**PILHA:**

- pode ser implementada por meio de uma lista encadeada ou de um vetor

- o que difere uma pilha de uma fila é o mecanismo responsável pelo armazenamento e recuperação de seus elementos

- mecanismo da pilha : **LIFO(**Last In First Out**) último que entra primeiro que sai**

- **Funções:** push() – coloca um item no topo da pilha. Pop() – retira o item do topo da pilha. Peek – mostra o elemento que esta no topo da pilha

- **OPERAÇÕES BÁSICAS:**

* Criar uma pilha vazia
* Inserir um elemento (push)
* Retirar um elemento (pop)
* Verificar se a pilha esta vazia
* Liberar a estrutura pilha
* Mostrar a pilha (\*)

**FILA:**

- mecanismo da pilha: **FIFO(**First in First Out**)primeiro que entra primeiro que sai**

- pode ser implementada por meio de um vetor ou de uma lista encadeada

- **OPERAÇÕES BÁSICAS:**

* Criar uma fila vazia
* Inserir um elemento no fim
* Retirar um elemento do inicio
* Verificar se a fila esta vazia
* Liberar a estrutura fila
* Mostrar a fila (\*)

- a fila é manipulada pelas suas duas extremidades : inicio e fim

- **inserções sempre no final, remoções sempre do inicio da fila**

- implementações: estática(usando vetor). Dinâmica(usando uma lista encadeada)

- **Estatica (usando vetor) pode ser implementada:**

* Linear sem compactação (sem qualquer movimentação de dados)
* Linear com compactação burra(ou seja, com movimentação de dados a cada retirada)
* Linear com compactação inteligente(ou seja, com movimentação de dados apenas quando estritamente necessário)
* Forma circular (sem movimentação dos dados)

Fila estática sobre um vetor:

* Sem compactação: Inserções incrementam o FIM, e remoções incrementam o INICIO (INICIO e FIM variando)
* Compactando a fila – movimentação de dados: Cada vez que um elemento é removido move-se toda a fila em direção ao seu inicio de forma a preencher o espaço deixado pela remoção. Portanto: As inserções incrementam o FIM mas o INICIO fica fixo na posição (zero) do vetor
* Com reaproveitamento de espaços – compactação inteligente: Ao invés da compactação ocorrer a cada remoção, ela será feita apenas ao ser detectado um falso sinal de “fila cheia”

**Considerações:** As alternativas que envolvem compactação de dados, são, evidentemente, melhores que a alternativa que deixa a fila se esgotar sem o reaproveitamento de espaços entretanto, estas alternativas causam overhead de processamento , pois a movimentação dos dados na fila pode ser uma operação bastante lenta! Se o vetor tiver N posições e a fila tiver N-1 elementos, serão necessárias N-2 movimentos, dos N-2 elementos desde o final da fila até o seu início.

**-Fila Circular:** na sua implementação considera o vetor como um arranjo circular, em que o seu final se liga ao seu inicio, sem interrupções. A fila circular pode se tornar bastante vantajosa ao viabilizar a reutilização das posições desocupadas sem a necessidade de movimentação de dados. **CONSEQUENCIAS:** **início == fim não implica mais em FILA VAZIA!! 1**

**Considerações:** A alternativa para se determinar se a fila está ou não vazia é incluir dentro do TDA\*( (\*)Tipo de Dados Abstrato) fila mais um campo: o tamFila que determinará efetivamente quantos são os elementos da fila!

**LISTAS ENCADEADAS:**

- são usadas para trocar coleções de dados por meio de uma estrutura (de dados) mais flexível do que um vetor

- as listas diferentes dos vetores podem crescer ou diminuir quando necessário, por isso são chamadas de estruturas dinâmicas

**OPERAÇÕES BÁSICAS**

* criar uma lista vazia
* inserir um elemento na lista (no inicio ou de forma ordenada)
* mostrar os elementos da lista
* verificar se a lista esta vazia
* buscar um elemento na lista
* retirar um elemento da lista
* liberar a lista

- uma lista é chamada encadeada quando um elemento aponta para o seu próximo

- a retirada de um elemento da lista é um pouco mais complexa pois implica no religamento dos ponteiros

**Listas simplesmente encadeadas:**

* caracterizam-se pelo simples encadeamento entre os seus elementos
* não permitem o percurso reverso da lista, a não ser pela recursividade
* para remover um elemento, devido ao simples encadeamento, não adianta nada conhecermos o ponteiro do elemento a ser retirado, pois, de qualquer modo, teremos que percorrer a lista a procura do seu antecessor

**LISTAS DUPLAMENTE ENCADEADAS:**

* cada elemento da lista tem um ponteiro tanto para o seu próximo como para o seu anterior
* assim, dado um elemento, podemos encontrar com facilidade os elementos que lhe são adjacentes
* se tivermos um ponteiro para o ultimo elemento da lista, poderemos facilmente percorre-la em ordem inversa
* OPERAÇÕES BASICAS: criar uma lista vazia, inserir um elemento na lista (no inicio ou de forma ordenada), mostrar os elementos da lista, verificar se a lista esta vazia, buscar um elemento na lista, retirar um elemento da lista, liberar a lista

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**O pior é o bubble sort, o melhor é o merge**

**Texto, Carta

Descrição gerada automaticamente**

**Texto, Carta

Descrição gerada automaticamente**

**Texto, Carta

Descrição gerada automaticamente**

**Texto, Aplicativo, Carta

Descrição gerada automaticamente**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

**Texto, Carta

Descrição gerada automaticamente**