Relatório Técnico – Projeto de Introdução à Linguagem de Programação / unidade 1

Aluno: Bruno Ferreira da SIlva

Introdução

Este projeto foi desenvolvido como parte da disciplina Introdução à Linguagem de Programação, com o objetivo de aplicar na prática os principais conceitos estudados ao longo do semestre.

O programa implementa uma calculadora multifuncional, que combina uma calculadora científica com suporte a operações matemáticas diversas e um conversor de unidades, abrangendo moedas, comprimento, área, volume, massa e velocidade.

O projeto buscou consolidar o aprendizado dos seguintes conteúdos: variáveis, tipos e operadores, estruturas condicionais, laços de repetição, funções e vetores. Além disso, teve como foco o desenvolvimento de boas práticas de organização e clareza no código.

Justificativa da Escolha do Projeto

Quando eu estava escolhendo o meu projeto, decidi optar por algo que não exigisse muito trabalho, pois já possuo certo conhecimento de programação — obtido durante o curso técnico em Informática integrado ao ensino médio — e não estava com muito tempo livre para algo mais complexo.

Por esse motivo, escolhi inicialmente uma calculadora científica. Entretanto, percebi que apenas isso seria simples demais, então resolvi ampliar o projeto, criando uma calculadora multifuncional, com a adição de um conversor de unidades e planos futuros de incluir uma calculadora de matrizes.

Metodologia

O desenvolvimento foi realizado na linguagem C, utilizando o compilador GCC em ambientes Linux e Windows, garantindo a portabilidade do código.

As principais ferramentas empregadas foram:

- Visual Studio Code (VS Code) como editor de código;
- Compilador GCC para testes e execução;
- GitHub para versionamento e histórico do projeto.

A abordagem de desenvolvimento adotada foi incremental:

1. Implementação das funções básicas da calculadora científica;
2. Desenvolvimento dos módulos de conversão de unidades;
3. Inclusão de recursos adicionais, como o histórico de cálculos.

A modularização por funções foi uma prioridade, pois melhora a clareza, a legibilidade e o reaproveitamento do código.

Análise do Código

O código foi organizado em funções específicas para cada funcionalidade:

- Função `calcular`: responsável pelas operações matemáticas principais (soma, subtração, multiplicação, divisão, potência, raiz quadrada, logaritmo, módulo e fatorial).
- Função `calculadora_cientifica`: interface principal da calculadora, responsável por ler comandos do usuário, chamar a função de cálculo e gerenciar o histórico.
- Funções `conversor_*`: cada categoria de unidades (moedas, comprimento, área, volume, massa e velocidade) possui sua própria função, garantindo organização e clareza.
- Funções auxiliares (`add_hist`, `mostrar_hist`, `limpar_hist`): gerenciam o histórico de cálculos, utilizando um vetor de estruturas.

Aplicação dos Conceitos da Disciplina

- Variáveis e Tipos: foram utilizados inteiros e números de ponto flutuante (`int`, `float`) para armazenar e processar cálculos e valores convertidos.

- Estruturas Condicionais: amplamente utilizadas em blocos `if-else` para identificar operações e conversões.

- Laços de Repetição: aplicados no cálculo de fatoriais (com `for`) e no menu principal (com `do-while`).

- Funções: cada módulo foi encapsulado em uma função independente, facilitando a manutenção e o entendimento do código.

- Vetores e Structs: o histórico de cálculos foi implementado através de um vetor de estruturas (`struct CalcHist`), armazenando expressões e resultados.

Dificuldades Encontradas

Durante o desenvolvimento, algumas das principais dificuldades foram:

- Falta de prioridade nas operações em expressões numéricas (ordem incorreta de execução das operações);

- Saída de números inteiros sendo exibidos como números flutuantes (com casas decimais desnecessárias);

- Implementar uma função de histórico realmente satisfatória utilizando apenas os conteúdos da unidade;

- Dificuldade na organização do código em arquivos separados, com chamadas entre funções e um arquivo principal (`main`).

Soluções Implementadas

- Prioridade de operações e saída numérica: pretendo corrigir nas próximas unidades utilizando ponteiros, condicionais e conversões de tipo.

- Histórico de cálculos: para superar a limitação, implementei uma estrutura (`struct`) que armazena expressões e resultados, criando um histórico funcional dentro das restrições da unidade.

- Organização em arquivos: embora ainda não tenha conseguido modularizar o código, compreendi o processo e planejo aplicá-lo em unidades futuras.

Organização do Código

O código foi organizado em diferentes funções para deixar o projeto mais limpo, legível e modular, permitindo compreender facilmente a função de cada parte.

Gostaria de ter dividido o código em múltiplos arquivos (`.c` e `.h`), mas ainda não consegui implementar essa estrutura na prática.

Conclusão

Com este projeto, aprendi na prática o uso de estruturas condicionais, vetores, funções e structs em C.

Também percebi que, diferentemente de linguagens como Python, para dividir o código em arquivos reutilizáveis em C, é necessário criar um arquivo header (`.h`), que funciona como um cabeçalho de comunicação entre os arquivos do programa.

Como melhorias futuras, pretendo:

- Implementar prioridade de operações;
- Corrigir a exibição de números inteiros;
- Modularizar o código em múltiplos arquivos;
- Adicionar uma calculadora de matrizes como novo módulo da aplicação.