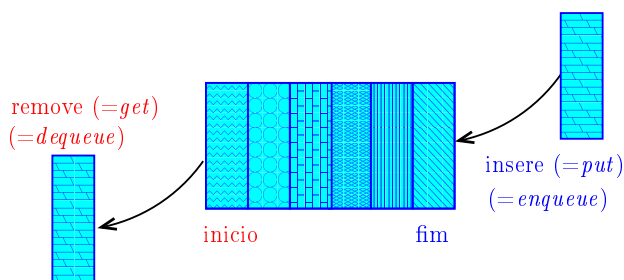


AULA 14

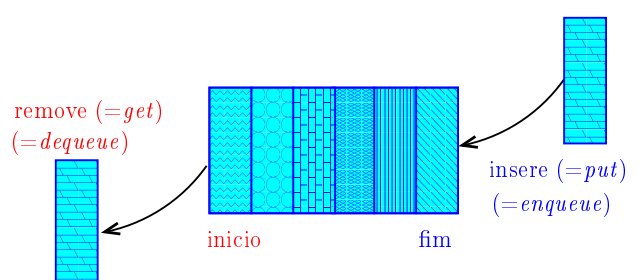
Filas

Uma **fila** (=queue) é uma lista dinâmica em que todas as **inserções** são feitas em uma extremidade chamada de **fim** e todas as **remoções** são feitas na outra extremidade chamada de **início**.



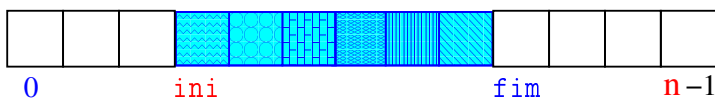
Filas

Assim, o **primeiro** objeto a ser **removido** de uma fila é o **primeiro** que foi **inserido**. Esta política de manipulação é conhecida pela sigla **FIFO** (=First In First Out)



Implementação em um vetor

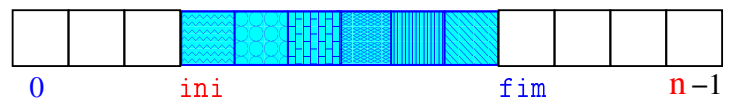
A fila será armazenada em um vetor $q[0 \dots n-1]$.



- O índice **ini** indica o **primeiro** da fila.
- O índice **fim-1** indica o **último** da fila.
- fim** é a **primeira posição vaga** da fila.
- A fila está **vazia** se "**ini == fim**".
- A fila está **cheia** se "**fim == n**".

Implementação em um vetor

A fila será armazenada em um vetor $q[0 \dots n-1]$.



Para **remover** (=dequeue=get) um elemento faça

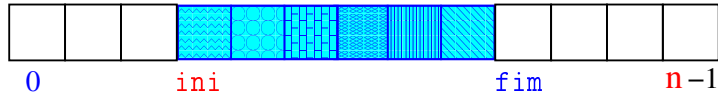
```
x = q[ini++];
```

que é equivalente a

```
x = q[ini];
ini += 1;
```

Implementação em um vetor

A fila será armazenada em um vetor $q[0 \dots n-1]$.



Para *inserir* (= *queue=put*) um elemento faça

```
q[fim++] = x;
```

que é equivalente a

```
q[fim] = x;
fim += 1;
```

< > < > < > < > < > < > < >

< > < > < > < > < > < > < >

Implementações de filas

PF 5.3

<http://www.ime.usp.br/~pf/algoritmos/aulas/fila.html>

< > < > < > < > < > < > < >

Interface item.h

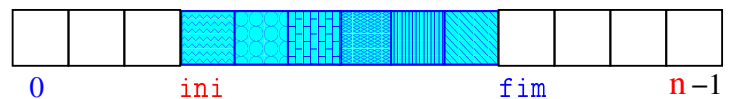
```
/*
 * item.h
 */
typedef int Item;
```

< > < > < > < > < > < > < >

AULA 15

Fila implementada em um vetor

A fila será armazenada em um vetor $q[0 \dots n-1]$.



O índice *ini* indica o *primeiro* da fila.

O índice *fim-1* indica o *último* da fila.

fim é a *primeira posição vaga* da fila.

A fila está *vazia* se "*ini == fim*".

A fila está *cheia* se "*fim == n*".

