## Instituto de Computação da UNICAMP

## Disciplina MC202: Segundo Semestre de 2014

## Laboratório Nº 11

*Prof. Tomasz Kowaltowski*

---

O objetivo deste laboratório é a implementação e teste de uma pequena base de dados que conterá registros referentes a alunos da UNICAMP. As informações correspondentes a cada aluno serão o RA e o nome, sendo que o RA constituirá a chave para as operações de manipulação da base de dados.

Esta base de dados será implementada através de uma tabela de espalhamento com encadeamento (*hashing with chaining*). A tabela terá *MaxHash* entradas, numeradas de *0* a *MaxHash-1* (a constante *MaxHash* está declarada no arquivo `base.h`.) A função de espalhamento a ser usada deve ser (já declarada no arquivo `base.c`):

$$f(RA) = RA \% MaxHash$$

**Importante**: Dentro de cada lista, as entradas deverão estar em ordem crescente de RA.

Analogamente aos laboratórios anteriores, a implementação seguirá a idéia de um tipo abstrato de dados realizado através de um par de arquivos `base.h` e `base.c`, e testado por um programa `principal.c`. Os arquivos `base.h` e `principal.c` são fornecidos na íntegra; o arquivo incompleto `base.c` é um esboço e deve ser completado. Os comentários esclarecem a finalidade de cada função. **Sugestão:** Inclua no arquivo `base.c` funções auxiliares, particularmente as que manipulam as listas ligadas.

Diferentemente das tarefas anteriores, o programa de teste `principal.c` recebe dois parâmetros que são os nomes dos arquivos de entrada. O primeiro arquivo (`.in1`) contém os dados iniciais a serem inseridos na base, uma linha para cada aluno, com um espaço em branco separando o RA do nome. O segundo arquivo (`.in2`) contém as operações a serem executadas. A saída do programa de teste continua sendo para o arquivo padrão de saída. Desta maneira, a execução do programa de teste pode ser invocada por comandos da forma:

```
./principal  arq.in1  arq.in2
```

ou

```
./principal  arq.in1  arq.in2 > arq.res
```

onde `principal` é o arquivo executável gerado pela compilação e `arq.res` é o arquivo que conterá os resultados, conforme indicado pela redireção de saída.

As operações (ações) interpretadas pelo programa de teste são:

| | | | |
|---|---|---|---|
| **i** | *ra* | *nome* | insere o aluno correspondente na base de dados |
| **r** | *ra* | | remove o aluno correspondente da base de dados |
| **c** | *ra* | | consulta a base de dados |
| **w** | | | imprime a base de dados |

Note que o programa principal cria uma base inicialmente vazia (chamando a função *CriaBase*) e a coloca na variável *p*. A página deste laboratório apresenta alguns conjuntos de arquivos de testes e os seus respectivos resultados.

**Observações**

1. Um **RA** sempre será **representado por um número inteiro** e **não** é precedido das letras `RA`.

2. A representação da base como uma tabela de espalhamento está escondida do usuário que deve enxergá-la apenas como um *tipo abstrato de dados*. Note no arquivo `base.h` que isso é feito em *C* definindo o tipo `Base` como um apontador do tipo `void`. A declaração do tipo da tabela (*ImplBase*) ficará escondida no arquivo `base.c` onde as conversões de tipo apropriadas (*casting*) deverão ser efetuadas.

3. Note que a implementação da tabela inclui o campo *numregs* que indica o número de registros contidos na tabela.

4. **Não é permitida** nenhuma modificação das declarações de tipos ou funções já completas do arquivo `base.c`.

5. A administração de memória deve utilizar, em lugar das funções `malloc` e `free`, as macros `MALLOC` e `FREE` definidas no pacote modificado `balloc`. As chamadas das macros são análogas às respectivas funções. Com isto, ao final da execução do programa principal, será verificado se foi liberada toda a memória dinâmica.

6. Veja no programa principal como é feita a passagem de parâmetros na linha de comando na linguagem *C*.

7. Os arquivos de entrada serão lidos até que seja detectada a condição de fim de arquivo (`EOF`).

8. Deve ser submetido apenas o arquivo `base.c` que implementa as funções especificadas no arquivo `base.h`, além de funções e tipos auxiliares.

9. O número máximo de submissões é 10.

---