Universidade Federal de Santa Catarina - Curso de Sistemas de Informação

Disciplina Estruturas de Dados - Turmas 338A e B - Prova 1 – 2008/1

1. Se balanceamos uma árvore binária (com o algoritmo estudado em aula), o resultado das 3 caminhadas seria o mesmo tanto na árvore original quanto na árvore balanceada? Por quê? (2 pts)

2. Um hash tradicional é composto por uma tabela de espalhamento em um array de listas (ou seja, o desenho tradicional de hash). Imagine que, em vez do array, alguém resolve implementar uma lista encadeada para a tabela de espalhamento, na expectativa de flexibilizar a coisa. Faz sentido (o cara é um gênio) ou é uma bobagem (é um bobo alegre)? Analise os aspectos que julgar importante (isto também será avaliado na questão). (2,5 pontos)

3. Pense numa situação parecida com a acima, mas em vez de trocar o array tabela de espalhamento por uma lista encadeada (a tabela fica como a original), mas o cara trocou o array de listas por um array de árvores binárias de busca. Faz sentido? Analise os aspectos que julgar importante. (2,5 pontos)

4. Uma *fila de prioridades* é uma estrutura de dados em que cada elemento possui uma prioridade associada (por exemplo, de 0 a 10, sendo 0 a menor prioridade e 10 a maior). Esta prioridade definirá quem sairá da fila em cada operação de saída (o elemento que possuir a maior prioridade dentre todos os presentes em cada instante é o elemento que sairá, se houver mais de um com a mesma prioridade, sairá um deles, tanto faz qual). Dada a classe abaixo que implementa parcialmente uma fila circular, implemente o código do método "sairDaFila" que transformaria esta classe em uma *fila de prioridades (e, caso seja necessário, indique as alterações em outros métodos). (4 pts)*

```
class fila {
int [ ]  dados;
int inicio; int fim; int max;
int contador;
boolean filaCheia() {
return contador == max;
}
```

```
/* construtor */
fila (int tam) {
dados = new dados[tam];
max = tam;
inicio = 0; fim = -1;
contador = 0;
}
```

```
void entrarNaFila(int elem) {
if ( ! filaCheia() {
fim = fim + 1;
if (fim > max) { fim=0; }
dados[fim] = elem;
contador++;
} else {   exceção...
```