ICMC-USP

SCC0603 – Algoritmos e Estruturas de Dados II
Profa. Rosane Minghim – 1º Semestre 2012

Trabalho 4 – Árvore B

Este trabalho consiste na criação e impressão de um índice de chaves como Árvore B. Assim como no trabalho anterior (3), esse índice pode ser utilizado para facilitar a manipulação de registros de um arquivo muito grande para ser mantido completamente na memória primária. Novamente implementaremos uma versão simples de um conceito muito utilizado na prática em sistemas de gerenciamento de bancos de dados.

A entrada é similar à do trabalho 3 e consistirá em um conjunto de referências bibliográficas (o nosso banco de dados). Nesse caso, no entanto, você deverá imprimir como saída a Árvore B resultante.

ATENÇÃO: Não serão realizadas operações na árvore; ela deverá ser apenas gerada e impressa na saída padrão.

1 Detalhes da entrada

Abaixo segue um exemplo de entrada e a explicação de cada parte:

```
3
2
Leung, Joseph Y.-T./A new algorithm for scheduling periodic, real-time
  tasks/Algorithmica/4/2/1989/209-219/0178-4617
Jansen, Klaus/Scheduling Malleable Parallel Tasks An Asymptotic Fully
  Polynomial Time Approximation Scheme/Algorithmica/39/1/2004/59-81/
  0178-4617
```

A primeira linha fornecea ordem D da árvore. A segunda linha fornece o número N de referências que formam o banco de dados. As N linhas seguintes são as referências; cada referência possui vários campos, que estão no formato:

```
author/title/journal/volume/number/year/pages/issn
```

Observe que não há espaços em branco ou quaisquer outros símbolos separando cada campo, apenas o "/". Além disso, não há quebras nas linhas (aqui no enunciado as linhas estão quebradas para que caibam na página). O `ISSN` é um número que identifica de forma única um *journal* no mundo.

2 Detalhes da saída esperada

Você deverá imprimir o índice (Árvore B) de volta na saída padrão após a sua geração completa. Cada nó (página) da árvore deve ser impresso em uma linha, precedido pelo seu nível. Consideraremos a raiz como sendo nível 0, seus filhos como nível 1 e assim por diante. A ordem deve seguir uma busca por largura, ou seja, todos os nós de nível 1, seguidos de todos os nós de nível 2, etc. Além disso, entre os nós de um mesmo nível, deve ser seguida a ordem crescente das chaves. Por fim, cada linha (nó) é composta pelas chaves em ordem crescente e separadas por espaços.

Um exemplo de saída de uma árvore de ordem 3 (não equivalente ao exemplo de entrada) seria:

```
0 0178-4617/048/001/0037-0066
1 0178-4617/021/001/0119-0136 0178-4617/042/001/0001-0002
1 0178-4617/055/004/0666-0702 0178-4617/062/003/0982-1005
```

Observe que é necessário incluir 0's para padronizar a saída. Dessa forma as comparações de string funcionarão corretamente. Os formatos de cada campo são como a saída do comando abaixo:

```
sprintf(key, "%04d-%04d/%03d/%03d/%04d-%04d", issn1, issn2, volume,
  number, page_ini, page_end);
```

3 Implementação

Os recursos disponíveis (memória e processamento) para o trabalho serão extremamente limitados; ou seja, não será possível carregar todos os dados na memória para realizar as operações. Você deverá implementar o índice na memória carregando as referências/operações uma de cada vez.

4 Observações

- A entrada foi formatada especificamente para facilitar a leitura com a função `strtok`. Exemplo:

  ```
  fgets(line, MAX_LINE_CHARS, stdin);
  author = strtok(line, "/");
  title = strtok(NULL, "/");
  ...
  ```

- Ao manipular strings, cuidado com as operações de comparação. Uma chave com volume "4", por exemplo, pode ser considerada maior do que uma chave com volume "35". A função "strcmp"compara os caracteres um a um até achar um que seja maior. Para reforçar essa necessidade e evitar esse problema, neste trabalho é necessário imprimir os 0's de cada campo da chave.