2º LABORATÓRIO de CES-11 / 2013
CTA - ITA - IEC

Objetivo: Manipulação de listas encadeadas.

Tarefa: Construção de um Dicionário

Uma das dificuldades que todo calouro enfrenta ao chegar ao ITA é dominar o dialeto próprio do H8. Como são diversas expressões desconhecidas ("bizu", "mocado", "cancerizar", "vibrão", "suga", etc.), a aclimatação vai ocorrendo aos poucos...

Seu trabalho é ajudá-lo nesse importante período através da criação do *Dicionário do Iteano*, que conterá as expressões utilizadas no H8 com seus respectivos significados.

Requisitos:

a) O programa deverá implementar seis operações diferentes:

1) Carregar base de dados. O programa deverá ler o arquivo **`DicionITA.txt`**, que contém uma base de dados inicial. Exemplo:

```
bizu|dica
mocado|pessoa muito calada
mocado|pessoa que nao participa de nada
meter gaga|estudar
cancer|perfeccionista
cancerizado|excessivamente aperfeicoado
suga|atividade que demanda muito tempo
suga|pessoa chata
suga|atividade muito dificil

```

Linha vazia final

Repare na formatação desse arquivo: somente contém letras minúsculas e caracteres não acentuados (para evitar problemas de impressão); cada linha é composta por um par "expressão|significado" (cuidado para não acrescentar espaços extras); o caractere "|" é reservado exclusivamente para separar uma expressão do seu significado; pode haver significados distintos para uma mesma expressão; as expressões não estão ordenadas.

Dicas:
a) Nesta implementação, utilize a função da operação 3, descrita a seguir.
b) Os comandos abaixo (alguns deles pertencentes à biblioteca **`string.h`**) realizam a leitura de uma linha do arquivo da base de dados, armazenando a expressão e seu significado em **`expr`** e **`signif`**, respectivamente:

```
FILE *arqEntrada;
int i,j;
char *pch;
char expr[50], signif[50], aux[101];
arqEntrada = fopen("DicionITA.txt", "r");
fgets(aux,101,arqEntrada);
aux[strlen(aux)-1]='\0';
pch = strtok(aux,"|");
strcpy(expr,pch);
pch = strtok(NULL,"|");
strcpy(signif,pch);
```

2) Consultar uma expressão. Dada uma expressão, se ela estiver presente no dicionário, imprimirá em um arquivo de saída todos os seus significados disponíveis, um por linha.
Obs: Se a expressão não existir, imprimirá "nada consta".

3) Inserir um significado. Possibilita a inclusão de dados no dicionário, através de uma expressão e um significado. Nada será impresso na saída.
Obs 1: Deve permitir a inserção de um significado para uma expressão ainda inexistente.
Obs 2: Será preciso evitar redundância de dados, ou seja, uma expressão não poderá ter significados iguais.
Obs 3: No caso de um significado repetido, o programa não fará nada.

