Ciência da Computação
Campus Arapiraca

Aula 07: Listas

Prof. Dr. Rodolfo Carneiro
rodolfo.cavalcante@arapiraca.ufal.br

Introdução

- Uma das formas mais simples de interligar elementos de um conjunto
- Itens podem ser acessados e inseridos ou retirados de qualquer posiçaõ
- Duas listas pode ser concatenadas para formar uma lista única
- Uma lista pode ser partida em duas ou mais listas

Introdução

Operações comuns:

- criar lista vazia
- verificar se a lista está vazia
- inserir novo item em uma posição
- retirar um item em uma posição
- localizar um item para examinar e/ou alterar o conteúdo de seus componentes
- combinar duas ou mais listas em uma lista única
- partir uma lista em duas ou mais listas
- fazer uma cópia da lista
- ordenar os itens da lista em ordem ascendente ou descendente
- pesquisar a ocorrência de um item com um valor particular
- Imprimir os itens da lista na ordem e ocorrência

Introdução

- Pilhas e filas são tipos especiais de listas lineares
- Particularidades na implementação das operações de inserção e retirada de elementos
- Pilhas são listas lineares em que inserções, remoções e acessos a elementos ocorrem em apenas uma das extremidades
- Filas são listas lineares em que todas as inserções de novos elementos são realizadas em uma das extremidades da lista e as remoções são feitas na outra extremidade
- Listas lineares tem inserções e retiradas em qualquer posição da lista

Introdução

- Da mesma forma que as filas e pilhas, existem várias estruturas de dados que podem ser utilizadas para implementar listas lineares
- Cada uma com suas vantagens e desvantagens
- Durante o projeto de implementação de listas é importante identificar as operações que serão mais frequentes
- Não existe uma única implementação eficiente para todas as operações
- Por exemplo, não há implementação eficiente para ambas as operações a seguir:
  - ter acesso fácil a um item em qualquer posição
  - inserir ou remover elementos em uma posição i qualquer

Introdução

- A operação 1 é mais eficiente com implementação por meio de arranjos ou estruturas auto-referenciadas?
- E a operação 2?

Introdução

- A operação 1 é mais eficiente com implementação por meio de arranjos
  - Por meio da alocação sequencial em memória, acessar um índice do vetor é uma operação muito rápida
  - Se utilizarmos uma estrutura auto-referenciada, precisamos percorrer todas as células da estrutura até a célula i
- Mas se eu precisar inserir um elemento em uma posição i qualquer utilizando um array (operação 2)?

Introdução

- Mas se eu precisar inserir um elemento em uma posição i qualquer utilizando um array (operação 2)?
  - É preciso realocar todos os itens antes de i
- Nesse caso, a implementação com estruturas auto-referenciadas é mais eficiente
  - É preciso apenas criar uma nova célula e fazer x_{i-1} apontar para x_i e x_i apontar para x_{i+1}

Implementação com Vetor

- Itens da lista são armazenados em posições contíguas de memória
- Inserção de um novo item pode ser realizada após o último item com custo constante
  - Assim como nas pilhas e filas
- No entanto, a inserção de um novo item no meio da lista requer um deslocamento de todos os itens localizados após o ponto de inserção
- Retirar um item do início da lista requer um deslocamento de itens para preencher o espaço deixado vazio

Implementação com Vetor

- Assim como nas implementações anteriores, um vetor é utilizado para armazenar os itens da lista
- Da mesma forma, é necessário que esse vetor tenha tamanho suficiente para comportar os itens da lista
- Esse tamanho deve ser fixo ou não deve precisar ser alterado com frequência

