Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Estrutura de Dados II – BCC203
Prof. Guilherme Tavares de Assis – 2025/1

Trabalho Prático II – Ordenação Externa

Grupo de 4 a 6 alunos - Valor: 15,0 pontos - Data de entrega: 12/08/25

O objetivo deste trabalho consiste em um estudo mais profundo da complexidade de desempenho dos seguintes métodos de ordenação externa apresentados em sala de aula:

1. Intercalação balanceada de vários caminhos (2f fitas) utilizando, na etapa de geração dos blocos ordenados, qualquer método de ordenação interna apresentado em "Estrutura de Dados I".
2. Intercalação balanceada de vários caminhos (2f fitas) utilizando, na etapa de geração dos blocos ordenados, a técnica de seleção por substituição apresentada em "Estrutura de Dados II".
3. *Quicksort* externo.

A 1ª fase deste trabalho corresponde à implementação em C dos 3 métodos mencionados. Para os métodos 1 e 2, deve ser considerada a existência de (a) memória interna disponível para armazenar um vetor de, no máximo, 20 registros, e (b) 40 fitas de armazenamento externo, sendo 20 fitas de entrada e 20 fitas de saída a cada etapa de intercalação. Para o método 3, deve ser considerada a existência de memória interna disponível para armazenar um vetor de, no máximo, 20 registros.

A 2ª fase deste trabalho corresponde à análise experimental da complexidade de desempenho dos 3 métodos mencionados, considerando os seguintes quesitos:

- número de transferências (leitura) de registros da memória externa para a memória interna;
- número de transferências (escrita) de registros da memória interna para a memória externa;
- número de comparações entre valores do campo de ordenação dos registros;
- tempo de execução (tempo do término de execução menos o tempo do início de execução).

Para a 2ª fase, experimentos devem ser realizados considerando o arquivo texto "PROVAO.TXT" (arquivo disponível na página da disciplina) que corresponde a uma base de dados de notas de 471.705 alunos do Ensino Superior que fizeram o "Provão" em 2003 (último ano em que tal avaliação foi aplicada). Cada linha do arquivo contém os seguintes dados relativos a um determinado aluno que fez o "Provão":

- número de inscrição do aluno (valor inteiro longo, ocupando as colunas 1 a 8 do arquivo texto);
- nota obtida pelo aluno (valor real entre 0.0 e 100.0, ocupando as colunas 10 a 14 do arquivo texto);
- estado do aluno (cadeia de 2 caracteres, ocupando as colunas 16 e 17 do arquivo texto);
- cidade do aluno (cadeia de 50 caracteres, ocupando as colunas 19 a 68 do arquivo texto);
- curso do aluno (cadeia de 30 caracteres, ocupando as colunas 70 a 99 do arquivo texto).

No caso, devem ser ordenadas **ascendentemente** as ***n*** primeiras linhas do arquivo texto "PROVAO.TXT", por meio das notas obtidas pelos alunos, considerando ***n*** igual a 100, 1.000, 10.000, 100.000 e 471.705. Ademais, distintas situações de ordem inicial do arquivo devem ser consideradas: arquivo ordenado ascendentemente, arquivo ordenado descendentemente e arquivo desordenado aleatoriamente. O arquivo original encontra-se desordenado aleatoriamente; logo, para as demais situações, devem ser gerados arquivos ordenados ascendente e descendentemente, pelas notas dos alunos, a partir do arquivo original. Cada experimento consiste, então, na execução de um determinado método de ordenação, considerando uma determinada quantidade de alunos e uma determinada situação de ordem inicial do arquivo.

Independente dos experimentos a serem efetuados, o programa **deve** ser implementado de tal forma que seja possível executá-lo, livremente, a partir da seguinte linha de comando no console:

ordena <método> <quantidade> <situação> [-P]

onde:

- <método> representa o método de ordenação externa a ser executado, podendo ser um número inteiro de 1 a 3, de acordo com a ordem dos métodos mencionados;

- <quantidade> representa a quantidade de alunos (linhas do arquivo texto) a serem ordenados;
- <situação> representa a situação de ordem inicial do arquivo, podendo ser 1 (arquivo ordenado ascendentemente pelas notas), 2 (arquivo ordenado descendentemente pelas notas) ou 3 (arquivo desordenado aleatoriamente pelas notas);
- [-P] representa um argumento opcional que deve ser colocado quando se deseja que os dados dos alunos a serem ordenados e o resultado da ordenação realizada sejam apresentados na tela.

Uma execução do comando "ordena" deve retornar, necessariamente, os valores referentes aos quesitos de análise (número de transferências (leitura e escrita) entre as memórias interna e externa, número de comparações e tempo de execução), obtidos a partir de tal execução.

Para fins de avaliação deste trabalho, devem ser entregues:
1. Código-fonte do programa, bem indentado e comentado.
2. Relatório contendo os seguintes pontos:
   - Introdução: especificação do problema a ser resolvido; descrição dos experimentos realizados.
   - Para cada método: análise dos quesitos especificados considerando todos os experimentos realizados sobre o método.
   - Conclusão: análise comparativa de desempenho entre os métodos; principais dificuldades de implementação dos métodos.

Observações Importantes:
- Toda mensagem de orientação e de erro deve ser devidamente tratada.
- O código-fonte do programa deve estar **bem indentado** e **comentado**.
- Trabalhos copiados terão suas notas divididas pelo número de cópias.
- Penalização por atraso: 2,5 pontos a cada aula.
- Para não ocorrer perda na pontuação do trabalho por atraso:
  - o código-fonte e o relatório solicitado devem ser submetidos ao Moodle até às 10h do dia 12/08;
  - o programa deve ser apresentado pelo grupo ao professor na aula do dia 12/08.