

Fila de prioridade

Em geral, o primeiro elemento que entra na fila é o primeiro que sai.

Na fila de prioridade a prioridade de saída do elemento é determinada pelo valor de um (ou mais) atributos.

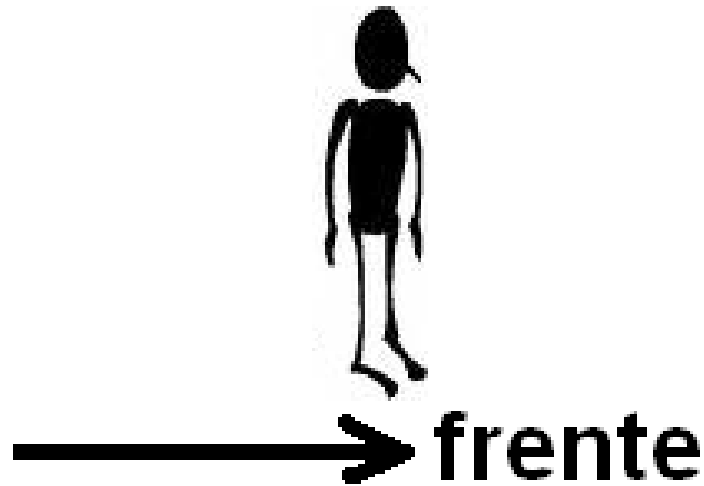
Dessa forma... o primeiro elemento que entra na fila de prioridade não necessariamente é o primeiro que sai.

Atributos são informações/dados associados a um indivíduo/objeto/elemento.

Em termos de implementação, os atributos são os campos de uma variável do tipo registro (*struct*).

Portanto, na fila de prioridade, os valores de um ou mais campos da *struct* que representa o elemento determinam uma posição de inserção não necessariamente no final da fila.

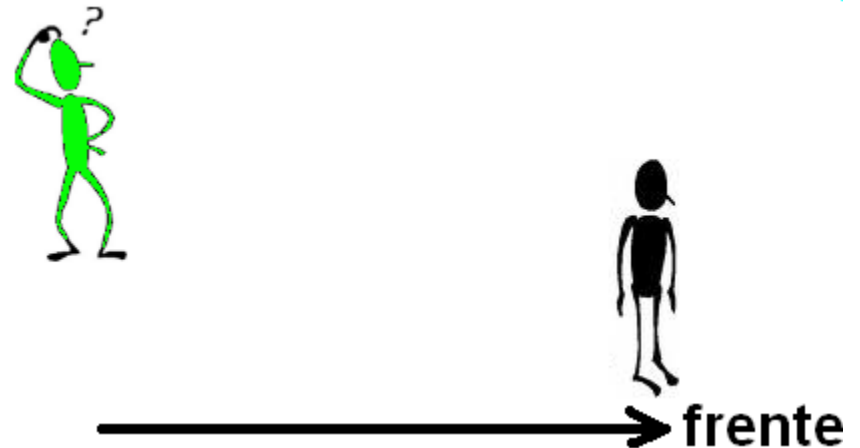
Fila de prioridade: a primeira inserção é trivial



Fila de prioridade: 2ª inserção

A inserção na FP não vazia, não necessariamente ocorre na *cauda*.

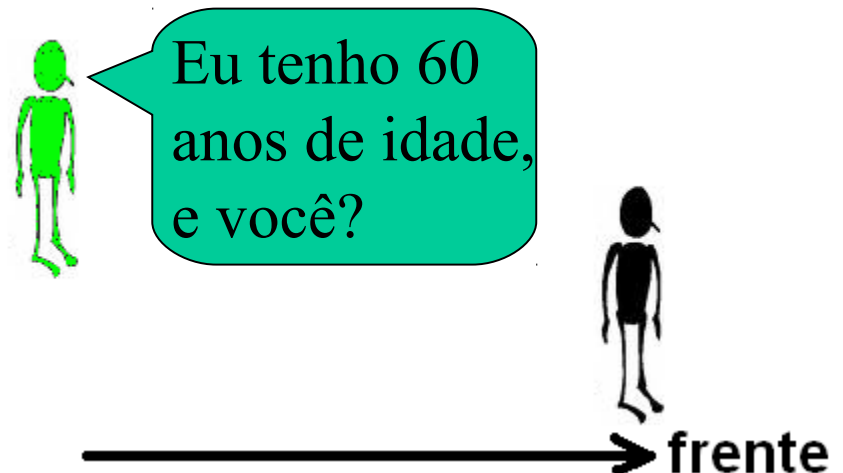
Como estabelecer o posicionamento do novo elemento na fila?



