2016.2

Algoritmo e Estrutura de dados I

Jesimon Barreto Santos

[CALCULADORA COMPLEXA 2]

A matéria é voltada para conhecimento de algoritmo na linguagem C e foi proposta a criação de uma calculadora para expressões complexas, dessa forma, este artigo tem o objetivo de apresentar o software criado como: objetivos, abordagens, estruturas da linguagem C utilizadas, visão geral do projeto, além das limitações em relação aos cálculos que o software pode calcular.

I. Introdução

O Curso de Algoritmo e Estrutura de Dados I tem como pré-requisito a construção de um algoritmo de calculadora complexa na linguagem C, na parte iI, com o valor de 25 pontos na matéria. A linguagem C na parte 2 foi apresentada pela professora à turma, cujo os assuntos foram Ponteiros; Tipos definidos pelo usuário; Alocação Dinâmica; Arquivos; Arquivos binários. O aprendizado foi guiado por meio de aulas com algoritmos na linguagem referente e exercícios voltados para a prática e aulas no laboratório. Um desses exercícios que serão avaliados é o Trabalho Prático II.

II. Objetivo

O objetivo geral deste artigo é apresentar o Trabalho Prático 2, que teve na sua totalidade o objetivo de refatorar o Trabalho Prático 1 que consistia em um programa em C que reconhece duas expressões do tipo complexa, que podem ser apresentadas de dois tipos (Retangular e Polar), para calcular soma, subtração, multiplicação, divisão e potência por um número natural, essas operações serão digitadas em conjunto com as expressões e precisam ser reconhecidas. Além disso, o software possibilitar conversão da equação no modo Polar para Retangular e o inverso. Ainda assim, a entrada é por meio de um arquivo que contem um cálculo em cada linha do arquivo. Esses valores fazem parte de uma expressão complexa que deve ser resolvida e gravada em outro arquivo. Porém, o trabalho prático 2 deve conter em seu código alocação dinâmica dos dados e ponteiros.

III. Desenvolvimento

Neste tópico de desenvolvimento terá menos descrição, pois boa parte da arquitetura do trabalho I foi reaproveitada para esse segundo trabalho. Mesmo assim, a figura 1 foi criada com objetivo de reunir ao máximo todas as informações possíveis do funcionamento geral do programa, em todas as camadas possíveis. Uma descrição superficial a primeira vista, apresentada as 4 camadas de funcionamento geral do programa em execução. A camada de arquivo, responsável pela entrada e saída dos dados de execução do programa; camada de tratamento de erro, responsável por estar verificando se ocorre erro na execução; camada calculadora é a parte principal do programa, visto que se trata de uma calculadora, ainda assim, é nessa camada que se encontram as funções de soma, subtração, multiplicação, divisão, potencia e conversão; e a camada de conversão de dados que é responsável por transferir os dados do tipo string, que é o tipo capturado no arquivo, e converter para inteiro e colocá-los na struct. Mesmo com essa explicação vale salientar que as camadas não seguem uma ordem necessária para acontecerem, com exceção da camada de Arquivo sempre ocorre em primeiro lugar, mas de modo geram quem gerencia é a execução geral do programa, representada pela engrenagem na figura 1.

