



Disciplina: Análise de Algoritmo  
Professora: Josiane Rodrigues  
Aluno: Nome \_\_\_\_\_

Período: 2016/1

### Avaliação Teórico-Prático 1

#### Visão geral do trabalho prático

O trabalho prático consiste na implementação de um algoritmo de **ordenação externa**. Neste componente de avaliação teórico-prático, o aluno deverá implementar em alto nível um algoritmo de ordenação externa por intercalação (**Merge-Sort multivias**) para ordenar um arquivo de dados fornecido como entrada. Observe abaixo, a descrição e orientações pertinentes e importantes acerca da implementação e entrega do mesmo.

#### Conceitos Importantes

Métodos de **ordenação externa** são aqueles utilizados para ordenar um conjunto de dados que se encontra em memória secundária e que não cabe tudo fisicamente na memória principal. Os métodos de ordenação externa devem observar as seguintes restrições:

- Durante o processo de ordenação, parte dos dados deve ser mantida externamente, tipicamente em disco;
- O custo de acessar os dados armazenados externamente, ou seja, em memória secundária, é significativamente maior que o custo de comparação dos itens do conjunto na memória principal;
- Portanto, o número de acessos aos itens em memória secundária deve ser evitado sempre que possível.

#### Disponibilizando sua implementação para correção

A implementação do trabalho prático deverá ser compartilhada por meio do GitHub (<https://github.com/>). Para corrigir seu trabalho, a implementação será clonada por meio do comando git clone.

#### Data de entrega

O trabalho deverá ser entregue até o dia 03 de Maio de 2016.

#### Observações

Esse trabalho deve ser desenvolvido por equipe composta por 4 estudantes.

A equipe poderá escolher a linguagem que preferir para implementação do algoritmo.



### Procedimento e Ética

Comece a fazer esse trabalho logo, enquanto os conceitos estão frescos em memória e o prazo para terminá-lo está tão longe quanto jamais poderia estar. Seja ético, desenvolva seu trabalho em equipe, não copie de outras equipes e nem da internet. A professora poderá pedir defesa do trabalho apresentado.

### Critérios para correção

Implementação correta: 10%,  
Execução correta: 60%,  
Saída Legível: 10%,  
Código inteligível, bem estruturado e comentado: 20%.

### Relatório do trabalho prático

Deverá ser entregue um relatório com no máximo 4 (duas) páginas, contendo as seguintes informações:

- Os nomes e matrículas dos componentes da equipe;
- Breve descrição sobre os objetivos do trabalho;
- Uma breve descrição sobre sua implementação (linguagem, bibliotecas, estruturas de dados);
- O link de sua implementação no GitHub;
- Uma breve tutorial de compilação e execução de sua implementação, que apresentar o que se pede na **Questão 1, Questão 2 e Questão 4**.
- Sucinta descrição sobre os resultados obtidos.

### Desenvolvimento e questões do trabalho prático

**Questão 1-** Descreva em alto nível um algoritmo de ordenação externa por intercalação (**Merge-Sort**) **multivias** para ordenar um arquivo de dados fornecido como entrada;

**Questão 2** - Apresente a complexidade de tempo do seu algoritmo considerando o ambiente de execução em memória secundária;

**Questão 3** - Baseado em seu algoritmo, implemente um programa para ordenar arquivos cujos registros são inteiros de 32 bits. O programa resultante (msort) deverá receber os seguintes parâmetros:

- arquivo de entrada** - arquivo que contém os dados a serem ordenados;
- arquivo de saída** - arquivos que contém os dados ordenados;
- memória** - a quantidade total de memória disponível para a ordenação. O programa não deve alocar mais memória do que o especificado neste parâmetro;
- K** - número de vias usadas pelo Merge-sort.



**Questão 4** - Usando o programa implementado, execute e apresente gráficos dos seguintes experimentos de desempenho (tempo e execução) (Válida somente com a entrega da questão 3).

### Experimentos

Para os experimentos, deve ser gerado um arquivo não ordenado cujos registros são inteiros de 32 bits. Esse arquivo deve ter um tamanho de 6 Gbytes.

**b)** Fixando o parâmetro **K** e usando um mesmo arquivo de entrada, apresente a performance (em milissegundos) do programa implementado para os diversos valores de capacidade de **memória**.

**c)** Fixando o parâmetro **memória** e usando um mesmo arquivo, apresente a performance (em milissegundos) do programa implementado para diversos valores de **K**. Compare as curvas obtidas com a função de complexidade de tempo da Questão 02.