Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Algoritmos e Estruturas de Dados II (Turmas N, W, F)
1º Semestre de 2012

3ª Lista de Exercícios (parte 1)

1. Sejam N registros armazenados em um vetor A. Proponha um algoritmo para ordenar o vetor em tempo linear O(n) que não necessite de espaço adicional para os seguintes casos. i) Todas as chaves são ou 0 ou 1; ii) Todas as chaves são números inteiros entre [0,..,k] onde k é uma constante.
2. Use um método de ordenação instável A para criar um novo método de ordenação que seja estável e que apresente a mesma complexidade do algoritmo A.
3. Qual método roda mais rápido em um vetor com chaves idênticas: seleção ou inserção? Justifique sua resposta.
4. Um trabalhador de um depósito precisa rearranjar todas as caixas disponíveis no estoque de acordo com algum critério. Neste caso, o custo de comparações é pequeno comparado com o custo da troca de posições entre as caixas. Há espaço (extra) suficiente para apenas uma caixa. Qual método de ordenação deveria ser utilizado nesta situação.
5. Execute o heapsort no seguinte vetor de entrada (Indique os passos intermediários utilizando a representação por árvore).

X	T	O	G	S	M	N	A	E	R	A	I

6. Altere o código do quicksort visto em sala para contemplar a estratégia que utiliza a mediana de três elementos para escolha do pivô.
7. Considere o seguinte algoritmo (v: vetor de entrada, e: índice da esquerda, d: índice da direita do vetor).

```
void funcao(Item *v, int e, int d) {
int m = (d + e) / 2;

   if (d <= e)
      return;
   funcao(v, l, m);
   funcao(v, m+1, d);
   aux(v, e, m, d);
}
```

Escreva o código para função aux de modo que o algoritmo mostrado ordene os elementos no vetor v.

8. Descreva como você modificaria o RadixSort para ordenar cadeias de caracteres de tamanho diferente. Por exemplo, a chave “carrega” deve estar antes de “carregamento” e depois de “borboleta”.
9. Qual algoritmo de ordenação você usaria para cada um dos seguintes casos:
    a) A ordenação original de elementos com chave idêntica precisa ser mantida.
    b) O tempo de execução não deve apresentar grandes variações para nenhum caso.
    c) A lista a ser ordenada já está bem próxima da ordem final.
10. Modifique o algoritmo de ordenação por inserção de forma que ele utilize busca binária para encontrar a posição de inserção de um elemento no vetor destino. Considerando o número C(n) de comparações efetuadas, determine a complexidade do algoritmo obtido.