parenteses- Trabalho prático de CI1001
Tipos Abstratos de Dados e listas ligadas
TAD Pilha – uma aplicação
Departamento de Informática/UFPR
André Grégio, Luis Bona e Marcos Castilho

1 Sobre a entrega do trabalho

São requisitos para atribuição de notas a este trabalho:

- Uso de um arquivo makefile para facilitar a compilação. Os professores rodarão "make" e deverão obter o arquivo executável funcional com a sua solução. Este executável, cujo nome deverá ser parenteses, deverá estar no subdiretório parenteses;
- Ao compilar, incluir pelo menos `-Wall -Werror -Wextra`. Se não compilar, o trabalho vale zero. Haverá desconto por cada *warning*;
- Arquivo de entrega:
  - Deve estar no formato tar comprimido (.tar.gz);
  - O tar.gz deve ser criado considerando-se que existe um diretório com o nome do trabalho. Por exemplo, este trabalho é o parenteses;
  - Então seu tar.gz deve ser criado assim:
    - Estando no diretório parenteses, faça:
    - `cd ..`
    - tar zcvf parenteses.tar.gz parenteses)
  - Desta maneira, quando os professores abrirem o tar.gz (com o comando tar zxvf parenteses.tar.gz) terão garantidamente o diretório correto da entrega para poderem fazer a correção semi-automática.
  - O que colocar no tar.gz? Todos os arquivos que são necessários para a compilação, por isso se você usa arquivos além dos especificados, coloque-os também. Mas minimamente deve conter todos os arquivos `.c`, `.h` e o makefile;
  - Os professores testarão seus programas em uma máquina do departamento de informática (por exemplo, cpu1), por isso, antes de entregar seu trabalho faça um teste em máquinas do dinf para garantir que tudo funcione bem.

2 O trabalho

Você deve usar o mesmo arquivo `libpilha.h` fornecido no `tp3`, bem como a sua implementação do `libpilha.c` feito para o mesmo trabalho. O `makefile` deve ser adaptado para substituir o `testa_pilha.c` pelo arquivo `parenteses.c`. Desta forma, você deve entregar no seu `tar.gz`:

libpilha.h: o mesmo fornecido no `tp3`;

libpilha.c: o mesmo que você fez no `tp3`, porém você pode submeter o arquivo com correções, caso queira;

parenteses.c: seu programa que resolve o problema descrito abaixo;

makefile: o makefile que você usou para gerar o executável de nome `parenteses`.

O arquivo `libpilha.h` não pode ser alterado. Na correção, os professores usarão o arquivo original.

- Use boas práticas de programação, como indentação, bons nomes para variáveis, comentários no código, bibliotecas, defines... Um trabalho que não tenha sido implementado com boas práticas vale zero.
- Quaisquer dúvidas com relação a este enunciado devem ser solucionadas via email para `prog1prof@inf.ufpr.br` pois assim todos os professores receberão os questionamentos. Na dúvida, não tome decisões sobre a especificação, pergunte!
- Dúvidas podem e devem ser resolvidas durante as aulas.

3 O problema

Você deve implementar um programa que verifica se uma expressão artimética digitada pelo usuário está com os parênteses balanceados, considerando que existem alguns tipos de "parênteses" tais como os próprio parênteses (`(, )`), colchetes (`[, ]`), chaves (`{, }`), dentre outros.

Não é necessário fazer a conta, é só para verificar os parênteses. Basta não fazer nada com os números lidos.

O programa vai ler uma expressão da entrada padrão e verificar se os símbolos de precedencia (parentesis, colchetes e chaves) foram corretamente utilizados.

```
Exemplo:
{(3 + 5) * (2 + 1)*1}/[3+1]
CORRETA

(3+2)*5}
INCORRETA

{3+2)*(3+}
INCORRETA
```

Seu código deve ler a string que contém a expressão e deve verificar sua validade. Ao final imprima `CORRETA` se a expressão for válida ou `INCORRETA` caso contrário.

Bom trabalho!