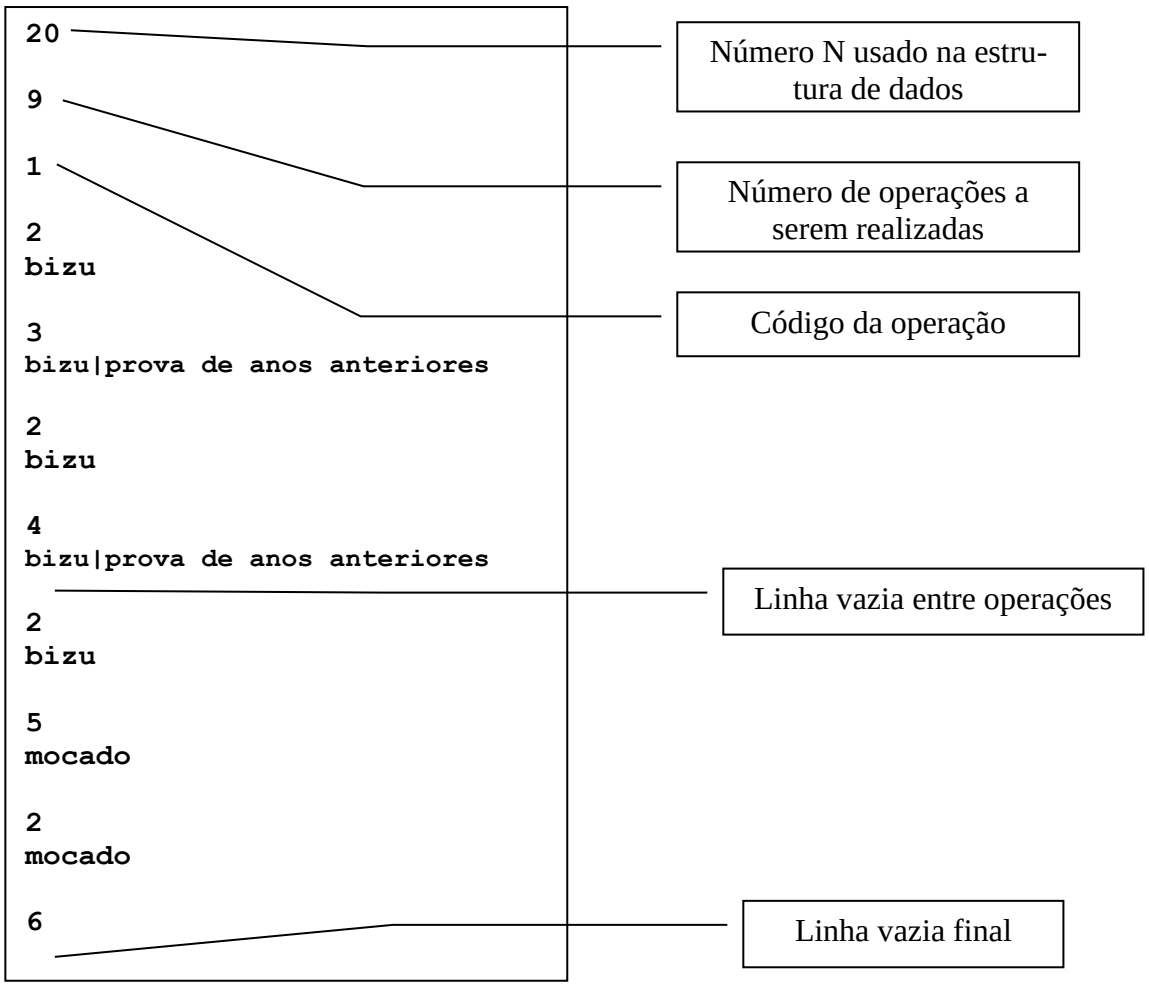
4) Eliminar um significado. Dada uma expressão e um significado, permite a exclusão desse significado. Nada será impresso na saída.
Obs 1: Se esse significado (ou expressão) não existir, o programa não fará nada.
Obs 2: O dicionário não poderá conter expressões sem significados; portanto, ao eliminar o último significado de uma expressão, deverá também eliminar essa expressão.

5) Eliminar uma expressão. Elimina uma dada expressão e todos os seus significados. Nada será impresso na saída.
Obs 1: Se a expressão não existir, o programa não fará nada.

