MC202 — ESTRUTURAS DE DADOS

Lab 10 — Hashing

Déjà vu

Déjà vu é uma expressão de origem francesa que, hoje em dia, em diversas partes do mundo, é utilizada para designar a sensação de algo já visto ou vivido anteriormente, como, por exemplo, ao experimentar que já se viveu uma determinada situação com outras pessoas, em outros lugares, etc. No idioma francês, no entanto, o significado da expressão é bem simples, podendo ser traduzida literalmente como "já vi" ou "eu já vi".

Sua tarefa é "seguir" uma lista de ***chaves*** (por exemplo, nomes de pessoas, nomes de cidades, números reais, etc.) e dizer, para cada *chave*, se ela já foi vista anteriormente na mesma lista.

Evidentemente, como a tarefa é demasiadamente cansativa para ser realizada manualmente, além de depender de uma grande capacidade de memorização, caso a lista seja muito longa, você deverá implementar um programa para obter tal informação.

Entrada

Cada linha (com término em '`\n`') apresentada como entrada ao programa será uma *chave*. O programa deverá ler tantas linhas quantas forem apresentadas até que uma linha contendo apenas o caractere '`#`' seja lida. Quando este caracteres for lido, o programa pode terminar sua execução. Cada *chave* terá tamanho de no máximo 50 caracteres.

Saída

Exceto para a linha contendo apenas o caractere '`#`', para cada linha de entrada (*chave*), seu programa deverá imprimir a informação se a *chave* já foi ou não vista anteriormente na mesma lista. Seu programa deverá fazer isso seguindo o seguinte formato:

```
<bool> <chave>
```

onde `<chave>` representa a *chave* lida da entrada padrão e `<bool>` é um número inteiro que indica se a `<chave>` foi vista anteriormente na mesma lista: `1` em caso positivo e `0` em caso negativo.

Exemplo

Entrada

```
abacate
laranja
abacaxi
abacate
chave1
chave 2
chave 1
observe que uma chave pode conter espacos
chave1
chave2
melancia
larajna
abacaxi
laranja
0000
0001
0002
0001
0003
laranja
#
```

Saída

```
0 abacate
0 laranja
0 abacaxi
1 abacate
0 chave1
0 chave 2
0 chave 1
0 observe que uma chave pode conter espacos
1 chave1
0 chave2
0 melancia
0 larajna
1 abacaxi
1 laranja
0 0000
0 0001
0 0002
1 0001
0 0003
1 laranja
```

Observações

- **Você deverá fazer uso de ao menos uma função *hash* como parte da solução para o problema.**
- Você deverá submeter pelo menos o arquivo **lab10.c** no SuSy.
- Você poderá submeter outros arquivos auxiliares.
- Para as turmas A, B e C, esse laboratório tem peso 4.
- Observações sobre SuSy:
    - Versão do GCC: 4.4.7 20120313 (Red Hat 4.4.7-17).
    - Flags de compilação:
        - `-ansi -Wall -pedantic-errors -Werror -g -lm`
    - Utilize comentários do tipo `/* comentário */`;
      comentários do tipo `//` serão tratados como erros pelo SuSy
    - Tempo máximo de execução: 1 segundo por teste.
    - Evite usar o sistema como compilador ou apenas para testar as instâncias.
    - Sempre use o software **valgrind** para detectar erros decorrentes do uso incorreto da memória dinâmica **em todos os testes abertos**. Relembrando:
        - `valgrind --leak-check=full ./lab10 < arq01.in`
- Além das observações acima, esse laboratório será avaliado pelos critérios:
    - Indentação de código e outras boas práticas, tais como:
        - uso de comentários (apenas quando são relevantes);
        - código simples e fácil de entender;
        - sem duplicidade (partes que fazem a mesma coisa).
    - Organização do código:
        - arquivo de cabeçalho (`.h`) contendo definições de constantes, tipos de dados criados pelo usuário, protótipo de funções etc;
        - arquivos de implementação (`.c`) que implementam as funções definidas no arquivo de cabeçalho;
        - 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;
        - desalocar toda memória alocada dinamicamente durante a execução do programa;
        - se não realiza leitura ou escrita em blocos de memória não alocados;
        - inicialização de variáveis sempre que for necessário;
        - dentre outros critérios.