

### Trabalho de Programação 3

#### Programação com o Processador Intel 80x86

##### 1. Descrição Geral do Trabalho

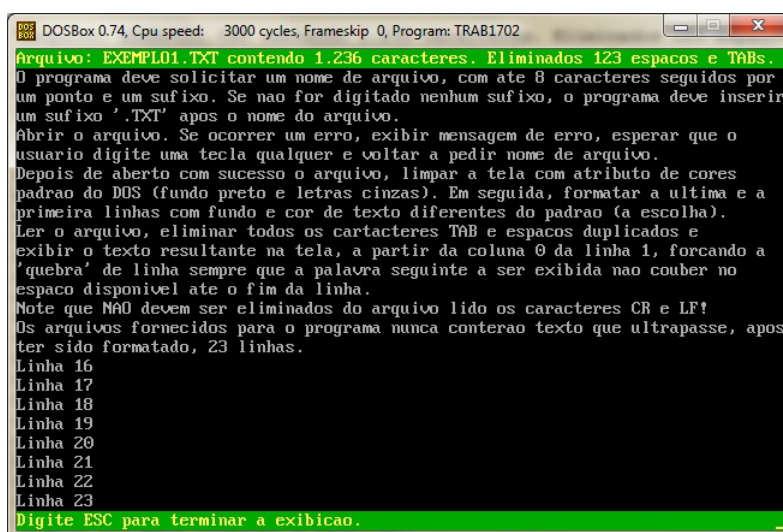
---

O programa a ser desenvolvido para o processador Intel será uma nova versão daquele feito para o processador Cesar. Neste trabalho, você usará chamadas de serviços do sistema operacional DOS e do BIOS para interagir com periféricos reais (teclado, tela e disco). O programa vai ler os dados de um ou mais arquivos em disco gravados no formato "texto do MS-DOS" e escrever na tela os textos formatados, usando o modo de vídeo com texto em 25 linhas de 80 caracteres. O processamento dos dados lidos do arquivo será muito semelhante ao feito no trabalho com o processador Cesar, mas desta vez o resultado será exibido na tela com algumas informações adicionais. A seguir estão indicadas as diversas etapas do processamento que deverão ser implementadas.

##### 2. Etapas do algoritmo básico a ser desenvolvido

---

1. O programa deverá limpar a tela e escrever na primeira linha uma mensagem solicitando ao usuário que digite o nome do arquivo a ser processado. O nome digitado deve ser ecoado na mesma linha da mensagem. Durante a leitura do nome do arquivo, o programa deverá tratar adequadamente o caractere BACKSPACE (código ASCII 8). Quando o usuário digitar ENTER (código ASCII 13 ou 0DH), a digitação do nome do arquivo de entrada estará terminada. Se for digitado somente ENTER, o programa deverá limpar a tela, exibir uma mensagem de encerramento (texto da mensagem a escolher, mas deve ser bem educada ☺) e terminar a execução. O nome do arquivo de entrada poderá ter um sufixo de tipo ('.xxx') ou não. Caso o usuário forneça apenas o nome (sem nenhum caractere '.'), o programa deverá acrescentar, antes de realizar a abertura do arquivo, um sufixo '.TXT' ao nome digitado pelo usuário. O arquivo poderá ter qualquer nome, mas, para ser compatível com o MS-DOS, este nome (sem contar o sufixo de tipo) deve ter, no máximo, 8 caracteres (de preferência, apenas letras e números e iniciando com uma letra – o DOS não faz distinção entre letras maiúsculas e minúsculas ao procurar um arquivo no diretório do disco).
2. Abrir o arquivo especificado. Se ocorrer um erro na abertura, exibir na segunda linha da tela uma mensagem de erro (explicando qual o tipo de erro) e esperar que o usuário pressione qualquer tecla antes de continuar. Voltar para a etapa 1. Se não ocorrer erro, prosseguir para a etapa 3.
3. Limpar novamente a tela, definindo as cores de fundo e texto padrão DOS, ou seja, fundo preto e letras em cinza claro, para todas as linhas. Depois, formatar a última e a primeira linha da tela (nesta ordem) para exibir texto com cores de fundo e de letras diferentes do padrão DOS (escolher uma combinação de cores que permita ler com facilidade o que estiver escrito ☺).
4. Ler o arquivo especificado, eliminando todos os caracteres TAB (código ASCII 9) e espaços (código ASCII 32 ou 20H) duplicados. Note que, neste trabalho, os caracteres CR (código ASCII 13 ou 0DH) e LF (código ASCII 10 ou 0AH) **NÃO** devem ser eliminados! Contabilizar a quantidade total de caracteres lidos do arquivo de entrada e quantos caracteres foram eliminados. Quando encontrar o fim do arquivo, exibir na primeira linha da tela estas estatísticas e exibir na linha 24 as instruções de operação para o usuário (ver exemplos nas Figuras 1 e 2).
5. Exibir o texto resultante após a eliminação dos caracteres na tela, a partir da linha 1 até a linha 23 (não alterar o que estiver sendo exibido nas linhas 0 e 24), cuidando para que nenhuma palavra seja “quebrada” em duas linhas, ou seja, deixando em branco o resto da linha quando a próxima palavra não couber no espaço restante. Para esta finalidade, usar o mesmo conceito de “palavra” do trabalho com o Cesar, ou seja, considerar que uma “palavra” é qualquer sequência (não vazia) de caracteres, delimitada por espaços em branco. Assim, no texto “teste, teste e mais testes.” existem as palavras “teste,”, “teste”, “e”, “mais” e “testes.”. Para a versão básica do trabalho, o programa será testado apenas com arquivos que, após a formatação em linhas aqui descrita, possam ser exibidos em apenas 23 linhas da tela. As Figuras 1 e 2 mostram exemplos de exibição de textos lidos.
6. Após exibido o texto, aguardar que o usuário digite os caracteres especificados nas instruções exibidas na linha 24 e executar o que for previsto para cada caractere. Na versão básica, se for digitado um caractere ESC (código ASCII 27 ou 1BH), voltar para a etapa 1 e ignorar todos os demais caracteres.



