

# **O que é um Programa?**

Um programa pode ser visto como a especificação formal da solução de um problema.

N. Wirth expressa em sua equação

programa = algoritmo + estruturas de dados

Como vimos em AED I, a qualidade da solução de um problema depende, entre outros fatores, da forma como estão organizados os dados relevantes.



7

# **O que é Estruturas de Dados?**

Estruturas de dados são formas genéricas de se estruturar dados/informação de modo a serem registradas e processadas pelo computador. Ex.:

 vetores;  matrizes;  listas (pilhas e filas);  árvores;  grafos, etc.

Contudo, estas só adquirem significado quando associadas a um conjunto de operações, que visam, de um modo geral, manipulá-las (algoritmos).

Visando recordá-los dos pontos principais e estabelecer um elo entre as disciplinas, faremos agora um overview dos tópicos vistos na disciplina Algoritmos e Estrutura de Dados I através um conjunto de exercícios.

**O que é uma lista?**

Uma lista é uma sequência de zero ou mais elementos, cada um deles sendo um valor primitivo (átomo ou nodo) ou composto. Em outras palavras, a estrutura de lista representa a ordem linear entre os elementos. Listas podem ser homogêneas ou 9heterogêneas, lineares ou não lineares.

# **Listas, Pilhas e Filas**

Cite algumas operações primitivas necessárias a uma implementação de uma lista:

-criar uma nova lista;

-verificar se a lista é vazia;

-acessar o k-ésimo elemento;

-inserir um elemento como k-ésimo da lista;

-remover o k-ésimo elemento;

-etc.

No que tange a gerência de memória como uma lista pode ser implementada?

Com relação a gerência de memória uma lista pode ser implementada de forma sequencial ou encadeada.

Utilizando a alocação encadeada uma lista pode ser implementada de algumas formas, cite-as:

-simplesmente encadeada;

- com nó de cabeçalho;

-circular;

-duplamente encadeada;

11-etc.

Disciplinas de acesso

Muitas vezes é útil impor, para manipulação de uma certa estrutura de dados, restrições quanto à visibilidade de seus componentes ou quanto à ordem que deve ser respeitada para se efetuarem operações, como inserções ou retiradas, por exemplo. Isto ajuda na modelagem de certos processos que ocorrem no mundo real.

Com o tempo e a prática, foram identificadas algumas disciplinas de acesso aplicadas a estruturas de dados, úteis em diversas aplicações. Dois casos dos mais importantes são casos particulares de listas com disciplinas de acesso, denominados: filas e pilhas.

**O que é uma fila** **?**

Uma fila é uma lista com restrições de acesso, em que as operações de inserção são realizadas sobre uma das extremidades, o fim da lista, enquanto operações de consulta e retirada são feitas na outra extremidade, o início da fila.

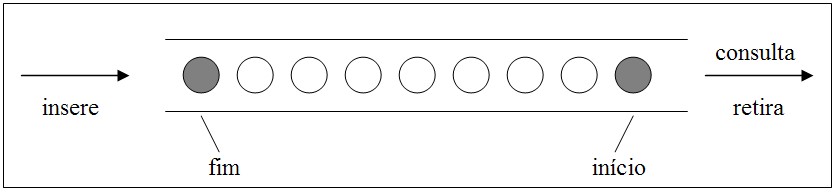
Isto leva ao critério FIFO (first in, first out) que indica que o primeiro item que entra é o primeiro a sair da estrutura. O modelo intuitivo para isto é o de uma fila para atendimento em um guichê, na qual são atendidas as pessoas pela ordem de chegada.

# **Listas, Pilhas e Filas**

O atendente só tem contato com (só pode consultar) o primeiro (ou o mais antigo) da fila. Novos pretendentes ao serviço entram no fim da fila.

No modelo formal, não há opção de abandono da fila: somente o primeiro pode ser retirado

(sair da fila).



Especifique uma forma simples e eficiente de armazenar em memória uma fila.

Podemos utilizar uma lista simplesmente encadeada com um nó de cabeçalho (descritor) contendo um ponteiro para o primeiro elemento e outro para o último elemento da fila.





Nodo n

Nodo 2

. . .

F



INICIO

Nodo 1

FIM

Exemplo, utilizando a linguagem C, da definição de um TAD FILA\_ENC (de valores inteiros). typedef struct nodo

{ int inf;

struct nodo \* next;

}NODO;

typedef struct

{

NODO \*INICIO;

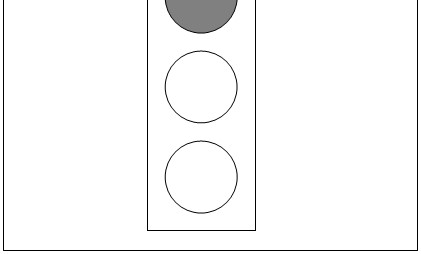
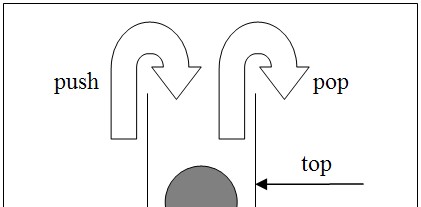
NODO \*FIM; }DESCRITOR; typedef DESCRITOR \* FILA\_ENC; void cria\_fila (FILA\_ENC \*); int eh\_vazia (FILA\_ENC); void ins (FILA\_ENC, int); int cons (FILA\_ENC); void ret (FILA\_ENC); int cons\_ret (FILA\_ENC);

Defina uma pilha.

Uma pilha é uma lista com restrições de acesso, onde todas as operações só podem ser aplicadas sobre uma das extremidades da lista, denominada topo da pilha.

Com isso estabelece-se o critério LIFO (last in, first out), que indica que o último item que entra é o primeiro a sair.

O modelo intuitivo para isto é o de uma pilha de pratos, ou livros, etc, na qual só se pode visualizar (consultar) o último empilhado e este é o único que pode ser retirado. E também qualquer novo empilhamento (inserção) se fará sobre o último da pilha.



Especifique uma forma simples e eficiente de armazenar em memória uma fila.

Uma pilha pode ser armazenada em uma lista simplesmente encadeada, sem perda de eficiência. P

. . . 

Nodo 1 Nodo 2 Nodo n

A implementação das operações é trivial.

Para fazer uma inserção, basta alocar um nodo para o novo valor, ligá-lo ao primeiro nodo da lista e fazer o ponteiro apontar para 19o novo nodo.



Uma retirada exige apenas que o ponteiro passe a apontar para o segundo nodo da lista (ou ser anulado, se houver apenas um nodo). Uma consulta exige apenas a recuperação do valor do primeiro nodo.

Desta forma, podemos definir o TAD PILHA\_ENC (de valores inteiros) da seguinte forma.



typedef struct nodo

{ int inf;

struct nodo \* next;

}NODO; typedef NODO \* PILHA\_ENC; void cria\_pilha (PILHA\_ENC \*); int eh\_vazia (PILHA\_ENC); void push (PILHA\_ENC \*, int); int top (PILHA\_ENC); void pop (PILHA\_ENC \*); int top\_pop (PILHA\_ENC \*)

Exercício: Faça um exemplo de cada usando a linguagem C++, exemplificando uma lista, uma Pilha e uma Fila.