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 possui (quando comparado a linguagens de alto nível), além da possibilidade de conversar diretamente com o processador, sem intermédios. Tudo isso faz com que o assembly, embora seja uma linguagem pouco intuitiva, necessite ser conhecida por todo programador.

1. **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.

1. **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 o arquivo desejado e imprimir as informações contidas;

5 – Declaração da sub-rotina *converteChar*, utilizada para transformar a informações em caracteres;

6 – Declaração da sub-rotina *printarNro*, utilizada para exibir ao usuário o número de palavras contidas no documento;

7 – Declaração da sub-rotina *salva*, utilizada para salvar o documento modificado;

8 – Declaração da sub-rotina *lerNovoTexto*, utilizada para receber as opções disponíveis ao usuário (“/slvr”, “/sair”, “/cpal”) e tomar as decisões necessárias;

9 – Declaração da sub-rotina *criaArquivo*, utilizada se, em caso de ausência de documento, criar o arquivo a ser modificado;

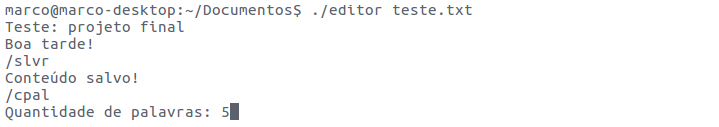
10 – Declaração da sub-rotina *abreArquivo*, utilizada para deixar o arquivo (caso exista) pronto para ser carregado, lido e escrito;

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.

1. **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.

 Figura 2.1 – Teste programa editor de texto

1. **Discussão**

GERV

1. **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.