01. Lista Sequencial
02. Lista Sequencial Ordenada
03. Lista Ligada – implementação estática [inserção ordenada]
04. Lista Ligada – implementação dinâmica [inserção ordenada]
05. Lista Circular Ligada Dinâmica – Nó Cabeça
06. Pilha – implementação estática
07. Pilha – implementação dinâmica
08. Deque – implementação dinâmica
09. Fila – implementação estática
10. Fila – implementação dinâmica
11. Duas pilhas – implementação estática

# 1. Estruturas

## 1.1 Lista

É uma lista.

## 1.2 Pilha

Uma pilha é uma estrutura de dados do tipo LIFO (Last In, First Out), ou seja, quando o último elemento é sempre o primeiro a sair. Uma estrutura desse tipo exige apenas um indicador para o último elemento, pois tanto a inserção, quanto a exclusão, necessitam apenas dessa variável. Um exemplo de uso desse tipo de estrutura é a chamada “pilha de execução” usada para a execução de chamadas recursivas.

## 1.3 Deque

Um deque é uma fila duplamente ligada que pode ter inserção, ou exclusão, tanto no começo quanto no final.

## 1.4 Fila

Uma fila é uma estrutura do tipo FIFO (First In, First Out), isto é, o primeiro elemento a ser inserido é sempre o primeiro a ser excluído. Filas não são legais.

# 2. Implementações Estáticas e Dinâmicas

## 2.1 Implementações Estáticas:

**Vantagens:**

- No geral, não é necessário liberar a memória usada por um elemento quando ele é excluído;
- Em muitos casos a busca binária pode ser usada.

**Desvantagens:**

- Tem um limite máximo, e caso esse limite já tenha sido atingido é necessária a criação de uma outra estrutura, que terá o mesmo tamanho. No caso em que apenas um elemento ficou de fora é necessário criar uma outra estrutura, que terá o mesmo tamanho, para guardar esse único elemento;

- Para inserir (ou excluir) um elemento do meio da lista é necessário mover todos os outros elementos.

### 2.2 Implementações Dinâmicas:

**Vantagens:**

- Aloca memória de acordo com a demanda;
- Inserir (ou excluir) um elemento no meio da lista não é uma tarefa difícil, pois basta acertar os ponteiros.

**Desvantagens:**

- Toda vez que um elemento é excluído é necessário liberar a memória ocupada por ele (a não ser que seja especificado que não);
- Não é viável usar a busca binária.

## 3. Estruturas

### 3.1 Lista Sequencial

**Vantagens:**

- Permite busca sentinela.

**Desvantagens:**

- Se um registro do meio da lista for excluído, é necessário mover todos os outros para manter uma sequência sem “buracos”.

**Descrição:**

Nesta estrutura os registros são colocados em um vetor, ou seja, são colocados em sequência na memória (do ponto de vista do SO).

### 3.2 Lista Sequencial Ordenada

**Vantagens:**

- Permite busca binária.

**Desvantagens:**

- É necessário mover vários elementos durante a inserção (ou exclusão).

**Descrição:**

Trata-se de uma série de registros colocados em sequência e em ordem dentro de um vetor.

## 3.3 Lista Ligada Estática

### Vantagens:

- Os custos para a inserção, ou exclusão, de um elemento são relativamente baixos, pois não é necessário mover os elementos da lista pelo vetor, basta acertar os ponteiros.

### Desvantagens:

- Consome mais memória devido ao campo para armazenar o índice do próximo elemento;
- Para realizar uma busca sentinela sem ter que percorrer toda a lista para achar o elemento da última posição é necessário ter um campo armazenando o índice do último elemento da lista.

### Descrição:

Em uma lista ligada estática cada elemento armazena o índice do próximo elemento. Armazenar o índice do próximo elemento é um sacrifício que vale à pena quando se pensa em uma inserção, ou exclusão, no meio da lista já que para fazê-lo não é necessário mover alguns elementos da lista através do vetor, basta acertar os ponteiros. Nessa estrutura também há uma segunda lista que é a lista de elementos livres para serem usados.

## 3.4 Lista Ligada Dinâmica

### Vantagens:

- A alocação de memória para cada elemento da lista ocorre de maneira dinâmica, evitando desperdício de memória;
- A inserção, ou exclusão, no meio da lista tem baixo custo pois para fazer isso basta acertar os ponteiros.

### Desvantagens:

- Não permite busca binária, pois os elementos não estão armazenados em sequência na memória.
- Para esvaziar a lista é necessário percorrer toda a lista e liberar a memória alocada para cada elemento de forma individual.

### Descrição:

Cada elemento de uma lista ligada dinâmica tem um campo do tipo ponteiro armazenando o endereço do próximo elemento da lista.

## 3.5 Lista Circular Dinâmica – Nó Cabeça

### Vantagens:

- Só usa a quantidade necessária de memória, pois é dinâmica;
- Permite uma busca sentinela eficiente devido ao fato de o nó cabeça poder ser usado como sentinela, ou seja, não é necessário percorrer a lista até o final.

### Desvantagens:

- Consome um pouco mais de memória por causa do nó cabeça, que tem o mesmo tamanho que os demais nós;
- Para achar um elemento é necessário percorrer todos os anteriores mesmo que a lista esteja ordenada, pois os elementos não estão dispostos em sequência na memória.

### Descrição:

A ideia de uma lista circular dinâmica com nó cabeça é ter uma lista com um elemento fixo que não é visível para o usuário e que não pode ser excluído. O nó cabeça é do mesmo tipo que os demais nós (possui os mesmos campos), porém os ponteiros são os mais importantes pois ele aponta para o primeiro elemento da lista, já que a estrutura “lista” em si só precisa ter um campo: ponteiro para o nó. Outra vantagem de usar o nó cabeça em uma lista circular é que não é necessário percorrer toda a lista para colocar um elemento sentinela no final da lista, pois usando um único campo somos capazes de saber onde começa e onde termina a lista.