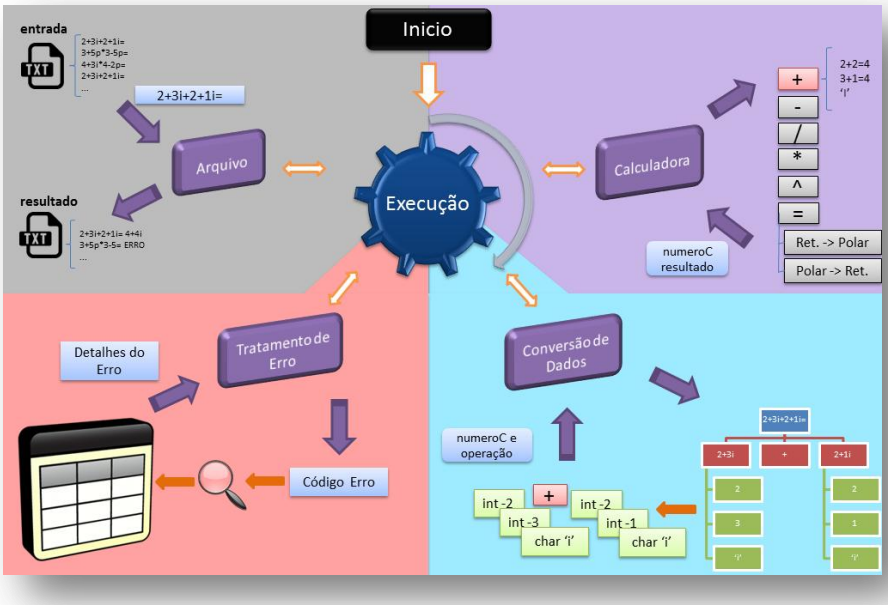

Figura 1. Descrição completa de funcionamento do programa

Partes do desenvolvimento incluem a adição da biblioteca ‘stdlib’ da própria liguagem com objetivo de usar o a alocação dinâmica da memoria. Como também, o uso do ‘typedef’ para facilitar a manipulação do struct que agora é referenciado no programa e neste trabalho como ‘numeroC’. Foram adicionadas funções para manipular arquivos, e ajudar na manipulação. O próximo tópico irá explicar sobre como ocorre a entrada de dados, quais tipos de cálculos são aceitos até a forma como são passados para o calculo.

IV. Entrada de dados

Esse tópico apresenta a entrada de dados para aplicação no programa, a figura 1 será usada como referencia para explicação de utilização das camadas no processo de entrada de dados a medida que será explicado as formas de cálculos aceitas.

A entrada de dados é realizada pela camada de Arquivo que acessa os dados e captura uma linha por vez para fazer o processamento. A linha é considerada até o ponto que aparece um caractere de ‘=’, capturada no formato string, é considerada um cálculo, então a primeira restrição de entrada é que cada cálculo seja finalizado com o caractere ‘=’.

Depois de capturada a string passa por uma serie de testes para verificar se tem algum tipo de irregularidade sobre caracteres diferentes dos padrões. Para explicar melhor essa série de testes será usado alguns cálculos da lista de entrada passada junto com a proposta de TP2, o que serve para qualquer cálculo que siga o modelo, verifique abaixo.

(1) -1+2i--1+2i=
(2) (2)-1+2i*-1+2i=

(3) -1+2i/-1+2i=
(4) ++1-2i=
(5) --1+2i=
(6) -12+23i*-1+-3p=

Nesse caso das equações (1;2;3;4;5;6), na verificação do TP1 seria considerada erro, porém no TP2 é aceito, o erro seria reconhecido no sinal de subtração depois da operação entre as duas expressões. O TP2 processa como dois sinais juntos, e nesse caso se aparecerem mais de um sinal de mais ou menos é feito jogo de sinal com o numero mais próximo, na situação da equação 1 admite o valor '-1'.

(7) -1+2i^2=

A equação (7) é do tipo de equação do cálculo de potência de um numero complexo por um numero real.

(8) 2+4i/2=

No caso da equação acima (8), foi acusado erro pois foi resolvido ignorar cálculos entre números complexos e reais, a não ser no caso acima, de potencia.

(9) -12+23i * -1+-3p=
(10) - 1 - 4 p =

As equações (9;10) estão sendo apresentadas para representar o grupo de equações que tem espaço entre seus dados, o programa ignora os espaços quanto está carregando os dados para serem processados.

(11)-1+2i&2-3p=
(12)-1+2i/0+0i=

Nesse caso as equações são rejeitadas e retornam erro no arquivo, o cálculo (11) retorna erro de caractere desconhecido, pois aparenta um caractere que não é numero, não é sinal de operação e não é código de expressão('i' ou 'I', 'p' ou 'P'). Já no caso de (12) a expressão retoma um cálculo com divisão por zero, então retorna um erro de não compatibilidade matemática.

Se os dados passam da filtragem inicial sem erros, são dirigidos para camada de conversão de dados, onde serão convertidos para inteiros, no caso dos números e os indicadores de expressão e operação entre as duas expressões para caracteres e colocados no formato da struct ('numeroC'). Isso permite que sejam direcionados para a camada de cálculo que reconhece a operação e executa entre as duas expressões.

a. Diferença entre TP1 e o TP2

Essa marcação tem o objetivo de destacar diferenças na entrada do TP1 para o TP2. No TP1 os cálculos eram feitos somente com valores inteiros e no TP2 continua assim, ou seja, as entradas só serão lidas se forem inteiras, senão acusará erro, por exemplo: “2.5+5i-3.5+2.4i=”. É importante salientar que essa limitação pode estar causando erro em alguns valores na conversão de polar para retangular.

