Universidade Federal de Minas Gerais **Departamento de Ciência da Computação**	**Estrutura de Dados** **Profs Wagner Meira Jr e Marcio Costa**

Trabalho Prático 3

Compactação de Arquivos de Texto

Valor: 15 pontos
Data de entrega: 04/07/2023 - 23:59
Atenção: Não serão permitidas entregas atrasadas, ou seja, não há política de desconto por atraso.

Neste trabalho os alunos terão contato com algoritmos de compactação e serão incentivados a usar seus conhecimentos para produzir algoritmos eficientes para reduzir o tamanho de arquivos de textos sem perda de informação.

1. Descrição

Você está trabalhando em um jornal e a empresa está passando por dificuldades financeiras. O departamento financeiro resolveu reduzir o número de discos rígidos utilizados para armazenar informações sobre as edições passadas do jornal. Como você é um muito fã do jornal, gostaria de poder manter as informações presentes em todas as edições do jornal. Para isso você precisará usar seus conhecimentos em estruturas de dados e utilizar um método de compactação famoso: ***O algoritmo de Huffman.***

2. O que deve ser feito?

Você deverá implementar um sistema de compactação baseado no algoritmo de Huffman. Seu sistema deve ser capaz de compactar e descompactar um arquivo passado como entrada. Uma das decisões de projeto é se o sistema deve compactar no nível de letra, no nível de sílaba, ou no nível de palavra. Mais formalmente seu programa deve receber dois nomes de arquivos como entrada por linha de comando e uma flag que pode ser -c (para compactar) ou -d (para descompactar)

No caso da compactação (-c): o primeiro arquivo conterá um texto que deve ser compactado utilizando o algoritmo de Huffman (No livro texto: capítulo 16 seção 3.) e o segundo arquivo estará vazio e conterá, ao fim da execução, as informações da compactação do primeiro arquivo. O arquivo de texto está codificado como UTF-8.

No caso da descompactação (-d): o primeiro arquivo conterá um texto que já foi compactado pelo seu sistema e que deve ser descompactado e o segundo arquivo estará

vazio e conterá a descompactação do primeiro arquivo. O formato do arquivo de saída deve ser UTF-8.

Uma vez implementado, o seu programa deve ser testado arquivos de tamanho variado e números de caracteres (ou sílabas, ou palavras) diferentes. Note que a compressão depende da natureza do arquivo e pode ser necessário avaliar uma amostra significativa de arquivos para ter resultados estatisticamente consistentes.

3. O que deve ser implementado

Você terá que implementar pelo menos um Tipo Abstrato de Dados (TADs)/Classe para representar a informação da compactação do seu arquivo de texto. Classes auxiliares para o processo do algoritmo de Huffman e para leitura e escrita podem ser interessantes.

4. Entregáveis

Você deve utilizar a linguagem C ou C++ para o desenvolvimento do seu sistema. O uso de estruturas pré-implementadas pelas bibliotecas-padrão da linguagem ou terceiros é terminantemente vetado.

Você DEVE utilizar a estrutura de projeto abaixo junto ao Makefile disponibilizado no Moodle:

- TP

    |- src

    |- bin

    |- obj

    |- include

    Makefile

A pasta **TP** é a raiz do projeto; a pasta **bin** deve estar vazia; **src** deve armazenar arquivos de código (*.c, *.cpp, ou *.cc); a pasta **include**, os cabeçalhos (*headers*) do projeto, com extensão ***.h**, por fim a pasta **obj** deve estar vazia. O **Makefile** disponibilizado no *Moodle* deve estar na **raiz do projeto**. A execução do **Makefile** deve gerar os códigos objeto ***.o** no diretório **obj** e o executável do TP no diretório **bin**.

Existe no Moodle um vídeo onde um dos monitores do semestre passado discute sobre a função e uso de makefiles.

Pontuação EXTRA:

Neste trabalho adotaremos uma pontuação extra que será dada através de uma competição entre os códigos gerados pelos alunos. Os códigos finais submetidos serão executados com 10 instâncias de testes. Os códigos que executarem em menos de 5 minutos para todas as

instâncias serão ordenados de acordo com a soma da taxa de compreensão obtida e a quantidade de pontos extras será inversamente proporcional à ordenação. Mais precisamente, a taxa de compressão para um arquivo é dada pela fórmula:

$$taxa_{com}(arquivo) = \frac{tamanho\ de\ arquivo\ em\ bytes}{tamanho\ de\ arquivo\ depois\ da\ compreensão\ em\ bytes}$$

5. Documentação

A documentação do trabalho deve ser entregue em formato **pdf** e também **DEVE** seguir o modelo de relatório que será postado no Moodle. Além disso, a documentação deve conter **TODOS** os itens descritos a seguir **NA ORDEM** em que são apresentados:

Título, nome, e matrícula.
Introdução: Contém a apresentação do contexto, problema, e qual solução será empregada.
Método: Descrição da implementação, detalhando as estruturas de dados, tipos abstratos de dados (ou classes) e funções (ou métodos) implementados.
Análise de Complexidade: Contém a análise da complexidade de tempo e espaço dos procedimentos implementados, formalizada pela notação assintótica.
Estratégias de Robustez: Contém a descrição, justificativa e implementação dos mecanismos de programação defensiva e tolerância a falhas implementados.
Análise Experimental: Apresenta os experimentos realizados em termos de desempenho computacional, assim como as análises dos resultados.
Conclusões: A Conclusão deve conter uma frase inicial sobre o que foi feito no trabalho. Posteriormente deve-se sumarizar o que foi aprendido.
Bibliografia: Contém fontes utilizadas para realização do trabalho. A citação deve estar em formato científico apropriado que deve ser escolhido por você.
Instruções para compilação e execução: Esta seção deve ser colocada em um apêndice ao fim do documento e em uma página exclusiva (não divide página com outras seções).
***Número máximo de páginas:** 10

A documentação deve conter a descrição do seu trabalho em termos funcionais, dando foco nos algoritmos, estruturas de dados e decisões de implementação importantes durante o desenvolvimento.

Evite a descrição literal do código-fonte na documentação do trabalho.

Dica: Sua documentação deve ser clara o suficiente para que uma pessoa (da área de Computação ou não) consiga ler, entender o problema tratado e como foi feita a solução.

6. Submissão

Todos os arquivos relacionados ao trabalho devem ser submetidos na atividade designada para tal no Moodle. A entrega deve ser feita **em um único arquivo** com extensão **.zip**, com nomenclatura nome_sobrenome_matricula.zip, onde nome, sobrenome e matrícula devem ser

substituídos por suas informações pessoais. O arquivo **.zip** deve conter a sua documentação no formato **.pdf** e a estrutura de projeto descrita no início da Seção 4.

7. Avaliação

O trabalho será avaliado de acordo com:

- Definição e implementação das estruturas de dados e funções - (30% da nota total)
- Corretude na execução dos casos de teste - (20% da nota total)
- Apresentação da análise de complexidade das implementações - (15% da nota total)
- Estrutura e conteúdo exigidos para a documentação - (20% da nota total)
- Indentação, comentários do código fonte e uso de boas práticas - (10% da nota total)
- Cumprimento total da especificação - (5% da nota total)

Se o programa submetido não compilar[1], seu trabalho não será avaliado e sua nota será zero.

8. Considerações Finais

1. Comece a fazer esse trabalho prático o quanto antes, enquanto o prazo de entrega está tão distante quanto jamais estará.
2. Leia **atentamente** o documento de especificação, pois o descumprimento de quaisquer requisitos obrigatórios aqui descritos causará penalizações na nota final.
3. Certifique-se de garantir que seu arquivo foi submetido corretamente no sistema.
4. Plágio é CRIME. Trabalhos onde o plágio for identificado serão **automaticamente anulados** e as medidas administrativas cabíveis serão tomadas (em relação a todos os envolvidos). Discussões a respeito do trabalho entre colegas são permitidas. É permitido consultar fontes externas, desde que exclusivamente para fins didáticos e devidamente registradas na sessão de bibliografia da documentação. **Cópia e compartilhamento de código não são permitidos**.

FaQ (Frequently asked Questions)

1. **Posso utilizar alguma estrutura de dados do tipo Queue, Stack, Vector, List, e etc..., do C++? NÃO**
2. Posso utilizar o tipo String? SIM.
3. Posso utilizar o tipo String para simular minhas estruturas de dados? NÃO
4. Posso utilizar alguma biblioteca para tratar exceções? SIM.
5. Posso utilizar alguma biblioteca para gerenciar memória? SIM.
6. As análises e apresentação dos resultados são importantes na documentação? SIM.

---

[1] Entende-se por compilar aquele programa que, independente de erros no Makefile ou relacionados a problemas na configuração do ambiente, gere um código executável e atenda aos requisitos especificados neste documento em um ambiente Linux.

7. Os meus princípios de programação ligados a C++ e relacionados a engenharia de software serão avaliados? NÃO
8. Posso fazer o trabalho em dupla ou em grupo? NÃO
9. Posso trocar informações com os colegas sobre a teoria? SIM.
10. Posso fazer o trabalho no Windows, Linux, ou MacOS? SIM.
11. Posso utilizar IDEs, Visual Studio, Code Blocks, Visual Code, Eclipse? SIM.