Fila de prioridade: 2ª inserção

Como estabelecer o posicionamento (prioridade de saída) do 2º elemento na fila de prioridade?

Através da comparação de valores individuais de um ou mais atributos, por exemplo: idade.



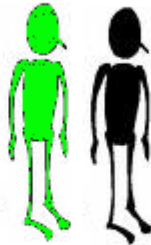
Fila de prioridade: 2ª inserção

O prioridade de saída pode obedecer o critério da idade.

A)

Eu tenho 65

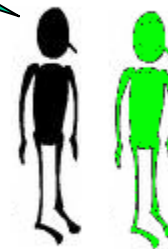
Eu tenho 60



B)

Eu tenho 15

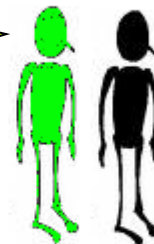
Eu tenho 60



C)

Eu tenho 60

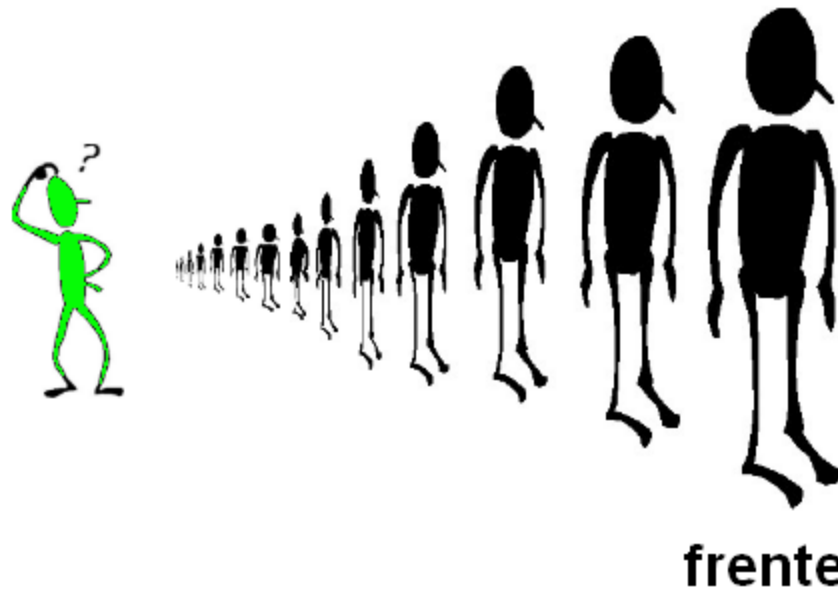
Eu **também** tenho 60, mas **cheguei primeiro!**



Fila de prioridade: n -ésima inserção

A fila já contém $n-1$ elementos.

Xiii, e agora?

