2016.2

Algoritmo e Estrutura de dados I

Jesimon Barreto Santos

[CALCULADORA COMPLEXA]

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 i, com o valor de 25 pontos na matéria. A linguagem C foi apresentada pelo professor à turma, cujo, o aprendizado foi guiado por meio de aulas com algoritmos na linguagem referente e exercícios voltados para a prática e conhecimento da linguagem. Um desses exercícios que serão avaliados é o Trabalho Prático.

II. Objetivo

Construção de 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. A entrada é baseada em uma string digitada pelo usuário e precisa ser tratado e reconhecido os valores. Esses valores fazem parte de uma expressão complexa que deve ser resolvida e apresentada ao usuário como resultado.

Um exemplo da entrada válida é:

- Forma retangular

  2+3i

- Forma polar:

  5-45p

- Expressões:

  2+3i * 5-45p =

  5-45p=

  2+3i=

III. Desenvolvimento

Inicialmente, foi feito um planejamento de acordo com o objetivo do programa, esse ajudou a ter visão prática da construção do projeto. Na figura 1 é apresentado em os passos do inicio da execução do projeto até a fim do processo. Porém, não leva em consideração o looping que garante o usuário utilizar até que não deseje mais.

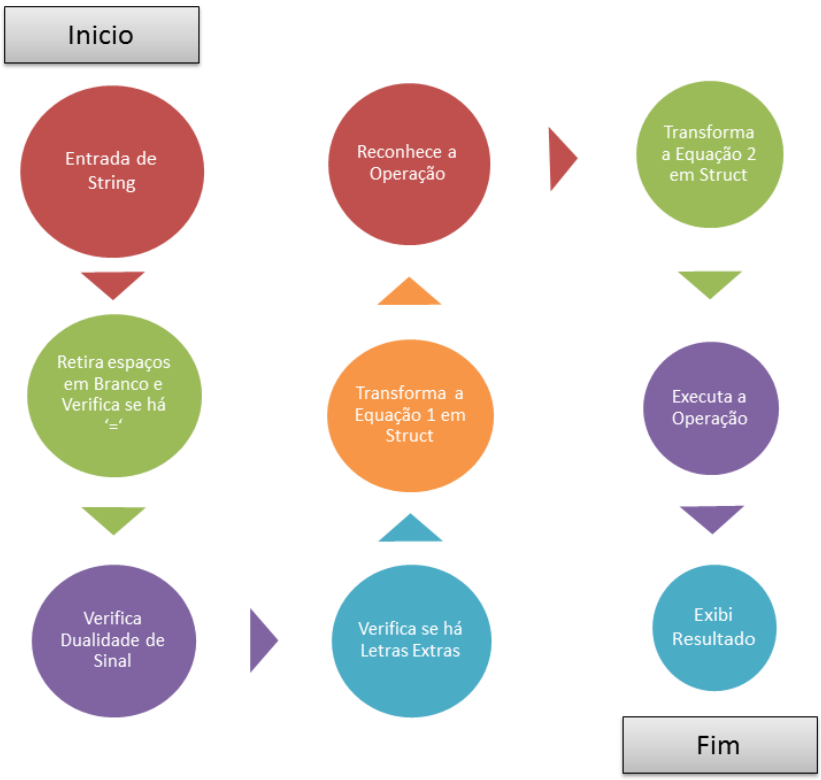

Figura 1. Planejamento inicial para execução do programa

A partir da modelagem do programa, cada circulo apresentado se tornou, na prática, uma ou mais funções que, apresentadas as estruturas e os cabeçalhos das funções, além da explicação.

IV. Estruturas usadas no Programa

Nesse programa foram definidas 2 estruturas para a auxilio no programa, a primeira foi um 'Enum', estrutura responsável por definir um tipo boolean, que é o 'true' ou 'false', definido como tipo 'bool' . Essa estrutura possibilitou reconhecer erros que são retornados pelas funções facilmente.

A segunda usada foi struct, possibilita unir vários tipos de variáveis com uma referencia. Aplicada para definir uma expressão, a explicação da estratégia usada e cada valor significa na struct é apresentado na figura 2. Quando o valor do código da struct recebe '0', acusação de erro na leitura, e ainda tem o caso de ser a operação de expoente, nesse caso, só é usado o 'primeiroValor' da 'struct expressao2' que é o valor que será elevado a 'struct expressao1'.

