SCC0502 - Algoritmos e Estruturas de Dados I

Prof. Diego Furtado Silva
Departamento de Ciências de Computação (SCC)
Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo

Lista de exercícios 01

1. O TAD dicionário possui as seguintes operações principais:

   Insere(k, D). Insere um elemento de chave igual a k no dicionário D;
   Remove(k, D). Remove o elemento de chave igual a k do dicionário D;
   Pesquisa(k, D). Retorna o elemento de chave igual a k do dicionário D.

   Assumindo que a chave k é primária, discuta os prós e contras e as complexidades de tempo de execução desse TAD utilizando as seguintes implementações:
   a. Um vetor não ordenado;
   b. Um vetor ordenado pela chave k;
   c. Uma lista ligada ordenada pela chave k;

2. Discuta a complexidade das operações de inserção e remoção em cada uma das implementações realizadas. Como tomar decisões de utilização de cada implementação a partir dessa análise?

3. Implemente uma operação de inserção ordenada em listas ligadas simples e duplas. Qual é a complexidade dessas operações?

4. Implemente uma operação ApagaLista capaz de apagar uma lista ligada, liberando a memória alocada para cada nó. Essa operação tem a mesma implementação para listas ligadas simples e duplas?

5. Identifique e demonstre com diagramas quais são os casos especiais (e como devem ser tratados) das operações de inserção e remoção em listas com as seguintes implementações:
   a. Vetor (implementação estática)
   b. Lista (simplesmente) ligada com ponteiros de início e fim

c. Lista (simplesmente) ligada com sentinelas
d. Lista duplamente ligada (com ponteiros de início e fim)
e. Lista duplamente ligada circular (com ponteiros de início e fim)

Discuta a complexidade de cada uma dessas operações.

6. Escreva uma função que rearranja os elementos de uma lista dada de forma que os elementos em posições pares sejam colocados antes de todos os elementos de posições ímpares. Preserve a ordem dos elementos pares entre si, e a ordem dos elementos ímpares entre si.

7. Uma palavra é um palíndromo se a sequência de letras que a forma é a mesma seja ela lida da esquerda para a direita ou da direita para esquerda. Exemplos: arara, raiar, omissíssimo. Escreva uma função palíndromo de maneira que, dada uma palavra, retorne true caso a palavra seja um palíndromo, e false caso contrário. Para isso utilize listas dinâmicas e discuta as diferenças nas implementações simplesmente e duplamente ligadas.

8. Discuta e implemente os tipos (TipoItem, TipoNo, TipoLista, etc) para cada uma das variações de lista discutidas em sala.

9. Critique a função abaixo. Ao receber uma lista encadeada com cabeça e um inteiro x, ela promete devolver o endereço de uma célula com conteúdo x. Se tal célula não existe, promete devolver NULL.

```
celula *busca (int x, celula *ini) {
  int achou;
  celula *p;
  achou = 0;
  p = ini->prox;
  while (p != NULL && !achou) {
     if (p->conteudo == x) achou = 1;
     p = p->prox; }
  if (achou) return p;
  else return NULL;
}
```

10. Escreva um programa que verifica se expressões aritméticas estão com o correto uso de parênteses. Seu programa deve verificar expressões para ver se cada “abre parênteses” tem um “fecha parênteses” correspondente. Utilize uma pilha.

11. [Desafio] Considere o cenário em que tenhamos uma lista dinâmica não-ordenada e queremos transformá-la em uma lista ordenada. Discuta as implementações da função para ordenar essa lista considerando lista simplesmente e duplamente ligada (usar o algoritmo de inserção e, caso estiver familiarizado com ele, o algoritmo de intercalação/mergesort).