UFMG/ICEx/DCC — Algoritmos e Estrutura de Dados 3

**Trabalho Prático 4**

Bacharelado — 1º Semestre de 2012

Este trabalho prático tem por objetivo exercitar a implementação de um algoritmo para ordenação externa, utilizado quando a quantidade de registros a serem ordenados não cabem na memória principal.

# 1 Definição do Problema

Dispositivos móveis, em sua maioria, possuem limitações de memória e processamento. Nesse trabalho prático, simularemos o problema da ordenação externa através da imposição de restrições em relação ao tamanho da memória principal para ordenar alfabeticamente registros que compõem a lista de contatos de um celular. Basicamente, a ordenação externa consiste em ordenar um conjunto de registros que possui tamanho maior que a memória principal disponível. Os métodos de ordenação externa consistem em:

1. Quebrar o arquivo em blocos de tamanho igual ou inferior ao da memória principal disponível.
2. Ordenar cada bloco na memória principal.
3. Intercalar os blocos ordenados através de várias leituras a esses blocos.
4. A cada leitura serão criados blocos ordenados cada vez maiores, até que todo o conjunto de arquivos esteja ordenado.

Para realizar essa tarefa, o programa receberá como entrada as seguintes informações:

- $n$: número de contatos a serem ordenados.
- $m$: tamanho do *buffer* a ser criado (correspondente ao tamanho limitado da memória principal) para ordenar os blocos.
- Lista dos $n$ contatos do celular.

O programa deverá alocar um único *buffer* dinamicamente para ordenar $m$ registros por vez. Sendo assim, serão criados $n/m$ blocos a serem ordenados. O aluno deverá ordenar esses blocos e produzir um arquivo de saída intermediário para cada bloco contendo os contatos ordenados desse bloco. Quando os $n/m$ blocos estiverem ordenados, deverá ser criado um único arquivo de saída contendo **todos** os contatos ordenados, sendo que **a intercalação dos contatos deverá utilizar um *heap***. O tamanho do *heap* também não deve ultrapassar o tamanho do *buffer*.

Deverá ser apresentada uma avaliação experimental (através de grafos) que compare o tempo de execução do programa – através do uso da função *gettimeofday()* – e o tamanho do *buffer* disponibilizado.

**OBSERVAÇÃO**: Os monitores irão verificar na correção desse trabalho prático se o aluno realmente implementou um algoritmo de ordenação externa com a memória principal limitada. Para isso, utilizarão o comando *ulimit* que impõe limitações quanto à memória disponível para um processo.

## 1.1 Entrada e Saída

O arquivo executável deve ser chamado de *tp4* e deve receber como parâmetro o arquivo de entrada *input.txt* e o arquivo de saída *output.txt*, conforme demonstrado a seguir:

```
./tp4 -i input.txt -o output.txt
```

O arquivo *input.txt* deve conter as seguintes informações, na ordem indicada abaixo:

1. $n$: número de contatos.

2. $m$: tamanho do *buffer*.

3. contatos do telefone ($n$ linhas com: nome do contato e número do respectivo telefone, separados pelo character #).

**Exemplo de entrada (arquivo *input.txt*):**

```
10
4
Amanda Pires#87345530
Zilda Soares#33739547
Joao Cunha#74856723
Beatriz Salles#94742045
Diego Sousa#89981441
Marta Maria#85642375
Antonio Luis#35615423
Cecilia Moura#99771265
Nilton Moraes#35624847
Tadeu Silva#78651294
```

Cada bloco de contatos ordenado deverá ser impresso em um arquivo intermediário **output_i.txt**, onde $i$ é o índice do bloco de contatos. Sendo assim, o programa deverá alocar um único *buffer* dinamicamente de tamanho $m$, ordenar os $m$ registros de cada bloco e produzir um arquivo de saída (**output_i.txt**) para cada bloco contendo os contatos ordenados desse bloco. Além disso, **todos** os contatos deverão ser impressos em ordem alfabética no arquivo *output.txt*.