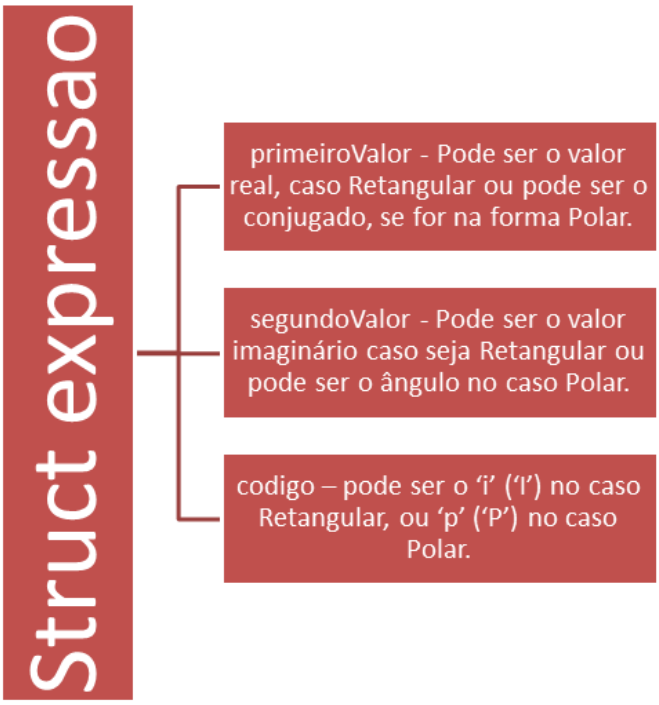

Figura 2. Apresentação da struct e explicação do significado de cada valor.

- void avisoErro(int vErro) a função avisoErro é responsável por reunir todos os erros possíveis no programa. Essa função cria uma estrutura de verificação de erros. Para essa estrutura dar certo foi criado um código para cada erro na execução do programa, muito importante no tratamento de erros, pois a medida que um erro acontece pode ser necessário futuramente incrementar algum código a medida que o erro aconteça. Na figura 3 é mostrado o diagrama de erros e os códigos.

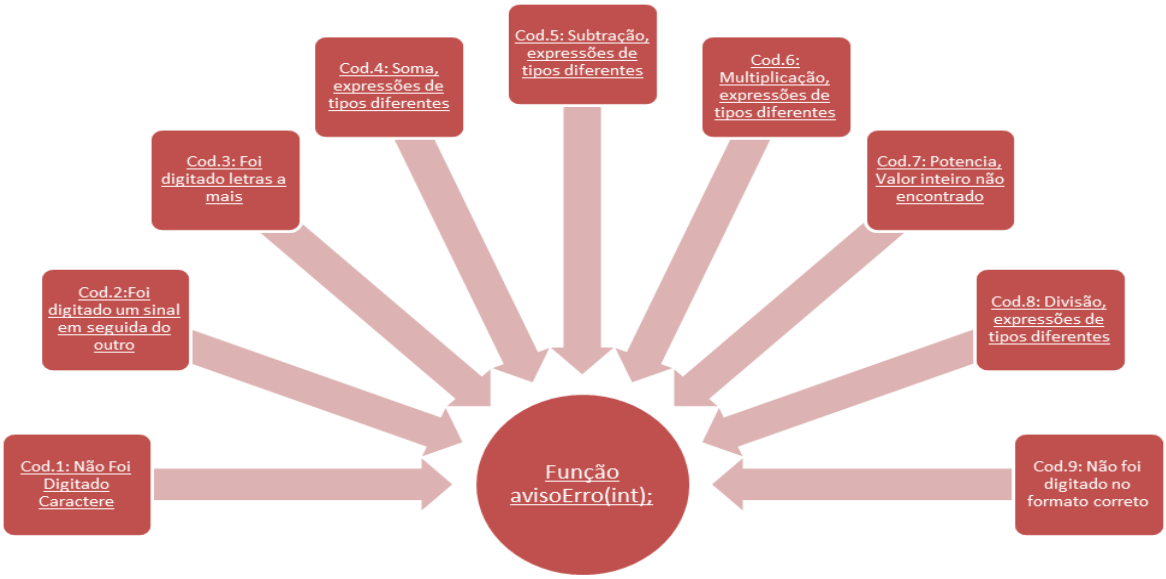

Figura 3. Imagem que apresenta os códigos de erro e a convergência para uma função.