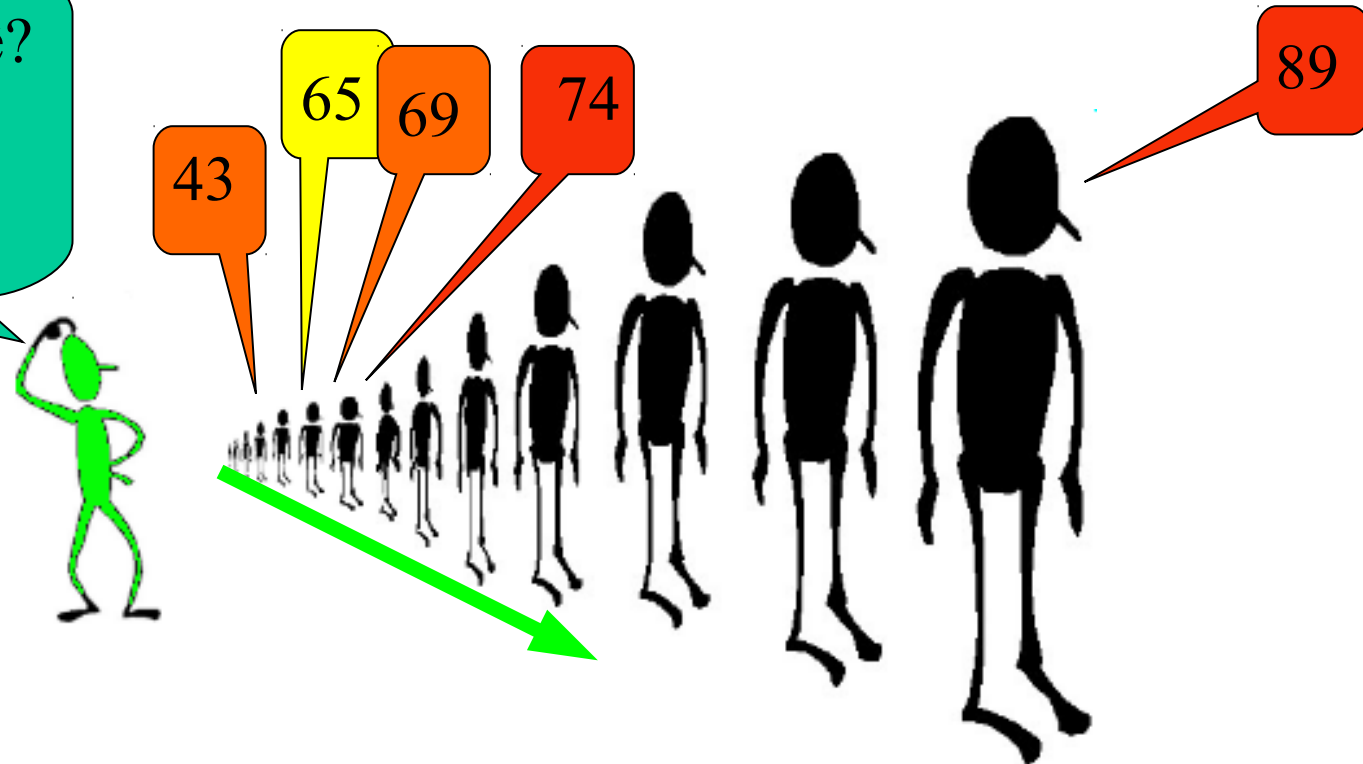


Fila de prioridade: n-ésima inserção

Exemplo: prioridade determinada pela idade:

Posso perguntar a cada um a **partir da cauda**.

Eu tenho 60, e você?
Você? Você?.....