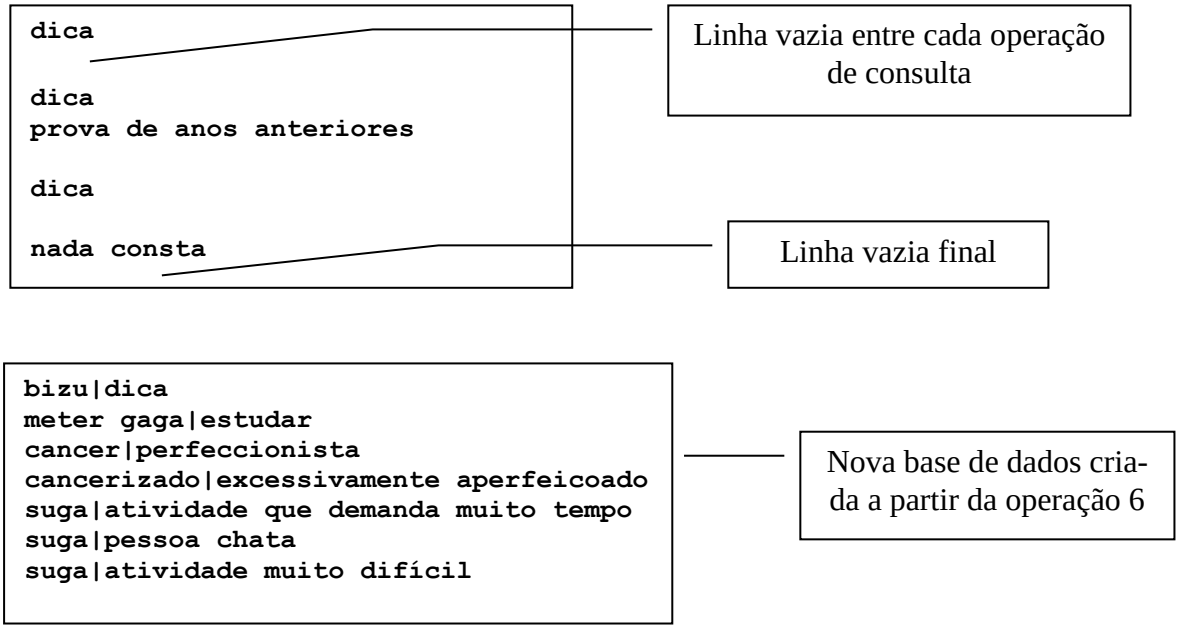
6) Gerar base de dados. Cria um novo arquivo **`DicionITA.txt`** com os dados correntes do dicionário, seguindo a mesma formatação descrita na operação 1.

Importante: É proibido o uso de variáveis globais.

b) O programa deverá processar o arquivo **`entrada.txt`**, presente no mesmo diretório do executável, que obedece ao formato abaixo (exemplo com 9 operações):



b) O programa deverá produzir o arquivo **`saida.txt`**, apenas com os resultados das eventuais operações de consulta (código 2). Veja a saída esperada no exemplo acima:



