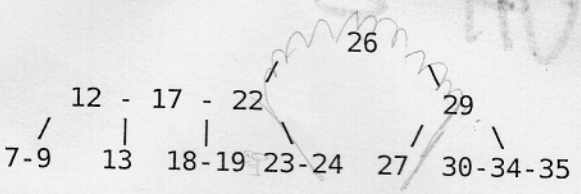
Prova 2 de Algoritmos e Estruturas de Dados III - CI057 - Turma A
Prof. Fabiano Silva
Data: 06/06/2014

Questão 1: (30 pontos)

Dada a árvore B abaixo, de grau mínimo igual a 2.

a) Inclua as chaves, 20, 21, 28 e 31, nesta ordem, e mostre a árvore B resultante após cada inserção. A árvore poderá ser dividida antes da inclusão, para que os nós abriguem as chaves corretas.



b) Porque não podemos incluir o elemento 25 direto na raiz, mesmo que o nó possua espaços livres?

Questão 2: (40 pontos)

Considere a estrutura abaixo para representar uma árvore TRIE N-ária. As chaves armazenadas na árvore são compostas pelas letras {a-z}. Cada nó da estrutura possui 26 apontadores para os filhos, sendo que a posição 0 do vetor de apontadores representa a letra 'a', a posição 1 representa a letra 'b', e assim por diante. A função ord(x) retorna o valor do índice do vetor que corresponde ao caracter x. Por exemplo: ord('a')=0, ord('b')=1, ord('x')=24.

```
typedef struct tNo *ApontadorNo;
typedef struct tNo {
        char *valor;
        ApontadorNo ponteiros[26];
} tNo;
```

Escreva um algoritmo em C ou em pseudocódigo semelhante a C que recebe como entrada uma árvore TRIE que utiliza a estrutura acima e uma chave, e retorna a chave sucessora da chave passada como parêmetro. A chave sucessora é dada pela próxima chave na ordenação lexicográfica das chaves armazenadas na árvore. Por exemplo, se as seguintes chaves estão armazenadas na árvore: c-a-m-a-d-a , c-a-n-a , c-a-m-a , o sucessor de c-a-m-a é c-a-m-a-d-a .

Questão 3: (30 pontos)

Projete uma função de hash para um conjunto de aproximadamente 16000 números inteiros positivos de 32bits. O espaço disponível para a tabela hash é de no máximo 4096 posições. A grande maioria dos números do conjunto está concentrada no intervalo de valores [17000..30000]. Explique como a sua função minimiza o número de colisões.