

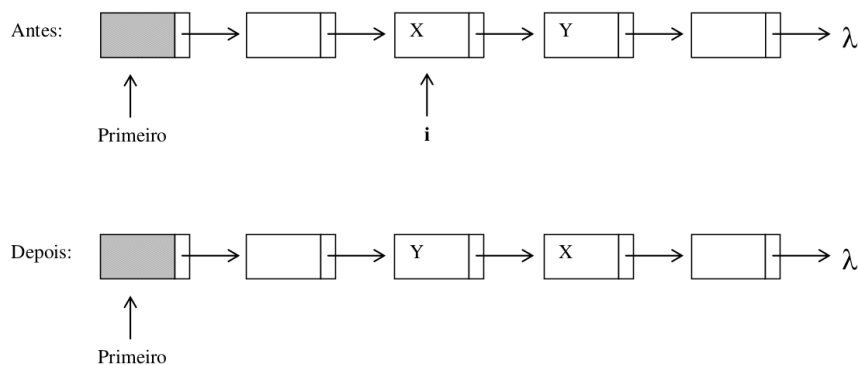
Lista 2

BCC202 - Estrutura de Dados I

2020.2

A presente lista de exercícios não precisa ser entregue.

[1] Escreva um procedimento `Troca(ListaEncadeada* pLista, int i)` troque os elementos de posição i (x na figura) e $i + 1$ (y na figura) na implementação de lista encadeada com cabeça vista em aula. (Obs. Não vale trocar apenas o campo item! Você deverá fazer a manipulação dos apontadores para trocar as duas células de posição). Não esqueça de tratar os casos especiais.



[2] Desenvolva um método para manter duas pilhas dentro de um único vetor linear (um arranjo) de modo que nenhuma das pilhas incorra em estouro até que toda a memória seja usada, e toda uma pilha nunca seja deslocada para outro local dentro do vetor.

[3] Implemente uma fila em C, onde cada item da fila consista em um número variável de inteiros.

[4] (Shaffer) Um problema muito comum em compiladores e editores de texto é determinar se os parênteses em uma string estão balanceados e corretamente aninhados. Por exemplo, “((()))()” é uma sequência válida, enquanto “)()”(e “())” não são.

- Apresente um algoritmo que retorna verdadeiro se a string composta somente de parênteses é válida e falso, caso contrário. Dica: use uma pilha e percorra a string da esquerda para direita – nunca você poderá achar mais parênteses à direita do que à esquerda e, ao final, o número deverá ser igual.
- Estenda seu algoritmo para indicar a posição do primeiro parênteses incorreto.

[5] Você foi contratado pela Google para refazer o Google++. A sua primeira tarefa é implementar o TAD AmigosGoogle para representar as relações de amizade. Ou seja, para cada pessoa cadastrada você tem que armazenar a sua lista de amigos. Considere que cada pessoa tem um identificador único que é um número inteiro e as operações básicas são:

```
1 | void relaciona(int Pessoa1, int Pessoa2, AmigosGoogle* pAmigos)
2 |
3 | int[] amigos(int Pessoa, AmigosGoogle* pAmigos)
```

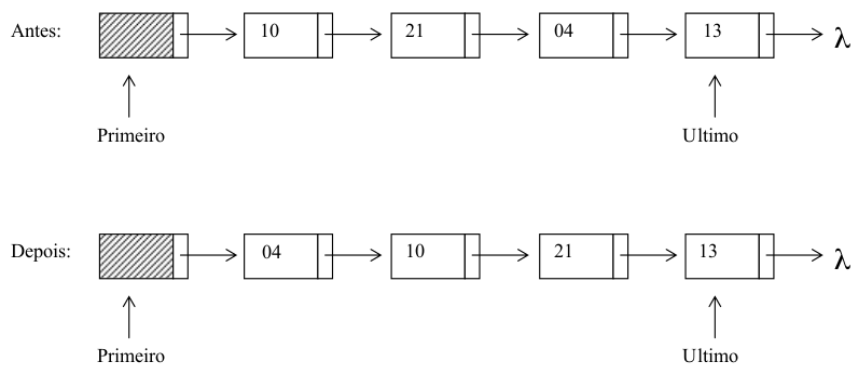
Explique qual estrutura de dados você utilizaria para implementar o TAD e apresente implementação completa para as operações.

[6] Apresente uma permutação de valores entre 0 e 7 que faça com que o Quicksort apresente o pior caso, explicando qual escolha do pivô.

[7] Quais das operações seguintes seria melhor realizada se primeiro fosse feita a ordenação dos números, ou seja, o custo da ordenação mais o da operação seria melhor do que o da operação sozinha. Em cada caso apresente o algoritmo e sua complexidade assintótica:

- a) Achar o menor valor.
- b) Achar o maior valor.
- c) Achar a média aritmética.
- d) Achar a mediana.
- e) Achar a moda.

[8] Escreva uma função `void MoveMenor(TipoLista Lista)` que, dada uma lista com um número qualquer de elementos, acha o menor elemento da lista e o move para o começo da lista, como exemplificado na figura abaixo. (Obs. Não vale trocar apenas os campos item ou usar uma lista / fila / pilha auxiliar! Você deverá fazer a manipulação dos apontadores para trocar as células de posição). Qual é a ordem de complexidade da sua função. Explique.



[9] Considere uma pilha P vazia e uma fila F não vazia. Utilizando apenas os testes de fila e pilha vazias, as operações Enfileira, Desenfileira, Empilha, Desempilha, e uma variável aux do TipoItem, escreva uma função que inverta a ordem dos elementos da fila.

[10] Implemente uma fila de inteiros em C, usando uma implementação por arranjos (um vetor `fila[100]`), onde `fila[0]` e `fila[1]` são usados para representar a posição inicial e final da fila respectivamente e `fila[2]` a `fila[99]` são usados para conter os elementos do vetor. Demonstre como inicializar esse vetor de modo a representar a fila vazia e escreva funções Desenfileira, Enfileira e Vazia para tal implementação.

[11] A ordenação por inserção bidirecional é uma modificação da ordenação por inserção simples da seguinte forma: um vetor de saída separado, de tamanho n, é reservado. Esse vetor de saída atua como uma estrutura circular. `x[0]` é posicionado no elemento do meio do vetor. Assim que um grupo contíguo de elementos estiver no vetor, será aberto espaço para um novo elemento deslocando todos os elementos menores um passo para a esquerda ou todos os elementos maiores um passo a direita. A escolha do deslocamento é feita para provocar o menor número de movimentações. Escreva uma função para implementar essa técnica.