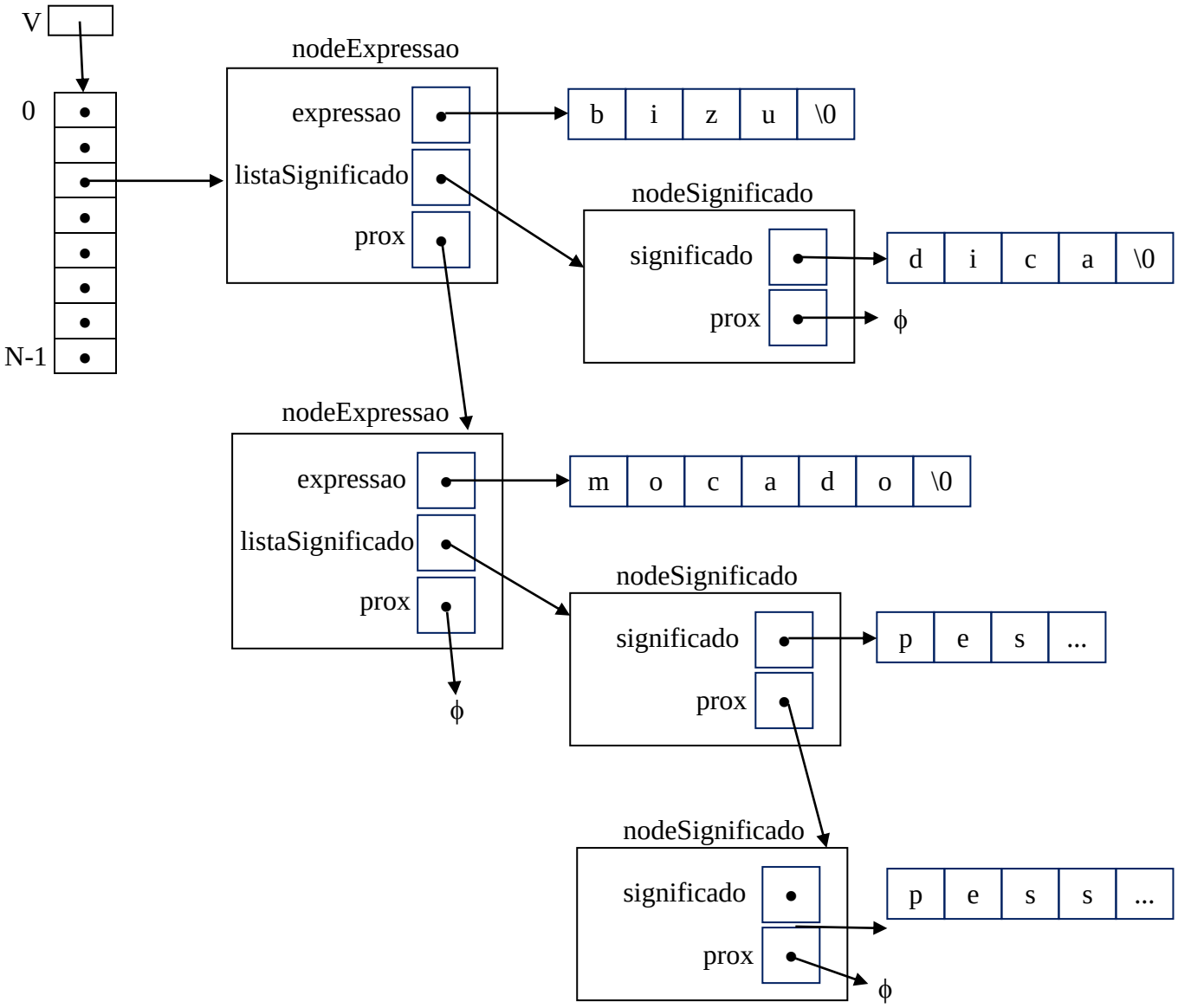
Estrutura de dados a ser utilizada:

Seu programa deverá implementar o *Dicionário do Iteano* com uma tabela de dispersão (também conhecida como *hash table*). Basicamente, é um vetor V de tamanho N apenas com ponteiros, e inicialmente vazio. Para cada expressão x do dicionário, é calculado f(x), onde $0 \leq x < N$, que será seu índice nesse vetor. f recebe o nome de *função de dispersão*. Desse modo, V[f(x)] apontará para x.

Em termos de eficiência, o ideal é que cada expressão seja associada a um único índice de V. No entanto, podem ocorrer *colisões*, isto é, a inserção de uma expressão no índice i de V, onde $0 \leq i < N$, quando V[i] não é nulo. Neste caso, será preciso criar em V[i] uma lista encadeada com todas as expressões x tais que f(x) = i.

Como função de dispersão, utilize $f(x) = [\Sigma_i \text{ ASCII}(x[i])] \% N$, onde x é uma expressão do dicionário. Seu programa, ao ler o valor N do arquivo de entrada, deverá alocar dinamicamente o vetor V. Por outro lado, cada expressão presente na tabela de dispersão estará associada a uma lista encadeada formada pelos seus respectivos significados (fique à vontade para usar ou não nó-líderes).

Veja o exemplo abaixo, onde ocorre colisão entre **`bizu`** e **`mocado`**:



Entregar (através do TIDIA):

- **Códigos fonte e executável:** esses arquivos deverão ter o nome do aluno e o número do laboratório.
- **Prazo**: 4 de setembro às 23h55. Desconto de 1 ponto na nota por dia de atraso.