MC202 — ESTRUTURAS DE DADOS

Laboratório 08 — Números astronômicos

Problema

Seu amigo Nicolas matriculou-se na disciplina de matemática astronômica, na qual ele tem que calcular as distâncias astronômicas entre as estrelas. Como Nicolas já cursou MC102 anteriormente, ele tem conhecimentos básicos de programação e decidiu criar uma calculadora que permita somar e subtrair números astronômicos, mas ele percebeu que tanto o valor máximo quanto o mínimo de um int em C (-2147483648; 2.147.483.647) não é suficiente para fazer cálculos com os números de que ele precisa. Como ele sabe que você está cursando MC202, ele pediu ajuda para criar uma estrutura de dados baseada em listas ligadas, onde cada nó representa um dígito do número e que permita somar e subtrair dois números e obter o inverso aditivo de um número. Como os números astronômicos são distâncias, a calculadora só deve manipular números positivos e zero

Entrada

Cada linha da entrada representa uma operação que deve ser executada na calculadora, o formato de cada linha depende do tipo da operação.

Somar: <+> <numero>	Esta linha é composta de um caractere '+' seguido por um espaço e um número astronômico. Essa operação soma o resultado da operação anterior e o novo número inserido.
Subtrair: <-> <numero>	Esta linha é composta de um caractere '-' seguido por um espaço e um número astronômico. Essa operação faz a subtração entre o resultado da operação anterior e o novo número inserido. **O resultado desta operação nunca será um número negativo**
multiplicação <*> <numero>	Esta linha é composta de um caractere '*' 'seguido por um espaço e um número astronômico. Essa operação multiplica o resultado da operação anterior e o novo número inserido.
end: <#>	Esta linha é composta de uma string '#'. Indica o término de operações

Cada número astronômico na entrada é representado por uma string **s**. A string **s** tem um tamanho máximo de 1000. **A calculadora deve ser configurada para ter um valor inicial de 0.**

Saída

A saída contém **N** linhas, cada uma dessas linhas representa o resultado de cada uma das **N** operações executadas.

Exemplo 1

Entrada

```
+ 6
- 2
+ 97
#
```

Saída

```
6
4
101
```

Exemplo 2

Entrada

```
+ 365913494
- 773240
- 611
+ 725479065728929
- 3106
- 428
- 400109
- 945
- 850703
+ 5
+ 3505259053056540
- 53174390230346
+ 49
+ 174621603465
+ 7735912739796657
#
```

Saída

```
365913494
365140254
365139643
725479430868572
725479430865466
725479430865038
725479430464929
725479430463984
725479429613281
725479429613286
4230738482669826
4177564092439480
4177564092439529
4177738714042994
11913651453839651
```

Dicas:

- Para transformar um caractere `'8'` em um número inteiro `8`, você pode fazer uso da tabela ASCII e subtrair um `'0'`.

  ```
  char a = '8';
  int b = a - '0';
  ```

  O valor armazenado em `b` será um inteiro `8`.

- É uma boa idéia realizar a implementação de números astronômicos usando uma lista duplamente ligada (com ponteiros para frente e para trás).

- Toda vez que você executar uma operação, lembre-se de eliminar todos os 0 no início do número, por exemplo: se o número for escrito como 000000123004, você deve excluir os nós que contêm um valor zero no início para que a lista contenha 123004.

- **Não esqueça de liberar a memória.**

Critérios específicos

- Para as turmas E e F, este laboratório tem peso 3.

- Para as turmas G e H, este laboratório tem peso 2.

- Deverão ser submetidos os seguintes arquivos:
    - **numero_astronomico.h**: Interface do número astronômico.
    - **numero_astronomico.c**: Implementação do número astronômico.
    - **lab08.c**: Programa principal.
- Tempo máximo de execução: 2 segundos.

Observações gerais

No SuSy, haverá 3 tipos de tarefas com siglas diferentes para cada laboratório de programação. Todas possuirão os mesmos casos de teste. As siglas são:

1. **SANDBOX:** Esta tarefa serve para testar o programa no SuSy antes de submeter a versão final. Nessa tarefa, tanto o prazo quanto o número de submissões são ilimitados, porém arquivos submetidos aqui **não serão corrigidos**.
2. **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: submeta aqui quando não for mais fazer alterações no seu programa.
3. **FORAPRAZO:** Esta tarefa tem limite de **uma única submissão** e serve para entregar a versão final após o prazo estabelecido para o laboratório, mas com nota reduzida (conforme a ementa). O envio nesta tarefa irá substituir a nota obtida na tarefa ENTREGA apenas se o aluno tiver realizado as correções sugeridas no feedback ou caso não tenha enviado anteriormente em ENTREGA.

Observações sobre SuSy:

- Versão do GCC: C-ANSI 4.8.2 20140120 (Red Hat 4.8.2-15).
- Flags de compilação:
  `-ansi -Wall -pedantic-errors -Werror -g -lm`
- Utilize comentários do tipo `/* comentário */`;
  comentários do tipo `//` serão tratados como erros pelo SuSy.

Além das observações acima, esse laboratório será avaliado pelos critérios gerais:

- Indentação de código e outras boas práticas, tais como:
    - uso de comentários (apenas quando forem relevantes);
    - código simples e fácil de entender;
    - sem duplicidade (partes que fazem a mesma coisa).

- Organização do código:
    - tipos de dados criados pelo usuário e funções bem definidas e tão independentes quanto possível.
- Corretude do programa:
    - programa correto e implementado conforme solicitado no enunciado;
    - inicialização de variáveis sempre que for necessário;
    - dentre outros critérios.