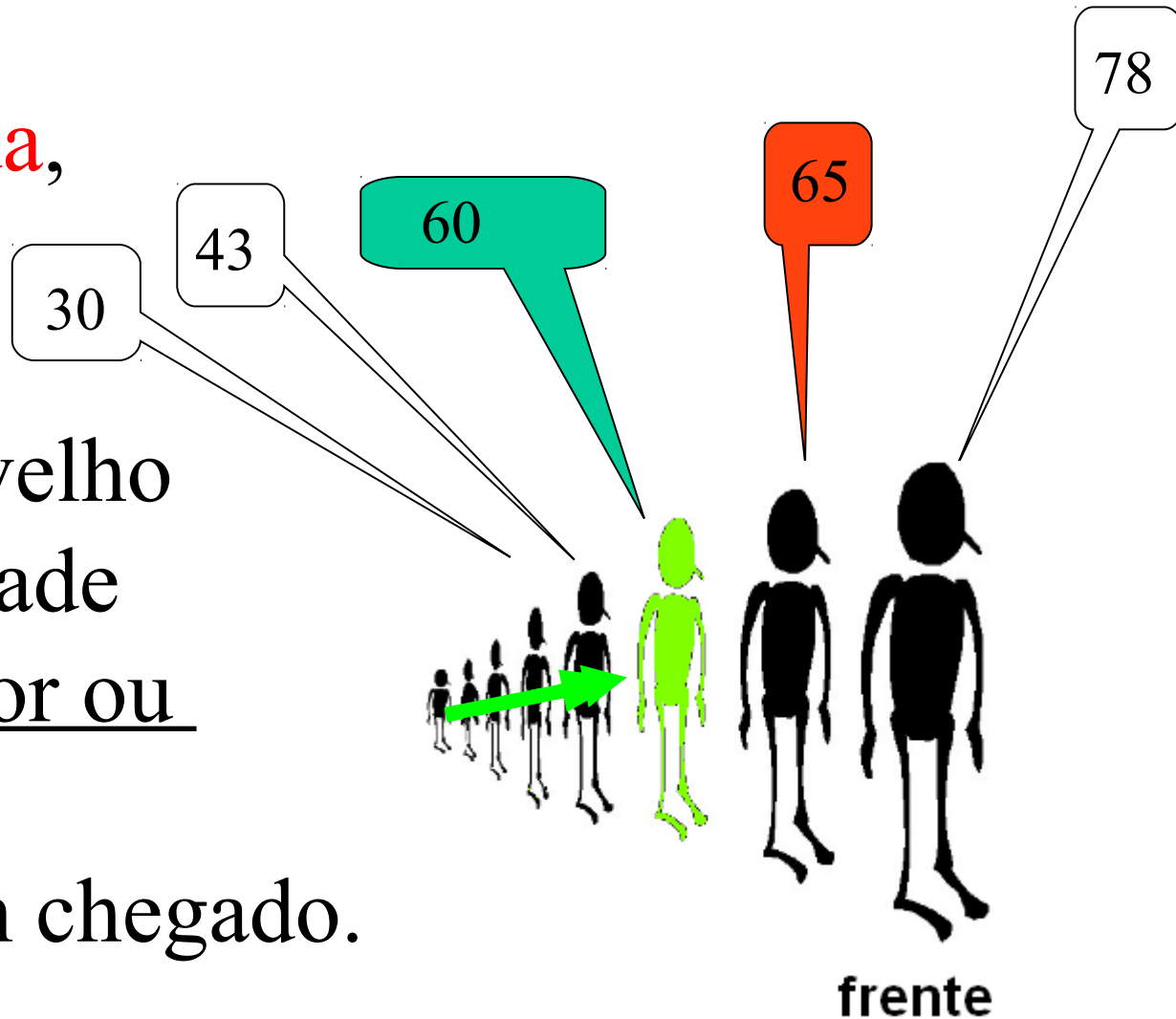


frente

Fila de prioridade: n-ésima inserção

Prioridade determinada pela idade.

