

Estruturas de Dados

José Ricardo Mello Viana

Aula 06
Listas Encadeadas

2010.2

1 Introdução

- Encadeamento
- Exemplo

- Listas lineares: grande esforço computacional para inserção e remoção
 - Pode ser determinante para o baixo desempenho
 - Caso sejam frequentemente usadas
- Temos que saber o comprimento máximo da lista a priori

- Encadeamento
 - Permitir o crescimento
 - Diminuir o esforço computacional
 - Nós ligados através de referências
 - Contiguidade *lógica*

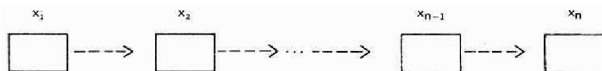


Figure: Esquema de encadeamento

- Mecanismo: referência (alocação dinâmica)
- Cada nó terá 2 partes (info e próximo)

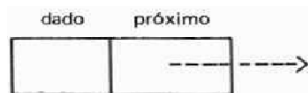


Figure: Estrutura de um nó

- tipo no::reg(dado: info, proximo: ref no)
- var primeiro: ref no //primeiro nó da lista
- primeiro \leftarrow nil //inicialização (lista vazia)
- var p: ref no // novo nó a ser colocado na lista
- aloque(p) // alocação da parte de valor de p
- p↑.dado // acesso ao dado
- p↑.proximo // acesso ao proximo

```
tipo no::reg (dado:info; próximo:ref no)
var lista, p: ref no; valor:info;
início
  lista ← nil
  leia(valor)
  aloque(p)
  p↑.proximo ← lista
  lista ← p
  p↑.dado ← valor
  ...
fim
```

- Passo 1: Estrutura do Nó

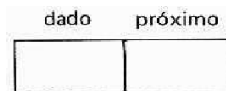


Figure: Estrutura do nó

- Passo 2: Lista vazia



Figure: Lista vazia

- Passo 3: Alocação de um nó

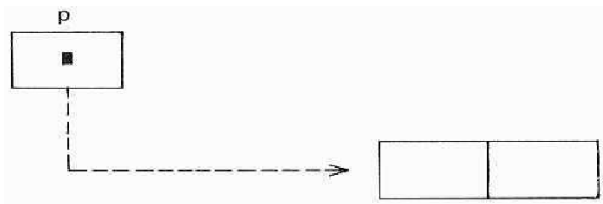


Figure: Alocação de um nó

- Passo 4: Atribuição de lista ao próximo do nó

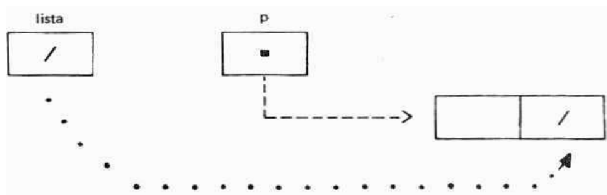


Figure: Lista estará depois desse nó

- Passo 5: p é atribuído à lista

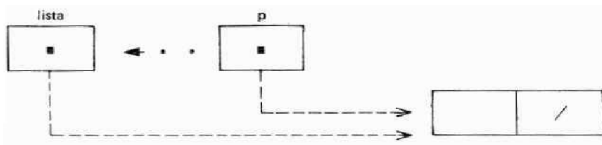


Figure: p estará no início da lista

- Passo 6: Atribuição do conteúdo lido à parte de dados de p

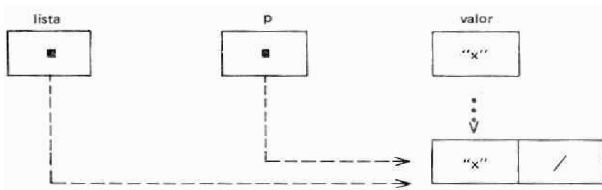


Figure: Conteúdo é atribuído a p

- Passo 7: Resultado final da inserção

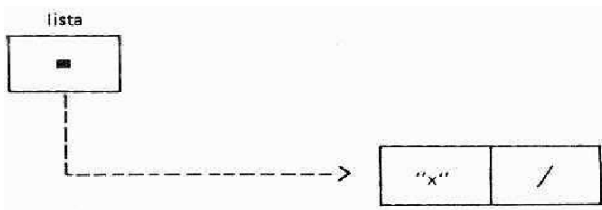


Figure: Resultado final da inserção