento individual: 20” por grupo

Semana 4

10/10 Apresentações: 30” por grupo

11/10 Apresentações: 30” por grupo

Avaliação do Projeto

Critérios de Avaliação do Projeto

- Qualidade e eficiência das soluções utilizadas
  - Algoritmos implementados
- Organização do código
  - Legibilidade do código (identação, identificadores, ...)
  - Documentação do código (comentários)
- Estrutura do código
  - Modularização em funções para organização e evitar duplicações
- Acabamento da interface em texto
- Divisão das atividades na dupla
  - Divisão balanceada
- Conhecimento sobre o programa implementado
  - Apresentação síncrona com questionamentos do professor
  - Cada aluno é responsável por sua parte

Questões sobre Avaliação

- Cópias: Qualquer tipo de cópia (trabalhos de colegas, internet, etc) anulará imediatamente o trabalho
  - Seja por porções de código ou pelo trabalho completo
  - Projeto deve ser de autoria exclusiva dos integrantes da equipe
  - Cada aluno deve necessariamente conhecer o seu código
    - Estruturas utilizadas
    - Algoritmos implementados

- O objetivo não é a nota, mas o processo pelo qual vocês devem passar para obtê-la. Desta forma, vocês não devem “entregar” um programa: vocês devem **construir um programa**.

- Entrega:
  - ZIP com código fonte pelo Moodle
  - Apresentação síncrona + Responder questionamentos do professor

Ratificando, conforme Plano da ADNP

Sobre apresentação

- “A apresentação de cada projeto deverá ser realizada de forma síncrona e consiste em expor as funcionalidades e sua implementação. O professor fará perguntas para atestar a qualidade da implementação, o cumprimento dos requisitos e a autoria do programa pelos alunos. Cada aluno da equipe deverá ser responsável por uma parte do programa.”

Sobre autoria do trabalho

- “Os projetos devem ser de autoria exclusiva dos alunos da equipe. Se o professor detectar cópia parcial ou total de código(s) de terceiro(s) ou perceber que os alunos não são os autores do código apresentado, a nota do trabalho será anulada.”