UNIVERSIDADE PORTUCALENSE
INFANTE D. HENRIQUE

Algoritmia Aplicada

Departamento de Inovação, Ciência e Tecnologia

Laboratório 8

Pesquisa

1. Escreva um procedimento, POSICOES (A, N, V, POSIC), que devolva num *array*, POSIC, todas as posições em que uma chave, V, ocorre num *array,* A, de N+1 posições (a posição N+1 é usada para colocar antecipadamente a chave V - função de sentinela). Admita que dispõe de uma função, PESQUISA (A, V, POS), que devolve (por pesquisa sequencial) a primeira posição (posição > POS) em que a chave V ocorre no *array* A.

2. Na pesquisa pode usar-se um método auto-organizativo: cada vez que um elemento é procurado, é movido para o início da estrutura. Implemente este método de pesquisa, considerando que os dados são números inteiros e estão armazenados num *array*.

3. Considere uma lista em que os elementos mais procurados são mantidos no início. Para o efeito, cada vez que um registo é procurado, deve incrementar-se um contador; se o valor deste igualar o dobro do valor do contador do registo anterior, deverá o registo ser recolocado na posição relativa, por ordem dos contadores. Implemente os algoritmos de pesquisa e de recolocação nesta estrutura, admitindo que são utilizados apontadores e que as chaves são caracteres.
Observe o que acontece ao ser procurado P no exemplo seguinte:

ANTES: A, X, B, C, M, P, Q, R, valendo os contadores, 10, 7, 9, 3, 2, 3, 1, 0.
DEPOIS: A, X, B, P, C, M, Q, R, valendo os contadores, 10, 7, 9, 4, 3, 2, 1, 0.

4. Implemente o método de pesquisa binária interpolada sobre um *array*, N, de nomes de 40 caracteres. Use, como apontadores, as variáveis, E, D e P, respectivamente para a esquerda, direita e ponto de partição.

5. Escreva um algoritmo que implemente a colocação de nomes numa tabela de *Hashing,* com entradas entre 0 e 999. Use a função de *Hashing* que entender e resolva as colisões por exploração linear.

6. Escreva um algoritmo que permita suprimir entradas da tabela referida no problema anterior.

7. Relativamente ao *Hashing*, responda às questões seguintes:

a) Insira as chaves numéricas, 28 , 2 , 44 , 36 , 38 , 8, numa tabela de *Hashing* de 10 posições, numeradas de 0 a 9, considerando a função, H (CHAVE) = CHAVE mod 10. As colisões devem ser resolvidas por *Re-Hashing*, usando a função, H1(CHAVE) = H(CHAVE) + 2.

b) Quantas comparações são necessárias para pesquisar cada uma das chaves, 27, 38 e 64?

8. Relativamente ao *Hashing*, responda às questões seguintes:

a) Insira as chaves numéricas, 1, 10, 11, 9, 18, 8, numa tabela de *Hashing* de 10 posições, numeradas de 1 a 10, considerando a função H (CHAVE) = CHAVE mod 10 + 1. As colisões devem ser resolvidas por *exploração linear*.

b) Quantas comparações são necessárias para pesquisar cada uma das chaves, 36, 38 e 8?

9. Considere as seguintes chaves e respectivos valores de função de *Hashing*:

UO	SD	NP	TC	PV	OR	OM	OP
6	8	13	18	20	17	6	12

AF	CS	AE	EM	UN	SF	SN	SA
17	7	19	4	2	5	4	19

a) Coloque as chaves num *array* de 20 posições, resolvendo as colisões pelo método do endereçamento aberto-exploração linear.

b) Repita, mas usando uma segunda função que tem por valor a soma dos dígitos da posição gerada de cada vez.

c) Escreva um algoritmo para realizar a tarefa da alínea b).

10. Considere um *array*, CHAVE, como base para um algoritmo de *Hashing*, com encadeamento separado com listas, onde as chaves são nomes. Para o efeito dispõe de uma função, H(NOME), que calcula o endereço no *array* CHAVE. Escreva uma função que efectue a pesquisa de um nome naquela estrutura.