

# Trabalho Prático II: Manipulação e Reorganização de Arquivos de Dados

Disciplina: Algoritmos e Estruturas de Dados II

Professor: Rafael Alexandre

## 1. Introdução

Durante a vida útil de um arquivo de dados, diversas operações de manutenção — como inserção, exclusão e atualização de registros — alteram a forma como os dados ocupam fisicamente os blocos de armazenamento. Com o tempo, essas operações tendem a gerar **fragmentação**, desperdício de espaço e perda de desempenho no acesso às informações.

Esses fenômenos estão diretamente relacionados à **gestão de espaço livre**, **reorganização física de arquivos** e aos conceitos de **fragmentação interna e externa**:

- **Fragmentação interna:** ocorre quando parte do espaço de um bloco permanece inutilizada devido ao tamanho fixo dos registros ou ao desalinhamento entre o tamanho do registro e o tamanho do bloco;
- **Fragmentação externa:** surge quando há pequenos espaços livres espalhados entre blocos, resultantes de exclusões ou realocações, que dificultam o armazenamento de novos registros contíguos;
- **Gestão de espaço livre:** envolve o controle das áreas disponíveis para reutilização, garantindo que novos registros possam ser alocados de forma eficiente;
- **Reorganização física:** consiste na compactação dos dados e redistribuição dos registros nos blocos, com o objetivo de reduzir a fragmentação e melhorar a eficiência de armazenamento.

Este trabalho propõe a implementação dessas operações e mecanismos, dando continuidade ao **Trabalho Prático I**, no qual foi desenvolvida a simulação de persistência de registros de alunos em blocos de tamanho fixo ou variável. Agora, o foco é a **manipulação dinâmica dos registros** e a **manutenção eficiente da estrutura física do arquivo**.

## 2. Objetivos

- Implementar operações de inserção, edição e exclusão de registros em um arquivo estruturado em blocos;

- Gerenciar o reaproveitamento de espaços livres oriundos de registros excluídos;
- Tratar casos de atualização que alterem o tamanho dos registros, considerando restrições de espaço;
- Permitir a inclusão de novos registros no último bloco ou em blocos parcialmente ocupados;
- Implementar uma rotina de **reorganização física** do arquivo, eliminando fragmentações e consolidando dados;
- Gerar relatórios comparativos de eficiência antes e depois da reorganização.

### 3. Especificação do Problema

O arquivo `alunos.dat`, criado no **Trabalho Prático I**, será utilizado como base. O programa deverá oferecer, via menu ou interface de linha de comando, as seguintes operações:

#### 1. Inserção de novo registro:

- Verificar se há espaço livre em algum bloco devido a exclusões anteriores;
- Reutilizar blocos parcialmente livres sempre que possível;
- Caso não haja espaço disponível, gravar o registro no último bloco ou criar um novo bloco.

#### 2. Edição de registro existente:

- Permitir a alteração de qualquer campo de um registro;
- Caso o novo tamanho seja igual ou menor, substituir o registro no mesmo local;
- Caso o novo tamanho exceda o espaço disponível, realocar o registro para outro bloco com espaço suficiente;
- Atualizar referências ou marcadores, quando necessário (no caso de registros espalhados).

#### 3. Exclusão de registro:

- Implementar **remoção lógica**, marcando o registro como excluído;
- Registrar o espaço liberado como área reaproveitável para futuras inserções.

#### 4. Reorganização do arquivo:

- Compactar os blocos, movendo registros de forma a eliminar fragmentações internas e externas;

- Reaproveitar espaços subutilizados e remover blocos vazios;
- Gerar um novo arquivo reorganizado (por exemplo, `alunos_reorg.dat`);
- Produzir relatório com:
  - Número de blocos antes e depois;
  - Ocupação média e eficiência antes e depois;
  - Ganho percentual de eficiência obtido.

## 4. Regras e Considerações

- Todas as operações devem respeitar o limite de bytes por bloco definido pelo usuário;
- O programa deve manter metadados internos para:
  - Localizar blocos com espaço livre;
  - Identificar registros removidos (para reaproveitamento futuro);
  - Mapear a posição física de cada registro (bloco e deslocamento);
- A integridade do arquivo deve ser preservada entre execuções (as operações devem persistir no arquivo `.DAT`);
- O aluno poderá utilizar estruturas auxiliares em memória (como listas ou índices simples) para apoiar as operações.

## 5. Saídas Esperadas

Após as operações, o programa deverá apresentar relatórios e estatísticas como:

- Mapa de ocupação dos blocos (bloco, bytes ocupados e livres);
- Quantidade de registros ativos e excluídos;
- Tamanho total do arquivo e eficiência percentual;
- Relatório de reorganização conforme o exemplo:

```
===== RELATÓRIO DE REORGANIZAÇÃO =====
```

```
Antes:
```

```
Blocos: 12
```

```
Ocupação média: 68,2%
```

```
Eficiência total: 71,5%
```

Depois:

Blocos: 8

Ocupação média: 94,1%

Eficiência total: 92,7%

Ganho de eficiência: +21,2%

Blocos liberados: 4

=====

## 6. Roteiro de Implementação (Sugerido)

1. Revisar a estrutura de armazenamento e metadados do Trabalho I;
2. Implementar marcação de exclusão e controle de áreas livres;
3. Implementar inserção reutilizando espaços liberados sempre que possível;
4. Implementar atualização com verificação de restrições de espaço;
5. Criar rotina de reorganização e geração de novo arquivo otimizado;
6. Calcular e exibir estatísticas comparativas antes e depois da reorganização;
7. Testar os casos de uso e documentar os resultados obtidos.

## 7. Funcionalidades Adicionais (Opcional)

- Implementar índice de busca por matrícula ou outro campo;
- Exportar registros ativos para CSV;
- Registrar logs de operações (inserções, exclusões, reorganizações);
- Gerar visualização gráfica da ocupação dos blocos e fragmentação.

## 8. Critérios de Avaliação

Critério	Peso
Implementação das operações de inserção, edição e exclusão	30%
Tratamento de espaços livres e realocação de registros	20%
Implementação e resultados da reorganização física	20%
Cálculo e apresentação dos relatórios de eficiência	15%
Clareza e organização do código	10%
Relatório explicativo (decisões e análise de resultados)	5%
<b>Total</b>	<b>100%</b>

## 9. Entrega

O trabalho deverá ser entregue em repositório (GitHub, GitLab ou similar), contendo:

- Código-fonte completo e documentado;
- Arquivos .DAT de teste (antes e depois da reorganização);
- Relatório em PDF contendo:
  - Descrição das estratégias e estruturas adotadas;
  - Casos de teste e resultados;
  - Análise dos ganhos de eficiência obtidos.