Avaliador de expressões numéricas

O que deve ser feito

O presente trabalho prático tem como objetivo desenvolver um programa em linguagem C capaz de avaliar expressões matemáticas escritas, com suporte às operações básicas e a funções matemáticas especiais. Além da avaliação, o sistema deverá converter expressões entre as notações infixa e pós-fixa, utilizando pilhas como estrutura de dados fundamental.

Funcionalidades obrigatórias

O código-fonte deve contemplar as seguintes operações:

1. Converter expressões da notação infixa para notação pós-fixa;
2. Converter expressões da notação pós-fixa para notação infixa;
3. Implementar as operações aritméticas básicas: +, –, *, /, % e ^;
4. Implementar raiz (raiz quadrada), sen (seno), cos (cosseno), tg (tangente) e log (logaritmo decimal);
5. Considerar que os ângulos utilizados em sen, cos e tg estão em graus;
6. As operações indicadas no item D devem operar sobre apenas um operando; e
7. Avaliar corretamente as expressões de teste apresentadas na tabela a seguir.

Teste	Notação Posfixa	Notação Infixa	Valor
1	3 4 + 5 *	(3 + 4) * 5	35
2	7 2 * 4 +	7 * 2 + 4	18
3	8 5 2 4 + * +	8 + (5 * (2 + 4))	38
4	6 2 / 3 + 4 *	(6 / 2 + 3) * 4	24
5	9 5 2 8 * 4 + * +	9 + (5 * (4 + 8 * 2))	109
6	2 3 + log 5 /	log(2 + 3) / 5	Aprox. 0.14
7	10 log 3 ^ 2 +	(log10)^3 + 2	3
8	45 60 + 30 cos *	(45 + 60) * cos(30)	Aprox. 90,93
9	0.5 45 sen 2 ^ +	sen(45) ^2 + 0,5	1

Outros testes devem ser elaborados pelo aluno para validar integralmente o programa.

Critérios de avaliação

Durante a correção, serão analisados:

- O código fonte enviado pelo AVA e código-fonte compartilhado em GitHub;
- Estruturação modular do código com uso da linguagem C padrão;
- Distribuição correta dos arquivos: expressao.c, expressao.h e main.c;
- Identificação e tratamento de inconsistências nas entradas;

- Condições adequadas para execução das operações matemáticas.

Estrutura do projeto

O código deve ser organizado em **três arquivos-fonte**, a serem compilados com o comando:

gcc expressao.c main.c -o expressao.exe

O cabeçalho **expressao.h**, abaixo, **não deve ser alterado**:

```
#ifndef EXPRESSAO_H
#define EXPRESSAO_H

typedef struct {
    char posFixa[512];     // Expressão na forma pos-fixa, como 3 12 4 + *
    char inFixa[512];      // Expressão na forma infixa, como 3*(12+4)
    float Valor;           // Valor numérico da expressão
} Expressao;

char * getFormaInFixa(char *Str); // Retorna a forma inFixa de Str (posFixa)
float getValorPosFixa(char *StrPosFixa); // Calcula o valor de Str (na forma posFixa)

#endif
```

Observações

1. O aluno deverá entregar apenas o arquivo calculadora.c, contendo as implementações das funções definidas em calculadora.h, sem incluir main().

2. Podem ser criadas funções auxiliares internas em calculadora.c, mas seus protótipos não devem ser incluídos em calculadora.h.

3. Caso ocorra erro na função char *getFormaInFixa(char *Str), esta deve retornar ponteiro NULL.

4. O código deve obedecer ao padrão C (bibliotecas, alocação de memória e escopo de variáveis), compatível com Windows e compiladores como Dev-C++ ou VSCode.

5. A string de retorno de getFormaInFixa() não deve conter espaços, nem parênteses além dos estritamente necessários.

6. A correção será automatizada com o comando gcc main.c calculadora.c -o calculadora.exe. Ademais, o código deve compilar sem erros.

7. Recomenda-se a criação de novos testes além dos fornecidos, para garantir o funcionamento completo e robusto da solução.

8. Este trabalho é individual e o descumprimento dos requisitos poderá implicar em redução de nota.