- bool retirarEspacos(char * entrada) retira espaços tem a função principal de excluir os caracteres em branco da string principal, caso aconteça algum erro inesperado retorna true(1) ou false(0) caso não exista o caracter '=' na instrução.
- bool verificarDualidadeSinal(char * entrada) verifica se há duplo sinal na string apresentada pelo usuário, retorna true (1) caso exista sinal duplo ou false(0) caso não exista.
- bool multiIndicadorExpre(char * entrada) verifica se existem letras i ou p a mais e se existem letras diferentes dessas, caso exista retorna true (1) caso não exista retorna false(0).
- Extrair informações
    - struct expressao extrairInformacoesExpressao1(char * entrada);
    - char extrairInformacoesOperacao(char * entrada,int posicao);
    - struct expressao extrairInformacoesExpressao2(char * entrada,char operacao, int posicao);
- O grupo de funções que tem no nome "extrair informações" são voltados para a extração de cada expressão e da operação requerida pelo usuário. Ainda converte cada expressão para o tipo struct e retorna-os. No caso da Operação, é retornado um tipo 'char', valor do caractere que representa a operação entre as expressões.
- Se a expressão for potencia, ele só irá reconhecer uma sequencia de números e irá atribuir a 'primeirovalor' do 'struct expressao2', assim os outros valores estarão com 0 e '0'.
- struct expressao converterRetangParaPolar(struct expressao expressaoLocal) receber um struct do tipo expressão no formato Retangular e retorna a forma polar, se já estiver na forma polar , a função só irar retornar os valores.
- struct expressao converterPolarParaRetang(struct expressao expressaoLocal) receber um struct do tipo expressão na forma Polar e retorna a forma Retangular, se já estiver na forma polar , a função só irar retornar os valores.
- struct expressao somaRetang(struct expressao expressao1, struct expressao expressao2) precisa de duas struct do tipo expressão no formato retangular, calcula a soma e retorna uma struct com o resultado. Caso sejam de tipos diferentes de retangular irá chamar a função de erro passando o cod do erro.
- struct expressao subRetang(struct expressao expressao1, struct expressao expressao2) precisa de duas struct do tipo expressão no formato retangular, calcula a subtração e retorna uma struct com o resultado. Caso sejam de tipos diferentes de retangular irá chamar a função de erro passando o cod do erro.
- struct expressao multiPolar(struct expressao expressao1, struct expressao expressao2) precisa de duas struct do tipo expressão no formato polar, calcula a multiplicação e retorna uma struct com o resultado. Caso sejam de tipos diferentes de polar irá chamar a função de erro passando o cod do erro.

- struct expressao potenciaPolar(struct expressao expressao1, int potencia) precisa de duas struct do tipo expressão no formato polar, calcula a potencia por um natural e retorna uma struct com o resultado.
- struct expressao divPolar(struct expressao expressao1, struct expressao expressao2) precisa de duas struct do tipo expressão no formato polar, calcula a divisão e retorna uma struct com o resultado. Caso sejam de tipos diferentes de retangular irá chamar a função de erro passando o cod do erro.

Figura 4. Essa imagem exibe as operações com números complexos.

V. Resultados

Neste tópico, será apresentada a tela de execução do software como resultados da abordagem explicada nos tópicos anteriores, exibido na figura 5.

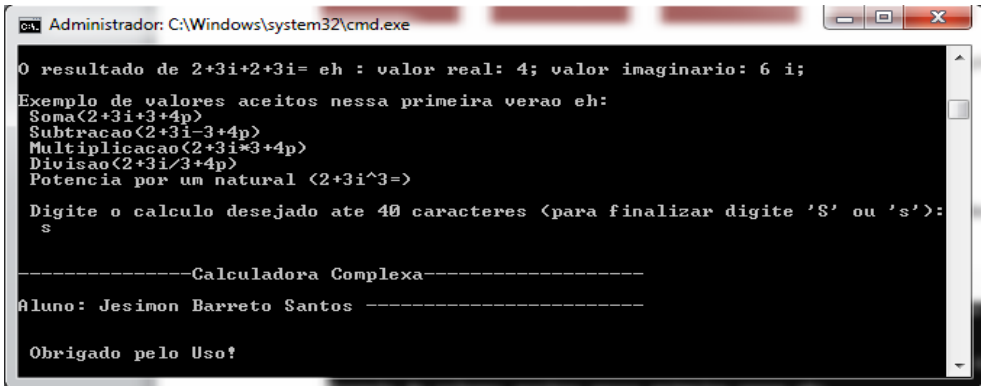

Figura 5. Exemplo do uso da calculadora

São exemplos de entradas, todas as entradas da documentação e mostradas no tópido II *objetivo* deste trabalho. Em contra partida, no tópico VII é mostrada as limitações do software, ou seja, os formatos e formas que o programa acusará erro.

VI. Conclusões

Conclui-se que foi um trabalho que exigiu domínio do conteúdo passado em sala de aula, além de muita pesquisa no livro de linguagem C [1]. Ainda assim, trouxe a possibilidade de conhecer um processo de construção de software, e entender e aplicar as ferramentas da

linguagem a favor de construir um software que execute da melhor forma possível. Porém, com o desenvolvimento é difícil aprender a desenvolver

VII. Limitações

Calculo somente com valores inteiros, ou seja, as entradas só serão lidas se forem inteiras, senão acusará erro, por exemplo: “2.5+5i-3.5+2.4i=”. Essa limitação pode estar causando erro em alguns valores na conversão de polar para retangular.

Na entrada, só são aceitos 40 caracteres, contando com espaços, caso precise de ampliação é preciso mudar o tamanho das strings na criação e o valor da variável ‘tamanhoString’ constante, criada no inicio do programa, usada como referencia para a string principal digitada pelo usuário.

Não reconhece um sinal entre a operação e a segunda expressão, por exemplo: “2+3i+-5-3i=”. Neste caso, acusará erro de repetição de sinal.

Operações devem ser digitadas sempre no modelo estabelecido, 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=”.

VIII. Referencias

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