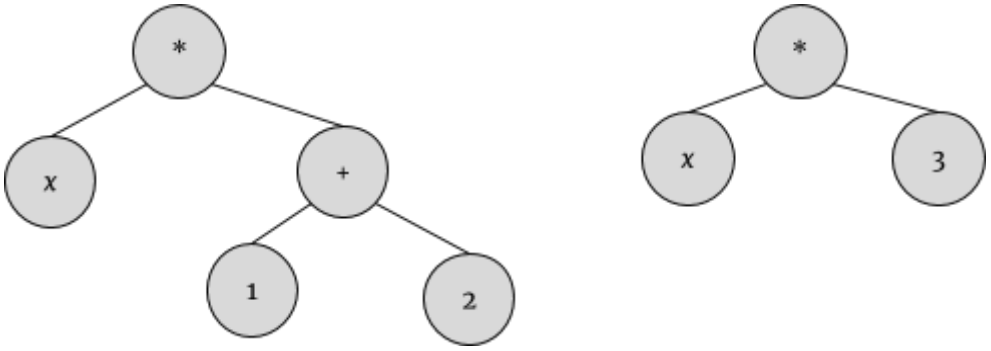
MC202E - ESTRUTURAS DE DADOS

Lab07: Otimizando Expressões

Uma companhia está desenvolvendo uma nova funcionalidade para ser adotado em suas calculadoras. Essa nova funcionalidade consiste em otimizar expressões aritméticas inteira informadas pelo usuário. As expressões aritméticas inteira são aquelas em que os operandos são valores numérico inteiro ou variáveis – e.g., ((4 * (5 - 2)) - (*x* + *y*)), que equivale a expressão otimizada (12 - (*x* + *y*)). Para implementar essa funcionalidade é necessário conhecimento em estruturas de dados. Entretanto, os programadores com tal conhecimento estão alocados a outros projetos. Como você está fazendo a disciplina de MC202, será que você consegue implementar essa funcionalidade?

Tarefa

Escreva um programa em C que recebe uma expressão aritmética inteira como entrada e gera uma **árvore binária** que representa a expressão dada. Após a construção da árvore, o programa deverá otimizar a árvore sempre que houver valores constante em alguma subárvore. A saída do seu programa deverá ser a expressão otimizada. Para ilustrar, veja o exemplo a seguir para a expressão (*x* * (1 + 2)).



A árvore binária da esquerda corresponde a árvore construída a partir da expressão dada como entrada; enquanto que a árvore binária da direita corresponde a árvore obtida após a otimização da expressão.

Entrada

A primeira linha da entrada contém um inteiro positivo m que corresponde ao número de expressões. Cada uma das m linhas seguintes contém uma expressão aritmética inteira. Os operadores aritméticos utilizados são de soma, subtração, multiplicação e divisão inteira. As expressões conterão números negativos e positivos, e cada operação está rodeada por parênteses, mesmo quando não houver ambiguidade. Além disso,

todas as variáveis são representadas por um único caractere e os parênteses, operadores e operandos estão todos separados por espaço em branco.

Restrição

- ❑ $3 \leq \boldsymbol{m}$

Como cada elemento da expressão está separado por espaço em branco, então você pode ler no formato *string* e descobrir o tipo de cada elemento, como por exemplo:

```
scanf(" %s", buffer);

if (buffer[0] == '(') { // início de alguma subexpressão
    ...
}
else if (buffer[0] >= '0' && buffer[0] <= '9') {
    valor = atoi(buffer);
    ...
}
else if (...) {
    ...
}
```

Saída

A saída do seu programa deverá conter $\boldsymbol{m}$ linhas, onde cada linha corresponde à expressão otimizada. No final de cada expressão otimizada deverá conter um espaço em branco.

Observações

- ❑ As expressões sempre poderão ser otimizados sem erros, ou seja, não há *overflow* e nem divisão por zero.

Exemplo

A grafia da saída abaixo deve ser seguida rigorosamente por seu programa, inclusive a impressão de uma linha em branco no final da saída.

Entrada

```
3
( 2 * 5 )
( ( 4 / 3) * s )
( ( 4 * ( 5 - 1 ) ) - ( x + y ) )
```

Saída

```
10
( 1 * s )
( 16 - ( x + y ) )
```

Critérios específicos

Os seguintes critérios específicos sobre o envio, implementação, compilação e execução devem ser satisfeitos.

i. Submeter no *SuSy* os arquivos:

Obrigatórios

⇒ `lab07.c`: Deverá conter a função principal para a solução do problema.

⇒ `arvore.*`: Arquivos de cabeçalho e fonte contendo respectivamente a interface e implementação da estrutura de árvore binária.

Opcionais

⇒ `*.*`: Enviar até 2 arquivos cabeçalho e 2 arquivos fonte, desde de que contribuam para a modularização da solução.

ii. **É obrigatório** implementar uma solução que utiliza a estrutura de **árvore binária**.

iii. Flags de compilação:

```
-std=c99 -Wall -Werror -g -lm
```

iv. Tempo máximo de execução: **1 segundo**.

Observações gerais

No decorrer do semestre haverá 3 tipos de tarefas no *SuSy* (descritas logo abaixo). As tarefas possuirão os mesmos casos de testes abertos e fechados, no entanto o número de submissões permitidas e prazos são diferentes. As seguintes tarefas estão disponíveis no *SuSy*:

- **Lab07-AmbienteDeTeste**: Esta tarefa serve para testar seu programa no *SuSy* antes de submeter a versão final. Nessa tarefa, tanto o prazo quanto o número de submissões são ilimitados, porém os arquivos submetidos aqui **não serão corrigidos**.
- **Lab07-Entrega**: Esta tarefa tem limite de uma **única** submissão e serve para entregar a **versão final** dentro do prazo estabelecido para o laboratório. Não use essa tarefa para testar o seu programa e submeta aqui apenas quando não for mais fazer alterações no seu programa.

- **Lab07-ForaDoPrazo**: Esta tarefa tem limite de uma **única** submissão e serve para entregar a versão final fora prazo estabelecido para o laboratório. Esta tarefa irá substituir a nota obtida na tarefa **Lab07-Entrega** apenas se o aluno tiver realizado as correções sugeridas no *feedback* ou caso não tenha enviado anteriormente na tarefa **Lab07-Entrega**.

Avaliação

Este laboratório será avaliado conforme o número de **casos de teste fechados** em que o seu programa apresentou saída correta, menos possíveis descontos referentes aos critérios de correção e de qualidade de código, os quais estão disponíveis na **planilha de notas**. Entretanto, outros critérios podem ser incorporados na avaliação desta tarefa se for julgado pertinente; e **a nota pode ser zerada caso não atender os critérios específicos.**