**Exemplo de saída genérico (arquivos *output_i.txt* e *output.txt*):**

*output_1.txt*:

```
Amanda Pires#87345530
Beatriz Salles#94742045
Joao Cunha#74856723
Zilda Soares#33739547
```

*output_2.txt*:

```
Antonio Luis#35615423
Cecilia Moura#99771265
Diego Sousa#89981441
Marta Maria#85642375
```

*output_3.txt*:

```
Nilton Moraes#35624847
Tadeu Silva#78651294
```

*output.txt*:

```
Amanda Pires#87345530
Antonio Luis#35615423
Beatriz Salles#94742045
Cecilia Moura#99771265
Diego Sousa#89981441
Joao Cunha#74856723
Marta Maria#85642375
Nilton Moraes#35624847
Tadeu Silva#78651294
Zilda Soares#33739547
```

Entrada e saída padrão devem seguir rigorosamente o formato descrito. Instâncias distintas para o problema devem ser geradas pelo próprio aluno para testar e avaliar seu algoritmo.

# 2 O que deve ser entregue:

## 2.1 Documentação: deve abranger pelo menos os seguintes pontos

- Introdução do problema apresentado.
- Modelagem e solução do problema.
- Complexidade de tempo e espaço.
- Principais decisões de implementação.
- Análise experimental.
- A documentação **não** pode exceder 10 páginas.

## 2.2 Código:

- O código fonte do trabalho deve ser submetido para compilação e execução em ambiente Linux, tendo como padrão os computadores dos laboratórios de graduação do DCC;
- Deve ser **obrigatoriamente** escrito na linguagem C (trabalhos implementados em outras linguagens como C++/Java/Python e outras **não** serão aceitos);
- As estruturas de dados devem ser alocadas dinamicamente e o código deve ser modularizado (ou seja, dividido em múltiplos arquivos fonte e fazendo uso de arquivos cabeçalho - .h);
- O utilitário Make deve ser utilizado para compilar o programa **(o arquivo de makefile deve ser submetido juntamente com o código fonte)**;
- A saída deve ser impressa seguindo estritamente o formato da especificação, caso contrário o resultado será considerado errado;
- O arquivo executável deve ser chamado de **tp4**. Não serão aceitos outros nomes de executáveis além do mencionado;
- Faça seu código de forma legível.

## 2.3 Entrega:

- Data de entrega: 23/05/2012
- Submissão: a documentação e o código do trabalho devem ser submetidos ao *minha.ufmg*. Para isso, compacte os dois (formato *tp4_NomeSobrenome.zip*) e faça a submissão. Teste seu arquivo compactado antes de enviá-lo.
- Apenas a documentação deve ser entregue impressa na secretaria do DCC. Não coloque nos escaninhos dos professores, entregue para a secretária para que sua documentação seja colocada no envelope de AEDS3. A documentação impressa pode ser entregue no dia útil seguinte da submissão digital. **Trabalhos que não tiverem a documentação entregue na secretaria, dentro do prazo de entrega, receberao nota 0.**
- Será postada uma planilha no Moodle sobre a entrevista do trabalho, leia-a e siga as orientações para o agendamento da sua entrevista.
- Será adotado média harmônica entre a pontuação obtida na execução e na documentação do TP, o que implica em valor zero caso alguma das partes não seja apresentada.
- A política para desconto por atraso de entrega do trabalho prático considera a fórmula:

$$\frac{2^{d-1}}{0.32}\%$$

onde d é o atraso em dias úteis. Note que após 5 dias úteis, o trabalho não pode ser mais entregue.

## Observações

- O formato de submissão **DEVE** ser respeitado: *tp4_NomeSobrenome.zip*. Por favor, **NÃO** submetam trabalhos em outros formatos (.rar, .tar, .tar.gz). Certifique-se que seu nome completo foi incluído no nome do arquivo.
- **EXCLUA** o arquivo executável do arquivo compactado a ser submetido no *minha.ufmg*.
- **INCLUA** seu email na capa da documentação.

## Referências


*Projeto de Algoritmos com implementação em Pascal e C.* Nívio Ziviani - *http://www.dcc.ufmg.br/algoritmos/.*

*Introduction to Algorithms.* T. Cormen, C. Leiserson, R. Rivest, and C. Stein. MIT Press, 3rd edition, 2009.

*Introdução aos Fundamentos da Computação.* Newton Vieira. Pioneira Thomson Learning, 2006.