Instituto de Computação da UNICAMP

Disciplina MC202: Segundo Semestre de 2014

Laboratório Nº 09

Prof. Tomasz Kowaltowski

O objetivo desta tarefa de laboratório é a implementação e teste de uma fila de prioridades (FP). A representação da fila será sequencial adotada para *árvores binárias quase-completas*.

A implementação seguirá novamente a idéia de um tipo abstrato de dados realizado através de um par de arquivos `heap.h` e `heap.c`, sendo que os elementos da fila serão apontadores para valores de tipo desconhecido. A comparação desses elementos será realizada por uma função passada como argumento quando da construção da fila — o arquivo `heap.h` contém a especificação exata das funções de interface.

A implementação será testada por um programa `principal.c` que usará a fila para ordenar um conjunto de dados referentes a alunos, com a mesma estrutura das tarefas anteriores. O RA do aluno será usado como chave para comparações, sendo que o elemento da raiz da FP será o de RA mais baixo; este efeito será produzido pela função *comparaAlunos* passada pelo programa principal. O programa principal aceita os seguintes comandos:

i ***ra nome***	insere um novo registro de aluno com *ra* e *nome* na FP;
r	remove e imprime o registro da raiz da FP;
p	imprime e remove todos os dados contidos na FP, em ordem natural desta estrutura (crescente de RA);
l	libera espaço ocupado pela FP (mas não por seus elementos);
n	imprime o número de elementos da FP;
v	verifica a estrutura da FP;
d	imprime a FP sob a forma de uma árvore binária "deitada" (depuração);
h	imprime o resumo dos comandos;
x ou **q**	encerra a interpretação;
#	linha de comentário.

Observações:

1. No caso desta tarefa, o tipo *Aluno* está declarado dentro do próprio programa principal.

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

3. A representação da FP está escondida do usuário que deve enxergá-la apenas como um *tipo abstrato de dados (TAD)*. Isto é feito no arquivo `heap.h` definindo o tipo `Heap` como um apontador para o tipo `void`. A declaração do tipo usado para implementação ficará escondida no arquivo `heap.c` onde as conversões de tipo apropriadas (*casting*) deverão ser efetuadas, se necessário (algumas são automáticas).

4. Para as operações de inserção e remoção na FP é importante que sejam seguidas as idéias explicadas em aula e expostas nas transparências e na apostila. Em particular, a função *CriaInicializaHeap* deve usar a construção mais eficiente.

5. O programa principal não testa a função *CriaInicializaHeap* — ela poderá ser utilizada numa tarefa posterior.

6. A função *Elemento* incluída em `heap.h` não faz parte, conceitualmente, do TAD pois ela pressupõe uma implementação sequencial. Será usada somente para fins de verificação da estrutura.

7. Devem ser utilizadas as macros *MALLOC* e *FREE* fornecidas com o pacote *balloc*. A sua implementação não pode utilizar as operações habituais de alocação.

8. O número máximo de submissões é 10.

Last update: "enunc.html: 2014-10-23 (Thu) 14:15:44 BRST (tk)"