Na nova versão a entrada podem ser quantos caracteres o usuário quiser na entrada, na versão anterior eram limitados a 40 caracteres, ainda assim, podem ser adicionados quantos espaços quiserem entre os caracteres válidos. Outra melhoria é que não era reconhecido um sinal entre a operação e a segunda expressão, por exemplo: “2+3i+-5-3i=”. atualmente é reconhecido e o valor é -5.

Nesse caso de operações que devem ser digitadas sempre no modelo estabelecido, continua na nova versão, em caso de ausência de valor, o usuário deve digitar zero(‘0’). O exemplo não aceito: “5i-3+5i=”. O exemplo aceito pelo programa: “0+5i+2-0i=”. A exceção está no caso de potencia, que o valor é apenas um numero real.

No próximo tópico será apresentado como a saída é processada e se torna disponível ao usuário para visualização.

V. Saída de dados

Nesse passo será apresentado como o programa processa a saída de dados, será menor que o tópico de entrada de dados, pois no caso da saída não exige processamento porque os dados estão calculados e no formato manipulável. Dessa forma, seguindo a figura 1, depois de calculado os dados retorna para a execução onde são direcionados de volta para a camada de Arquivo que cuidará para que sejam salvos em um arquivo “saída.txt”, no mesmo diretório do arquivo .exe do programa.

Exemplo de saída usando o mesmo arquivo que foi usado no tópico de entrada de dados:

-1+2i--1+2i= 0+0i;

-1+2i+1-2i= 0+0i;

-1+2i*-1+2i= 4-126p;

-1+2i/-1+2i= 1+0p;

-1+2i^2= 4-126p;

-1+2i= 2-63p;

2+4i/2= ERRO (Sinal ou Operacao entrada errada)

-1+2i/0+0i= ERRO

2+4i/2+0i= 2+63p;

1-4p= 0+0i;

0+4i= 4+90p;

0-4i= 4-90p;

-1-4p= 0+0i;

-+1+2i= 2-63p;

+-1-2p= 0+0i;

++1-2i= 2-63p;

-1+2i&2-3p= ERRO (Sinal ou Operacao entrada errada)

--1+2i= 2+63p;

OBS: Os dados são gravado usando a função ‘fprinf()’ da linguagem.

a. Diferença entre TP1 e o TP2

Não tiveram muitas diferenças na saída de arquivo, apenas o método de exibição, que no TP1 era mostrada para o usuário na janela de execução, agora é escrito em um arquivo tipo texto. Mesmo assim, ainda são exibida mensagem como o calculo que está sendo feito no momento e quando está gravando.

VI. Como utilizar

Neste tópico é mostrado um pequeno tutorial de uso do programa. Basicamente, deve ser criado um arquivo com o nome ”entrada.txt” do tipo texto, e os cálculos devem ser escritos nele com os indicadores ‘i’ para equação retangular ou ‘p’ que indica expressão polar, dois exemplos simples de expressão são:

2+3i -> o 2 representa a parte real e o 3 a parte imaginaria, pois estão no formato retangular.

1-4p -> o 1 representa o conjugado do numero complexo e o 4 é o ângulo.

As operações aceitas entre números complexos são soma, subtração, multiplicação, divisão, potencia (por um numero real), para finalizar um calculo deve ser usando o caractere ‘=’ são aceitos apenas calculo entre duas expressões, a calculadora também converte do tipo retangular para polar e de polar para retangular, nesse caso basta colocar a expressão seguida de ‘=’. Depois de escrito os cálculos no arquivo, o arquivo deve ser salvo e copiado para o mesmo diretório do programa compilado (.exe). com isso, basta executar o arquivo e esperar gerar o arquivo “saida.txt” com o calculo lido e a resposta, como exibido no tópico anterior de saída de dados.

VII. Testes

Para fazer o teste de execução foi usado o arquivo entrada.txt enviado junto com a proposta de TP2. A execução usando esse arquivo está correta e salvamento também.

Comentários

Todos os casos foram calculados e gravados de maneira rápida e sem erros no arquivo de saída.

VIII. Conclusões

O trabalho proposto teve papel importante na matéria por permitir o contato do aluno com um projeto de verdade e com os contratempos de criação de tal. Dessa forma, conclui-se que foi um trabalho que exigiu domínio do conteúdo apresentado na introdução, além de muita pesquisa no livro de linguagem C [1] e na internet. Assim, possibilitou aprender e criar algoritmos apenas usando as funções básicas com a linguagem C.

IX. Referencias

1.Backes A. **Linguagem c: completa e descomplicada.** Elsevier Brasil, 2012.