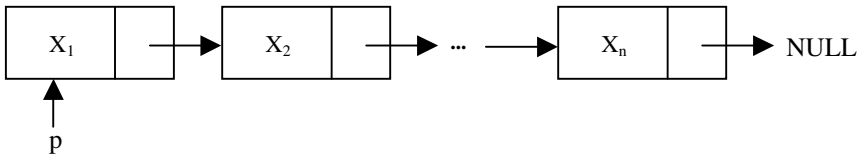
ESTRUTURAS DE DADOS LINEARES

Listas ligadas

Outra forma de implementar uma lista no computador é através da alocação dinâmica de memória. Neste caso, os elementos da lista são encadeados com o uso de ponteiros (ou apontadores), sendo possível preservar a ordem de seqüência dos elementos sem que eles ocupem posições contíguas de memória. A figura seguinte ilustra essa forma de implementação.



A lista é agora composta de nós ou células, sendo que cada um deles contém, além do item da lista, um campo do tipo ponteiro para armazenar o endereço de memória do nó seguinte. Já a alocação da memória necessária para o armazenamento de cada nó é realizada somente no momento de sua inserção, de forma dinâmica durante a execução do programa. Em contraste com a alocação estática, o programador não precisa saber previamente o tamanho máximo que a lista pode alcançar. Outra importante vantagem é o fato de ser possível inserir e retirar um elemento da lista sem que os demais tenham que ser deslocados.

Funções utilizadas (biblioteca stdlib.h)

- Alocar memória: void *malloc(unsigned número_de_bytes);
  Alocará uma área de memória contígua de tamanho igual a *número_de_bytes* e retornará um ponteiro para o primeiro byte dessa área alocada (byte do endereço mais baixo).
  Obs.: sempre deve verificar se o ponteiro retornado por malloc é diferente de NULL antes de utilizá-lo para armazenar algo.
- Desalocar memória: void free(void *);
  Obs.: deve sempre passar para a função *free* um ponteiro que foi utilizado anteriormente para alocar memória via a função *malloc()*.

Também estudar com atenção as seções 10.2 e 10.3 (págs. 166-173) do livro Cormen et al. **Algoritmos: Teoria e Prática**. 2ª edição. 2002. resolver todos os exercícios dessas duas seções

Exercício

Faça um programa que permita ao usuário construir uma lista ligada capaz de armazenar uma quantidade de números inteiros não conhecida em tempo de compilação. Esse programa deve dar as seguintes opções ao usuário:

1. Criar uma lista
2. Inserir um novo elemento na lista
3. Buscar por um elemento na lista
4. Imprimir todos os elementos da lista
5. Ordenar crescentemente e decrescentemente a lista
6. Excluir um elemento da lista
7. Inserir um elemento depois da i-ésima posição da lista
8. Inserir um elemento antes da i-ésima posição da lista
9. Excluir a lista