< > < > < > < > < > < > < >

Interface queue.h

```
/*
 * queue.h
 * INTERFACE: funcoes para manipular uma
 * fila
 */
void queueInit(int);
int queueEmpty();
void queuePut(Item);
Item queueGet();
void queueFree();
```

< > < > < > < > < > < > < >

Implementação queue.c

```
#include <stdlib.h>
#include <stdio.h>
#include "item.h"
/*
 * FILA: uma implementacao em lista
 * encadeada circular com cabeca
 */
typedef struct queueNode* Link;
struct queueNode {
    Item conteudo;
    Link prox;
};
static Link ini;
```

Implementação queue.c

```
void
queuePut(Item item)
{
    Link nova = mallocSafe(sizeof *nova);

    nova->prox = ini->prox;
    ini->prox = nova;

    /* insira item na celula cabeca (!) */
    ini->conteudo = item;

    /* mude a cabeca para nova (!) */
    ini = nova;
}
```

Implementação queue.c

```
void
queueFree()
{
    Link p = ini->prox;

    while (p != ini)
    {
        Link t = p->prox;
        free(p);
        p = t;
    }
    free(ini);
}
```

Implementação queue.c

```
void
queueInit(int n)
{
    ini = mallocSafe(sizeof *ini);
    ini->prox = ini;
}

int
queueEmpty()
{
    return ini->prox == ini;
}
```

Implementação queue.c

```
Item
queueGet()
{
    Link p = ini->prox;
    Item item = p->conteudo;

    ini->prox = p->prox;
    free(p);
    return item;
}
```

Compilação

```
cria o obj queue.o
> gcc -Wall -O2 -ansi -pedantic
-Wno-unused-result -c queue.c

cria o obj distancias.o
> gcc -Wall -O2 -ansi -pedantic
-Wno-unused-result \
-c distancias.c

cria o executável polonesa
> gcc -o distancia queue.o distancias.o
```

Makefile

Hmmm. Ler o tópico [Makefile](#) no fórum.

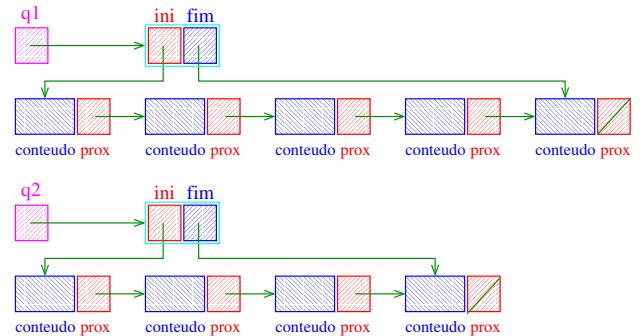
```
distancia: distancia.o queue.o
    gcc distancia.o queue.o -o distancia

distancia.o: distancia.c
    gcc -Wall -O2 -ansi -pedantic \
        -Wno-unused-result -c distancia.c

queue.o: queue.c item.h
    gcc -Wall -O2 -ansi -pedantic \
        -Wno-unused-result -c queue.c
```

FilaS implementadaS em listaS encadeadaS

As filas serão armazenada em lista encadeada.



FilaS implementadaS em listaS encadeadaS

Uma fila q é um ponteiro para uma `struct` com campos `ini` e `fim`.

Para cada fila q há ponteiros `ini` e `fim`.

$q \rightarrow ini \rightarrow conteudo$ é o primeiro elemento da fila q .

$q \rightarrow fim \rightarrow conteudo$ é o último elemento da fila q .

A fila q está vazia se " $q \rightarrow ini == NULL$ ".

Interface item.h

```
/*
 * item.h
 */
typedef int Item;
```

Interface queue.h

```
/*
 * queue.h
 * INTERFACE: funcoes para manipular filas
 * ATENCAO: Esta interface permite que
 * varias filas sejam utilizadas.
 */
typedef struct queue *Queue;
Queue queueInit(int);
int queueEmpty(Queue);
void queuePut(Queue, Item);
Item queueGet(Queue);
void queueFree(Queue);
```

distancias

A função `distancias` recebe um inteiro n , uma matriz A representado um rede de estradas entre n cidades e uma cidade c e devolve um vetor d que registra a distancia da cidade c a cada uma das outras: $d[i]$ é a distância de c a i .

```
int *
distancias (int n, int **A, int c) {
    int *d; /* d[i] = distancia de c a i*/
    int j;
    Queue q;
```