

LINGUAGEM DE MONTAGEM – RELATÓRIO 6

Marco Vinícius Guebarra

RA: 151045054

Guilherme Gervaes

RA: 151041946

Universidade Estadual Paulista “Júlio de Mesquita Filho”

1. Introdução

Durante a disciplina de Laboratório de Linguagem de Montagem, compreendemos as vantagens e a importância de utilizar assembly no desenvolvimento de projetos. Dentre uma série de pontos, podemos destacar a alta eficiência combinada com o baixo custo de memória que um programa trabalhado com esta linguagem demanda (quando comparado a linguagens de alto nível), além da possibilidade de utilizar registradores, que por se encontrarem bem próximos do processador, torna o processamento muito mais rápido. Tudo isso faz com que o assembly, embora seja uma linguagem pouco intuitiva em termos de comandos, necessite ser conhecida por todo programador.

2. Objetivo

O objetivo dessa atividade é implementar um programa simulando um editor de texto. O usuário deverá passar como parâmetro um arquivo “.txt”, e o programa deverá verificar se este existe. Se sim, o programa será responsável por carregar o arquivo na memória. Se não, deverá criar o documento. Além disso, existirão as opções “/slvr”, que permitirá ao usuário salvar qualquer alteração digitada no documento, “/sair” para encerrar o processo e “/cpal” para contar o número de palavras no arquivo de texto.

3. Metodologia

A linguagem de programação *assembly* foi utilizada para a realização do projeto. Foram utilizados os registradores e as diretivas de reserva “res” para reserva de memória para variáveis e diretivas de definição “d” e para definição de constantes. Para

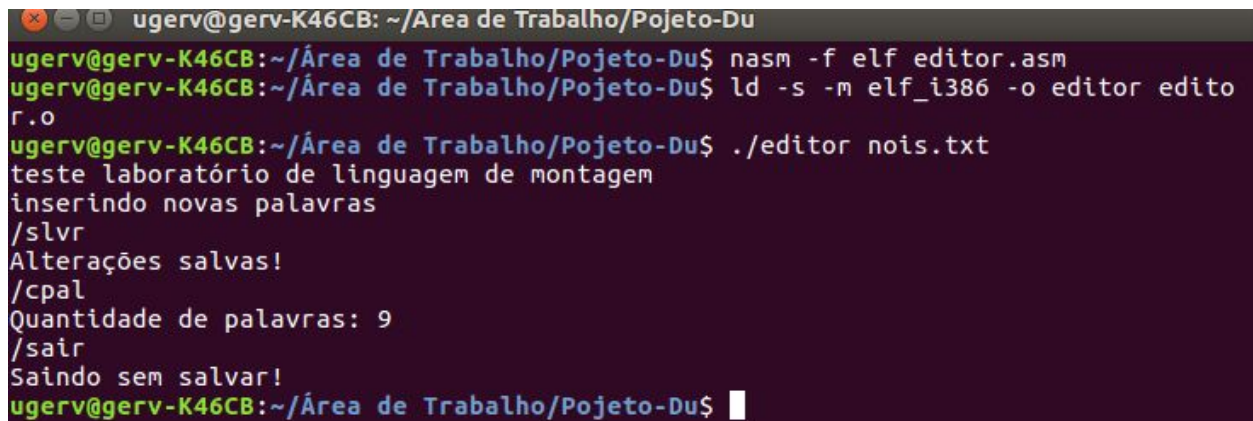
a implementação do programa foi utilizado o editor de texto Pluma (pluma editor) e o assembler NASM (Netwide Assembler), cuja função é converter códigos em *assembly* para linguagem de máquina executável.

A implementação seguiu os seguintes passos:

- 1 – Declaração na *section .data* contendo as mensagens que serão apresentadas ao usuário;
- 2 – Declaração na *section .bss* das variáveis que serão utilizadas na manipulação do programa;
- 3 – Declaração da *section .text*;
- 4 - Declaração da sub-rotina *exibeArquivo*, utilizada para ler informações do arquivo desejado e imprimir as informações contidas;
- 5 – Declaração das sub-rotinas *converteChar*, *printarNro* e *conta* utilizada para auxiliar a sub-rotina *contaPalavras* na contagem;
- 6 – Declaração das sub-rotina *.salvar*, *.sair* e *.palavras*, utilizadas para lidar com os comandos inseridos pelo usuário;
- 8 – Declaração das sub-rotinas *lerNovoTexto* e *.comando*, utilizadas para ler a entrada do usuário e determinar o que deve ser feito com base nos comandos inseridos;
- 9 – Declaração da sub-rotina *criaArquivo*, utilizada no caso do arquivo requisitado pelo usuário ainda não exista;
- 10 – Declaração da sub-rotina *abreArquivo*, utilizada para deixar o arquivo (caso exista) pronto para ser manipulado;
- 11 – Declaração da sub-rotina *encerrar*, utilizada para terminar o programa;
- 12 – Declaração da *__start*; na qual vai reconhecer o argumento passado (caso seja passado algum) e procurar o arquivo; analisar o argumento passado, caractere por caractere; abrir e/ou criar o arquivo necessário; exibir conteúdo do arquivo (caso haja algum) e permitir a modificação pelo usuário; carregar as alterações feitas pelo usuário, deixando as opções de manipulação em aberto para serem utilizadas; fechar o arquivo modificado; encerrar o programa.

4. Resultados

Após terminada a implementação do programa foi obtido o executável, que foi submetido a um teste para analisar se o programa correspondia ao que foi proposto em sala. A *figura 2.1* mostra o teste realizado.

A terminal window with a dark background and light-colored text. The prompt is 'ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du'. The user enters 'nasm -f elf editor.asm', followed by 'ld -s -m elf_i386 -o editor editor.o'. Then they run './editor nois.txt'. The program outputs: 'teste laboratório de linguagem de montagem', 'inserindo novas palavras', '/slvr', 'Alterações salvas!', '/cpal', 'Quantidade de palavras: 9', '/sair', and 'Saindo sem salvar!'. The prompt returns to 'ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du\$' with a cursor.

```
ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du$ nasm -f elf editor.asm
ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du$ ld -s -m elf_i386 -o editor editor.o
ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du$ ./editor nois.txt
teste laboratório de linguagem de montagem
inserindo novas palavras
/slvr
Alterações salvas!
/cpal
Quantidade de palavras: 9
/sair
Saindo sem salvar!
ugerv@gerv-K46CB: ~/Área de Trabalho/Pojeto-Du$
```

Figura 2.1 – Teste programa editor de texto

5. Discussão

O teste ocorreu como planejado e contou com um arquivo texto “nois.txt” previamente criado. Na linha de comando que executa o programa foi colocado o nome do arquivo e o arquivo foi aberto corretamente.

O programa imprimiu corretamente o conteúdo do arquivo e espera novas inserções. Foram adicionadas mais algumas palavras, foram solicitados os três comandos “/slvr”, “/cpal” e “/sair” e o programa correspondeu como esperado.

6. Conclusão

Com a execução deste projeto, pudemos colocar em prática todos os conceitos aprendidos durante a disciplina, bem como compreender como algumas funções básicas do sistema operacional funcionam. A partir disso, compreendemos ainda mais por que o uso do assembly é imprescindível nos dias de hoje, sendo de muito valor o conhecimento profundo nesta linguagem.