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: TRAB1702
Arquivo: EXEMPLO1.TXT contendo 1.236 caracteres. Eliminados 123 espaços e TABs.
O programa deve solicitar um nome de arquivo, com ate 8 caracteres seguidos por
um ponto e um sufixo. Se nao for digitado nenhum sufixo, o programa deve inserir
um sufixo ' .TXT' apos o nome do arquivo.
Abrir o arquivo. Se ocorrer um erro, exibir mensagem de erro, esperar que o
usuario digite uma tecla qualquer e voltar a pedir nome de arquivo.
Depois de aberto com sucesso o arquivo, limpar a tela com atributo de cores
padrao do DOS (fundo preto e letras cinzas). Em seguida, formatar a ultima e a
primeira linhas com fundo e cor de texto diferentes do padrao (a escolha).
Ler o arquivo, eliminar todos os cartacteres TAB e espacos duplicados e
exibir o texto resultante na tela, a partir da coluna 0 da linha 1, forçando a
'quebra' de linha sempre que a palavra seguinte a ser exibida nao couber no
espaco disponivel ate o fim da linha.
Note que NAO devem ser eliminados do arquivo lido os caracteres CR e LF!
Os arquivos fornecidos para o programa nunca conteraao texto que ultrapasse, apos
ter sido formatado, 23 linhas.
Linha 16
Linha 17
Linha 18
Linha 19
Linha 20
Linha 21
Linha 22
Linha 23
Digite ESC para terminar a exibicao.
```

Figura 1 – Exemplo de formatação da tela e exibição do texto na versão básica

### 3. Formato do Arquivo de Entrada

O arquivo a ser lido estará no formato “texto do MS-DOS”, onde cada caractere é codificado em ASCII padrão americano (códigos 0 a 127) e ocupa um byte. O fim de linha é representado por um par de caracteres CR e LF (*Carriage Return* e *Line Feed*), que podem ser precedidos/seguídos por espaço(s) ou não. Não haverá limite para o tamanho de cada linha, mas será observado, nos arquivos de teste, o especificado na descrição da etapa 5 na sessão 2. Não existe nenhum caractere indicador de fim de arquivo e a última linha pode ou não ser terminada por um par CR e LF. Em qualquer caso, o arquivo de entrada terá, no máximo, 16 kB de tamanho.

### 4. Versão estendida – opcional, valendo um “bônus” de 20% na nota do trabalho com Intel

Na correção do trabalho em sua versão básica, descrita nas sessões anteriores, será atribuída uma nota de 0 a 100. Para aqueles que desejarem melhorar sua média final, oferecemos a oportunidade de implementar uma versão estendida do trabalho, que se caracterizará por permitir a exibição de textos que, após formatados, ocupem mais de 23 linhas da tela. Nesta versão, além do indicado na sessão 2 do enunciado, devem ser observados as seguintes especificações:

- Na etapa 4, as instruções de operação exibidas na linha 24 devem ser as indicadas na Figura 2.
- Na etapa 5, devem ser exibidas nas linhas 1 a 23 da tela somente as primeiras 23 linhas do texto formatado.
- Na etapa 6, além do caractere ESC, devem ser aceitos os caracteres 'w' (ou 'W') e 's' (ou 'S'). Se for digitado um 'w', as linhas 2 a 23 da tela devem ser “roladas” para cima e na linha 23 deve ser exibida a próxima linha do texto que não aparecia na tela. Se for digitado um 's', as linhas 1 a 22 da tela devem ser “roladas” para baixo e na linha 1 deve ser exibida a primeira linha anterior do texto que não aparecia na tela. **Nas operações de rolagem, não deve haver rolagem para cima quando a linha 1 já estiver exibindo a primeira linha do texto formatado nem rolagem para baixo quando a linha 23 já estiver exibindo a última linha do texto formatado.**

### 5. Entregáveis: o que deve ser entregue?

Deverá ser entregue via Moodle da disciplina o arquivo fonte com a solução do problema apresentado, escrito na *linguagem simbólica de montagem* do 8086 da Intel (arquivo .ASM). O código do programa fonte deverá conter comentários descritivos da implementação. Para a correção, os programas serão montados e executados no ambiente DosBox usando o montador TASM. Aos programas que forem montados sem erros serão aplicados arquivos de teste. A nota final do trabalho será proporcional às funcionalidades que forem atendidas pelo programa. O trabalho deverá ser entregue até a data prevista (conforme indicado no MOODLE). **Não serão aceitos trabalhos entregues além do prazo estabelecido.**

Figura 2(a) – Exemplo de formatação inicial da tela e exibição do texto na versão estendida

Figura 2(b) – Exemplo de formatação da tela e exibição do texto na versão estendida, após rolagem da tela para baixo

## 6. Observações

Recomenda-se a troca de ideias entre os alunos. Entretanto, a identificação de cópias de trabalhos acarretará na aplicação do Código Disciplinar Discente e a tomada das medidas cabíveis para essa situação (**tanto o trabalho original quanto os copiados receberão nota zero**).

O professor da disciplina reserva-se o direito, caso necessário, de solicitar uma demonstração do programa, onde o aluno será arguido sobre o trabalho como um todo. Nesse caso, a nota final do trabalho levará em consideração o resultado da demonstração.