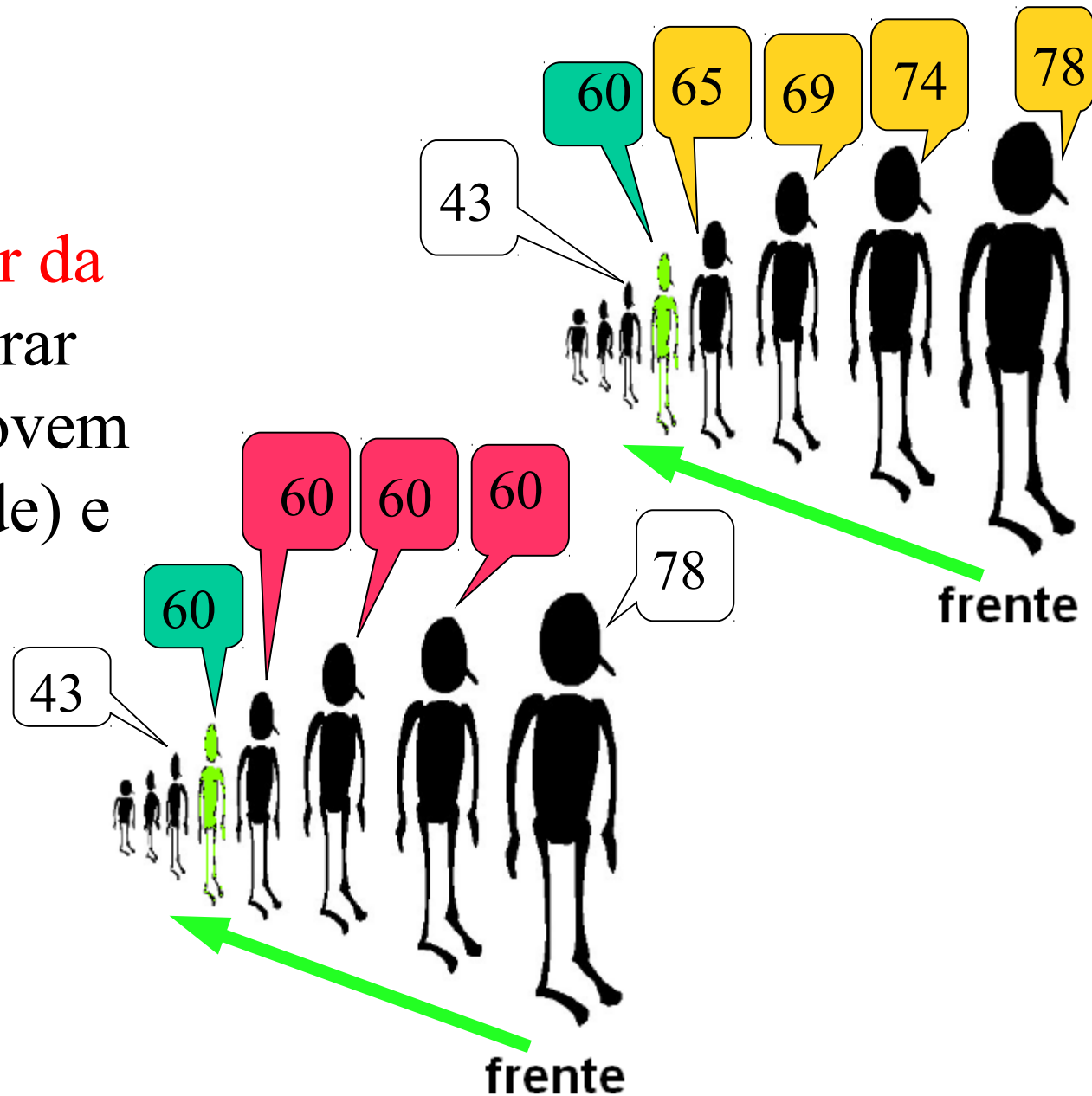
A **partir da cauda**,
inserir atrás do
primeiro
elemento mais velho
ou de mesma idade
(prioridade maior ou
igual) que a do
indivíduo recém chegado.



Fila de prioridade: n-ésima inserção

Alternativa:

Perguntar **a partir da frente** até encontrar elemento mais jovem (menor prioridade) e inserir à frente deste.



TDA-Fila de prioridade

ATENÇÃO: para realizar a inserção é necessário conhecer a relação entre os campos da *struct* que representa o elemento a ser enfileirado.

Ex: Pedro.idade < João.idade

Portanto, é necessário acessar campos encapsulados na aplicação da fila. Mas isso quebra o tão desejado encapsulamento!

Como implementar a inserção sem quebrar o encapsulamento do TDA?

Como implementar a inserção no TDA-Fila de prioridade sem quebrar o encapsulamento?

O TDA precisa conhecer apenas uma relação ($<$, $>$, $=$) entre a prioridade de cada elemento já inserido e a prioridade do novo elemento!

Uma função (Callback) construída no módulo cliente do TDA pode prover tal relação.

