

Relatório Trabalho 1 ED

Universidade de Brasília

Ian Nery Bandeira - 17/0144739

Estruturas de Dados - Prof. Caetano

- **Introdução à notação polonesa reversa**

O método utilizado para resolução das expressões matemáticas por meio de pilhas, é dependente do método de transformação da expressão de modo infixo (conhecido também como o método que escrevemos normalmente); para o método pósfixo, também chamada notação polonesa reversa. Por trabalharmos com uma lógica de pilhas, tal notação torna o algoritmo mais simples para sua solução, visto que a notação geral trabalha com “ $X Y +$ ” ao invés de “ $X + Y$ ”. Então, por este motivo, a utilizaremos na confecção do trabalho.

- **Arquitetura do sistema desenvolvido**

O método para resolução do problema, foi de primeiro implementar duas pilhas distintas, *pilhachar.c/h* e *pilhadouble.c/h*, sendo a primeira unicamente para empilhar caracteres, enquanto a segunda unicamente para empilhar números reais.

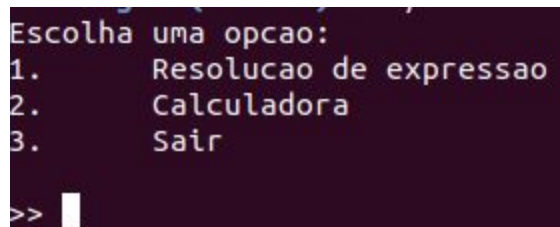
Após isto, foi necessário implementar as funções do algoritmo, sendo estas a resolução de expressões, contida nos arquivos *resprob.c/h* e a função calculadora, contida nos arquivos *calculator.c/h*.

A arquitetura do arquivo *resprob.c* foi implementada majoritariamente com as diretrizes do roteiro do trabalho, sendo que este determina os passos de: Validação da expressão, transformação de infixa para posfixa e avaliação da expressão. Junto dessas funções, foi instalada funções para checar prioridade entre operadores, se o determinado caracter é um operador ou operando, e como converter um string para um número real.

A arquitetura do arquivo calculator.c, por sua vez, foi implementada graficamente a partir das fotos presentes no roteiro, sendo que suas funções cobrem a inserção de números reais na pilha a partir de strings; operações de soma, subtração, divisão e multiplicação; operações especiais, como “!” e “c”; sendo o primeiro com o objetivo de fazer a operação com o operador que o antecede até que a pilha fique com apenas um valor, seu resultado, enquanto o segundo serve para empilhar o penúltimo valor desempilhado da pilha o número de vezes determinado pelo número do ultimo valor desempilhado da pilha.

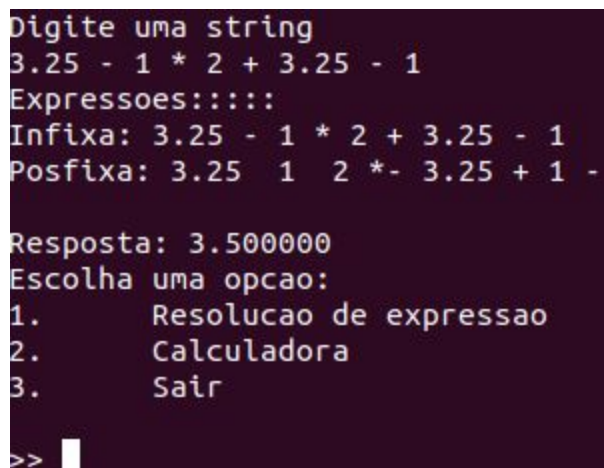
Para melhor entendimento das funções do trabalho, este documento acompanha um README.txt, que determina as instruções de compilação e as premissas assumidas para bom funcionamento do trabalho; e uma pasta contendo os arquivos html do DOXYGEN.

● Fotos do funcionamento do trabalho



```
Escolha uma opcao:
1.      Resolucao de expressao
2.      Calculadora
3.      Sair
>>
```

Menu principal do programa.



```
Digite uma string
3.25 - 1 * 2 + 3.25 - 1
Expressoes:::::
Infixa: 3.25 - 1 * 2 + 3.25 - 1
Posfixa: 3.25 1 2 *- 3.25 + 1 -
Resposta: 3.500000
Escolha uma opcao:
1.      Resolucao de expressao
2.      Calculadora
3.      Sair
>>
```

Resolvendo a expressão prevista no roteiro.

```
Digite uma string
5 + ( 5 - 5 )
Expressoes:::::
Infixa: 5 + ( 5 - 5 )
Posfixa: 5 5 5 - +

Resposta: 5.000000
Escolha uma opcao:
1.      Resolucao de expressao
2.      Calculadora
3.      Sair

>> █
```

Resolvendo expressão aleatória com parênteses.

```
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.

1
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
1. 1.000000
>> 2
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
2. 1.000000
1. 2.000000
>> 3
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
3. 1.000000
2. 2.000000
1. 3.000000
>> █
```

Empilhando valores na calculadora.

```
Digite os operadores e operandos como especificado no README.txt.  
Para sair, digite 'exit'.  
3. 1.000000  
2. 2.000000  
1. 3.000000  
>> +  
Digite os operadores e operandos como especificado no README.txt.  
Para sair, digite 'exit'.  
2. 1.000000  
1. 5.000000  
>> 
```

Mostrando como utilizar o operador soma.

```
Digite os operadores e operandos como especificado no README.txt.  
Para sair, digite 'exit'.  
6. 1.000000  
5. 2.000000  
4. 3.000000  
3. 4.000000  
2. 5.000000  
1. 6.000000  
>> *!  
Digite os operadores e operandos como especificado no README.txt.  
Para sair, digite 'exit'.  
1. 720.000000  
>> 
```

Mostrando a funcionalidade do operador “!” para calcular fatorial de 6.

```

Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
1. 1.000000
>> 5
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
2. 1.000000
1. 5.000000
>> c
Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
5. 1.000000
4. 1.000000
3. 1.000000
2. 1.000000
1. 1.000000
>>

```

Mostrando a funcionalidade do operador “c”

```

Digite os operadores e operandos como especificado no README.txt.
Para sair, digite 'exit'.
1. 5.000000
>> exit
Escolha uma opcao:
1.      Resolucao de expressao
2.      Calculadora
3.      Sair
>>

```

Mostrando funcionalidade de exit

```

Escolha uma opcao:
1.      Resolucao de expressao
2.      Calculadora
3.      Sair
>> 3
→ T1 git:(master) X

```

Mostrando opção de saída do programa.

O arquivo com o smoke test está na pasta com os arquivos-fonte do programa, junto a este relatório.