## 3.6 Pilha Estática

### Vantagens:

- Uma pilha estática pode usar menos memória do que uma dinâmica já que não é necessário armazenar o endereço para o próximo elemento;
- Não é necessário se preocupar com a alocação de memória durante a inserção, ou com a liberação de memória durante a exclusão, porque a memória usada para o armazenamento já está alocada estaticamente.

### Desvantagens:

- Como a memória já está alocada estaticamente, pode acontecer uma situação em que toda a memória alocada para estrutura não seja utilizada, ou seja, um desperdício.

### Descrição:

Em uma pilha estática toda memória que será usada já está alocada dentro da própria estrutura. O único campo adicional necessário nesse tipo de estrutura é um campo indicando a posição do topo da pilha.

## 3.7 Pilha Dinâmica

### Vantagens:

- A memória locada é proporcional à quantidade de elementos da pilha, ou seja, não há memória alocada sem uso.

### Desvantagens

- Pode precisar de mais memória do que uma pilha estática em uma situação em que ambas armazenam uma quantidade n de elementos, onde n é a quantidade máxima da lista estática. Isso acontece devido ao fato de que cada elemento da lista dinâmica precisa de um ponteiro para o próximo elemento.

### Descrição:

Na pilha dinâmica todos os elementos não estão armazenados em um pedaço de memória contínuo (do ponto de vista do sistema operacional), portanto cada elemento deve armazenar o endereço para o próximo elemento. Nesse caso, possuir apenas o campo “topo” já é suficiente para o funcionamento da estrutura.

## 3.8 Deque Dinâmico com nó cabeça

### Vantagens:

- Realiza uma busca sentinela com facilidade devido ao fato de possuir nó cabeça;
- O elemento final pode ser acessado sem ter que percorrer toda a lista ou alocando uma variável extra na estrutura em si para armazenar o último elemento. Como consequência disso, acessar um elemento que está mais próximo do final se torna mais eficiente e possui um custo menor.

### Desvantagens:

- Usa um pouco mais de memória devido ao nó cabeça;
- Todo elemento precisa de dois campos do tipo ponteiro para elemento, o que consome mais memória do que uma implementação estática.

### Descrição:

Em um deque dinâmico com nó cabeça o único campo necessário na estrutura é um campo que aponta para o nó cabeça. Por ser duplamente ligado e ser dinâmico a inserção (ou exclusão) tanto no final, quanto no começo, são fáceis.

## 3.9 Fila Estática

### Vantagens:

- Os elementos já estão em um vetor, logo para encontrar o próximo elemento basta percorrer o vetor, o que é menos custoso do que percorrer a memória como um todo.

### Desvantagens:

- Pode haver desperdício de memória devido ao fato da memória ser alocada estaticamente.

### Descrição:

Como a fila é uma estrutura do tipo FIFO, um novo elemento deve ser inserido sempre no final da fila, que nem sempre é o final do vetor. A posição do espaço vago no final da fila é dada pela fórmula **(número de elementos + início da fila) % tamanho do vetor,** justamente por isso que a estrutura em si armazena o índice do início da fila e a quantidade de elementos na fila.

## 3.10 – Fila Dinâmica

### Vantagens:

- Armazena elementos de forma dinâmica, podendo gastar menos memória do que uma fila estática.

### Desvantagens:

- Em certos casos gasta mais memória do que uma lista estática porque todos os elementos possuem, além do campo registo, ao menos mais um campo, que é um ponteiro para o próximo elemento.

### Descrição:

Uma fila dinâmica armazena o endereço do elemento do início e o endereço do elemento do final da fila, pois quando um elemento é inserido, ele é inserido no final, e quando a exclusão é solicitada, o elemento do começo é excluído. Assim, não é necessário ficar percorrendo a lista para inserir/excluir elementos.

### 3.11 Duas Pilhas Estáticas

#### Vantagens:

- Para excluir um elemento da pilha basta decrementar o topo.
- Caso haja um número fixo de elementos no qual cada elemento retirado de uma pilha é inserido direto em outra pilha, essa estrutura se mostra eficiente pois evita que duas pilhas sejam criadas.

#### Desvantagens:

- Há um campo a mais para armazenar o topo da segunda pilha.

#### Descrição:

Em uma pilha dupla estática um vetor unidimensional é dividido por duas pilhas, uma que cresce a partir do início do vetor e vai em direção ao final do vetor, e outra que faz o contrário, cresce partindo do final do vetor em direção ao começo do vetor. Os únicos campos necessários para controle nessa estrutura são dois campos no qual cada um aponta para o topo de cada pilha. É possível que uma pilha ocupe todo o espaço do vetor enquanto outra está vazia.

## 4. Precauções e Dicas

- Verificar se todas as variáveis foram inicializar em algum momento;
- Caso haja a função main, verificar se o “return 0” foi colocado;
- Verificar se os tipos de dados estão compatíveis e se a forma de os acessar está correta (“->” para ponteiros e “.” para o original);
- Se for retornar o tamanho em bytes de uma estrutura com o nó cabeça, não esquecer de incluir o nó cabeça na conta;
- Lembrar sempre de verificar se o que foi pedido foi realmente feito (às vezes uma coisa é pedida, mas para fazer tal coisa é necessário realizar tantos passos que o objetivo final acaba sendo esquecido);
- Para realizar uma busca binário toda a estrutura deve estar ordenada.