Esta função pode ser passada como um parâmetro para o TDA toda vez que a relação entre elementos for necessária.
Ex: para uma inserção na Fila de Prioridade.

TDA-Fila de prioridade

Inserção sem quebrar encapsulamento do TDA

Módulo Cliente

função de comparação:

*int compara(void *inf1, void *inf2)*

Módulo TDA

Protótipo da inserção:

*int insere(pFila p, void *novo, int (*compara)(void *inf1, void *inf2))*



TDA-Fila de prioridade

Inserção sem quebrar encapsulamento do TDA

A interface do TDA deve exibir nas pré/pós-condições a necessidade da função de callback determinando o tipo de parâmetros adotados, a ordem da lista de parâmetros e os valores de retorno em função dessa ordem:

/*

pré-condição: a função empilha em uma fila já criada...

o endereço de uma função de callback `int (*compara)(void *inf1, void *inf2)` deve ser fornecida para comparar o primeiro parâmetro (`inf1`) e o segundo (`inf2`). Esse callback retorna `'>'` no caso de `inf1 > inf2`, `'<'` se `inf1 < inf2` e `'='` caso contrário.

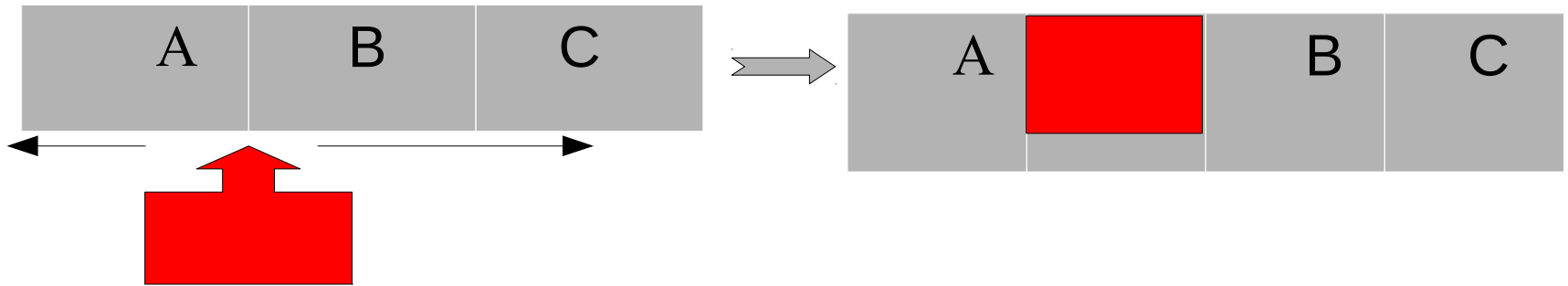
pós-condição: a função de inserção retorna `SUCESSO` no caso do enfileiramento por prioridade ter sido bem sucedido, caso contrário retornará `FRACASSO`...

*/

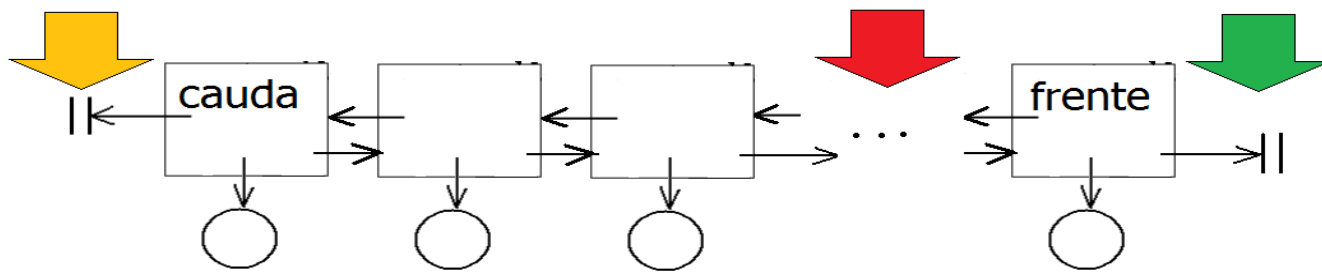
`int insere(pFila p, void *novo, int (*compara)(void *inf1, void *inf2))`

TDA-Fila de prioridade

É possível uma implementação estática (contida em um vetor), porém, exceto para as inserções nas extremidades, ela demanda muitas movimentações de dados para abrir espaço no vetor;



A implementação encadeada elimina essas movimentações. Por isso prefere-se a FDSE e, principalmente, a FDDE para implementar a fila de prioridade.



