UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO

LABORATÓRIO DE PROGRAMAÇÃO – CK0215 2021.1 T01A

TRABALHO 3: COMPACTAÇÃO VIA HUFFMAN (2021-07-16)

PROFESSOR: PABLO MAYCKON SILVA FARIAS

1 Resumo

Você deverá implementar em C++ um compactador e descompactador baseado na codificação de Huffman. O compactador deve gerar arquivos compactados de tamanho compatível com a codificação de Huffman, e o descompactador deve produzir saídas idênticas aos arquivos originais fornecidos ao compactador. Além disso, ambos devem executar em tempo aceitável, conforme os exemplos apresentados mais à frente. A implementação será entregue ao professor através de tarefa no SIGAA.

Atenção: Colabore para o bom andamento da disciplina e do aprendizado de todos, não copiando códigos de outras pessoas nem compartilhando seu próprio código. Em caso de dificuldade sua ou com algum colega, fale com o professor.

2 Compilação do Programa

O seu programa deverá ser escrito em C++17 padrão e ser compilável, sem erros ou avisos, através de uma linha como

```
g++ -Wall -Wextra -std=c++17 -pedantic -o programa  main.cpp
```

Caso algo além disso seja necessário para compilar o programa (por exemplo, caso o seu programa possua mais de um arquivo que precise ser explicitamente fornecido como entrada para o compilador), por favor inclua um arquivo `explicacoes.txt` com as devidas explicações.

3 Execução do Programa

A entrada do programa será necessariamente informada através dos argumentos fornecidos na chamada do mesmo, a qual terá o seguinte formato (abaixo, os colchetes apenas delimitam os nomes de *metavariáveis*, não devendo estar presentes nas chamadas ao programa):

```
./programa [operação] [arquivo de entrada] [arquivo de saída]
```

Você pode escolher as *strings* a serem utilizadas pelo usuário para informar a operação, se compactação ou descompactação. Elas podem ser tão simples quanto `c` para compactar e `d` para descompactar, mas você também pode fazer outras escolhas, como `-c`, `--comprimir` ou `--compress` para compactar, e analogamente para a descompactação. **Importante:** informe no arquivo `explicacoes.txt` a maneira pela qual as 2 operações do programa devem ser especificadas na chamada do mesmo.[1]

[1] Na prática, outra maneira interessante de fazer isso é imprimir na tela as maneiras de chamar o programa quando ele for chamado sem nenhum argumento. Você também pode implementar essa pequena funcionalidade,

Este trabalho não faz exigências sobre o comportamento do programa nos casos em que a chamada não estiver no formato esperado.

4 Instâncias de Teste

O seu programa deve ser capaz de, em princípio, compactar e descompactar arquivos quaisquer,[2] mas este trabalho possui um conjunto padrão de instâncias de teste, que estarão disponíveis no SIGAA e servirão para fornecer parâmetros de tamanho de arquivo e tempo de execução. Seguem abaixo resultados para um programa escrito pelo professor em 2015:[3]

Instância	Compressão	Descompressão	Tamanho Original	Tamanho Compactado
1.txt	0,002s	0,003s	6 bytes	32 bytes
2.txt	0,002s	0,002s	100 bytes	27 bytes
3.txt	0,002s	0,002s	0 bytes	0 bytes
4.txt	0,287s	0,314s	6,2M	3,6M
5.pdf	0,014s	0,023s	170K	169K
6.bmp	1,131s	2,809s	27M	26M

5 Critérios de Avaliação

A corretude do programa é critério básico de avaliação: o programa deve compactar e descompactar arquivos sem incorrer em erros de execução, e o resultado da descompactação deve ser igual ao arquivo inicialmente submetido à compactação.

O tempo de execução do programa também será considerado: tempos de execução tão longos que dificultem a realização de testes gerarão perda de pontos na avaliação. Observe que o professor também pode testar o seu programa para instâncias diferentes daquelas listadas acima.

O professor também avaliará o código-fonte à luz do conteúdo ensinado na disciplina. Trechos de código excessivamente ineficientes poderão levar à perda de pontos.

Por fim, a realização do trabalho de forma individual e de acordo com as regras da disciplina são, em qualquer trabalho, elementos básicos para a nota.

6 Submissão do Trabalho

A solução do trabalho deverá consistir num arquivo `[Matrícula].zip` (sem colchetes) entregue através do SIGAA em tarefa a ser cadastrada pelo professor. Naturalmente, apenas o código-fonte deve ser submetido, sem qualquer arquivo executável. Prezando pelo tamanho do arquivo `zip` final, **também não devem ser incluídas instâncias para o programa**. Quaisquer explicações adicionais podem ser incluídas através de um arquivo `explicacoes.txt`.

Em caso de dúvida, contate o professor rapidamente.

– Eu espero que você se divirta e aprenda. Bom trabalho! –

caso deseje, mas ela não é obrigatória neste trabalho.

[2] Exceto, claro, por restrições de transbordo de tipos como `int`, mas, numa arquitetura de 32 bits ou mais, um `int` é capaz de contar os bytes de arquivos com mais de 1 gigabyte.

[3] Os resultados são para uma compilação sem opções explícitas de otimização. O seu programa não precisa executar exatamente nesses tempos, até porque o tempo exato obviamente varia de máquina para máquina, mas de qualquer forma os tempos servem de parâmetro: se o seu programa executar num tempo muitas vezes maior, então você deve analisar se há algo implementado de maneira particularmente ineficiente.