Implementação com Vetor

- Implementação

Implementação com Vetor

Considerações

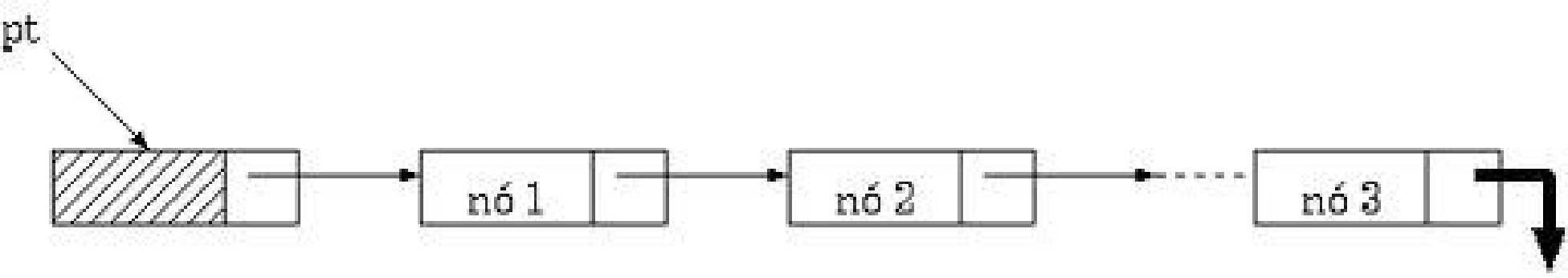
- Tem como vantagem a economia de memória, pois os apontadores são implícitos nessa estrutura
- Não há necessidade de se guardar memória na estrutura para armazenar um ponteiro para a próxima posição da lista
- Desvantagens:
  - custo para inserir ou retirar itens da lista – pode causar um deslocamento de todos os itens, no pior caso
  - em aplicações em que não existe previsão sobre o crescimento da lista, a utilização de arranjos pode exigir a realocação de memória
  - operação de alto custo em termos de tempo e memória, pois é preciso alocar uma nova área com mais posições do que a atual e copiar todos os itens para ela

Estrutura Auto-Referenciada

- Também chamada de lista encadeada ou lista ligada
- Cada item da lista contém a informação que é necessária para alcançar o próximo item
- Esta implementação permite utilizar posições não contíguas de memória
- É possível inserir e retirar elementos sem haver necessidade de deslocar os itens seguintes da lista

Estrutura Auto-Referenciada

Representação



- a estrutura de dados lista usando estruturas auto-referenciadas são constituídas de células
- a classe Lista contém uma referência para a célula da cabeça e uma referência para armazenar a posição corrente na lista

Estrutura Auto-Referenciada

- Para inserir um item no fim da lista
  - cria-se uma nova célula para o item
  - faz a lista apontar para a nova célula como sendo o fim da lista
  - faz essa nova célula apontar para None
- Para inserir um item em uma posição arbitrária i
  - cria-se uma nova célula
  - faz a célula x_{i-1} apontar para a nova célula
  - faz a nova célula apontar para a célula x_i

Estrutura Auto-Referenciada

- Implementação

Estrutura Auto-Referenciada

Considerações sobre estruturas auto-referenciadas

- Esse tipo de implementação permite inserir ou retirar itens do meio da lista a um custo baixo, aspecto importante quando a lista deve ser mantida em ordem
- Em aplicações em que não existe previsão sobre o crescimento da lista, é conveniente usar listas encadeadas, dado que o tamanho máximo da lista não precisa ser definido a priori
- A maior desvantagem desse tipo de implementação é a utilização de memória extra para armazenar as referências (os ponteiros prox)

Exercício

Questão 1

- Implemente uma função que recebe um vetor de valores inteiros com n elementos e construa uma lista encadeada armazenando os elementos do vetor nos nós da lista
- Se for recebido por exemplo o vetor {3,8,1,7,2}, a função deve retornar uma nova lista cujo primeiro nó tem a informação 3, o segundo nó tem a informação 8, e assim por diante.
- Se o vetor tiver zero elementos, a função deve ter como valor de retorno uma lista vazia

Exercício

Questão 2

- Implemente uma função que recebe duas listas lineares (implementadas com arranjo) e retorna a junção dessas duas listas
- A operação de junção deve entrelaçar os elementos das duas listas da seguinte forma:
  - seja L1 = {x_{11}, x_{12}, x_{13}, x_{14}} e L2 = {x_{21}, x_{22}, x_{23}, x_{24}}
  - a junção resultante seria l 3 = {x_{11}, x_{21}, x_{12}, x_{22}, x_{13}, x_{23}, x_{14}, x_{24}}
- Sejam as listas l 1 = {1, 3, 5, 7} e l 2 = {2, 4, 6, 8}, o resultado da operação de junção será l 3 = {1, 2, 3, 4, 5, 6, 7, 8}