TDA-Fila de prioridade

Supondo uma Fila Dinâmica Duplamente Encadeada

Quanto maior a prioridade → insere mais à frente

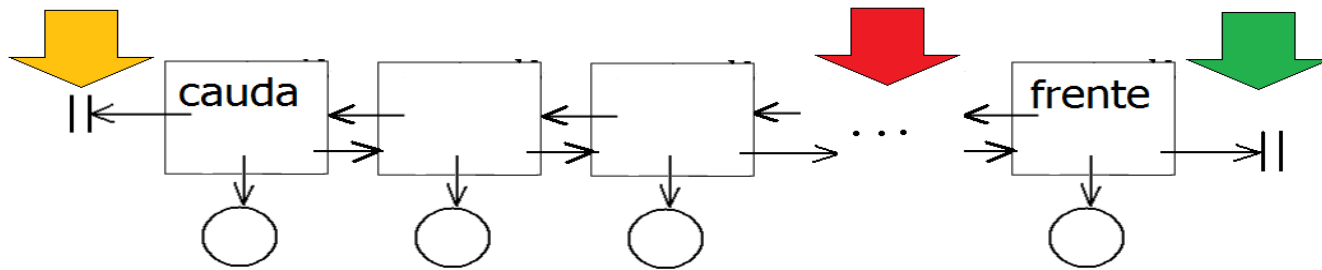
```
typedef struct noFDDE{ /*nó de dados*/  
    void * dados;  
    struct noFDDE *defrente;  
    struct noFDDE *atras;  
}NoFDDE, *pNoFDDE;
```

```
typedef struct FDDE{ /* Descritor */  
    pNoFDSE cauda;  
    pNoFDDE frente;  
    int tamInfo;  
}descFDDE;
```

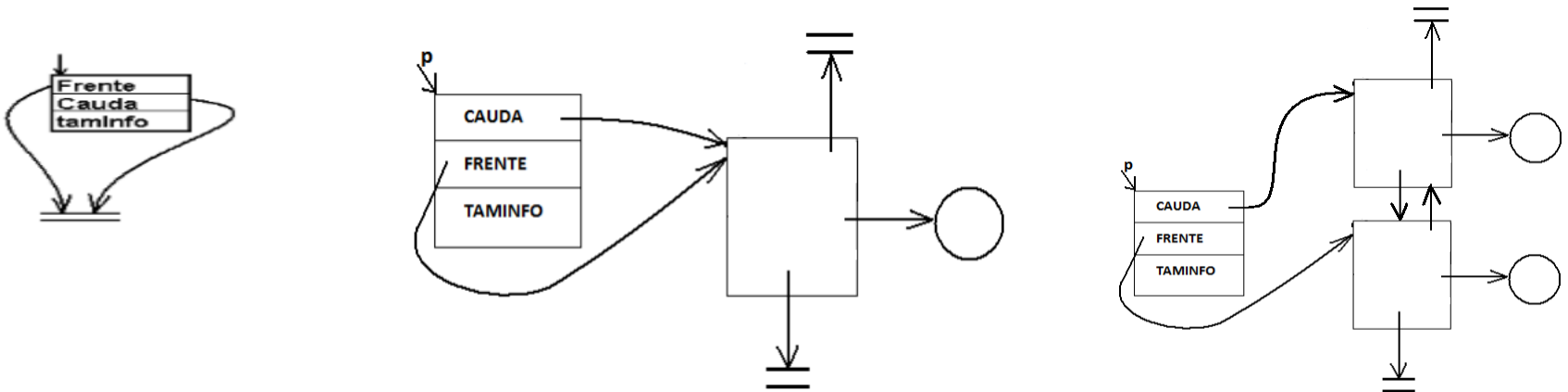
TDA-Fila (FDDE) de prioridade

Considerando: quanto maior a prioridade → insere mais à frente

A posição de inserção depende da prioridade e pode ocorrer em qualquer ponto da FDDE: na cauda (*), na frente (*) ou qualquer ponto intermediário entre elementos já inseridos (*):



Além disso, a operação de inserção deve tratar a FDDE vazia, contendo um único item ou vários elementos já inseridos:



Função de inserção V1

Cauda → Frente

int insere (pFila fp, void *novo, int (*compara)(void *inf1, void *inf2))

cria o novo nó da fila e região para dados;

executa a cópia dos novos dados para a região de dados recém criada;

SE (a fila está VAZIA)

insere o *novo* nó; /*o novo nó é o único na fila*/

SENAO

flag = 0;

visita a cauda da fila

FAÇA /*a partir da cauda*/

temp = (*compara)(itemVisitado, novo)

SE (temp == MAIOR OU temp == IGUAL)

insere *novo* item atrás do atual *itemVisitado*

flag = 1;

SENAO /* idade(itemVisitado) < idade(novo) → continua a busca*/

visita elemento imediatamente a frente do *itemVisitado*;

ENQUANTO(itemVisitado != null E flag == 0)

SE(flag == 0)

insere *novo* item na frente da fila. /* novo é elemento de maior prioridade na fila*/

(*compara)(itemVisitado, novo)

retorna :

'>' se idade(itemVisitado) > idade(novo)

'<' se idade(itemVisitado) < idade(novo)

'=' se idade(itemVisitado) == idade(novo)

Função de Inserção V2

Frente → Cauda

int insere (pFila fp, void *novo, int (*compara)(void *inf1, void *inf2))

cria nó de dados e região para dados;

executa a cópia dos novos dados para a região de dados recém criada;

SE (a fila está VAZIA)

insere o *novo* elemento; /*o novo nó é o único na fila*/

SENAO

flag = 0;

visita a frente da fila

FAÇA /* a partir da frente*/

temp = (*compara)(itemVisitado, novo)

SE (temp == MENOR)

insere *novo* item a frente do *itemVisitado*

flag =1;

SENAO /* idade(itemVisitado) ≥ idade(novo) → continua a busca */

visita o elemento imediatamente atrás do *itemVisitado*

ENQUANTO(*itemVisitado* != null E flag == 0)

SE(Flag == 0)

insere *novo* item no final da fila. /* novo é o elemento de menor prioridade na fila*/

(*compara)(itemVisitado, novo)

retorna :

'>' se idade(itemVisitado) > idade(novo)

'<' se idade(itemVisitado) < idade(novo)

'=' se idade(itemVisitado) == idade(novo)

Inserção TDA-Fila de prioridade

Cliente implementa a função de comparação

```
int compara(void *inf1, void *inf2)
{
    if( ((info *)inf1)->idade > ((info *)inf2)->idade)

        return MAIOR; /* 1 */
    else
        if ( ((info *)inf1)->idade < ((info *)inf2)->idade )

            return MENOR; /* -1 */
        else
            return IGUAL; /* 0 */
}
```

Protótipo



Cliente.h

```
#include "TDA_FPri.h"
...
```

Implementação



```
#include "Cliente.h"
...
```

Inserção TDA-Fila de prioridade

Cliente chamando à inserção

...

case '2':

```
printf("\nentre com o RG:\n");  
fflush(stdin);  
scanf("%i", &novolItem.chave);
```

```
if( insere(fila, &novolItem ,compara) == FRACASSO)
```

```
{...  
}
```

...

```
break;
```

...

Implemente uma fila de prioridade duplamente encadeada.

Implemente uma fila de prioridade em uma fila estática circular.

Quais as desvantagens da implementação da fila de prioridade em uma estratégia estática?