

Filas e filas circulares– *Queues*

Programação de computadores II

Prof. Renan Augusto Starke

Instituto Federal de Santa Catarina – IFSC

Campus Florianópolis

`renan.starke@ifsc.edu.br`

26 de março de 2018



INSTITUTO FEDERAL
SANTA CATARINA

Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
INSTITUTO FEDERAL DE SANTA CATARINA

Tópicos da aula

1 Introdução

2 Filas

3 Exemplo

4 Exercícios

1 Introdução

2 Filas

3 Exemplo

4 Exercícios

- Entender o conceito de filas – *queue*
- Aprender sua utilização
- Aprender sua implementação
- Utilizar o conceito de “dados abstratos e estruturados” para fornecer funções simples de manipulação de filas

1 Introdução

2 Filas

3 Exemplo

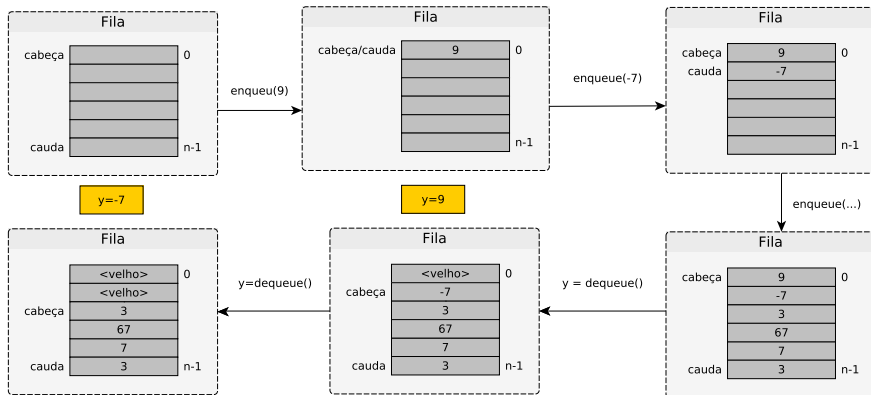
4 Exercícios

Fila – *queue* – é uma estrutura de dados simples onde os dados são “enfileirados”. Idealmente implementa o conceito de FIFO – *First-in, First-out*: primeiro a entrar é o primeiro a sair.

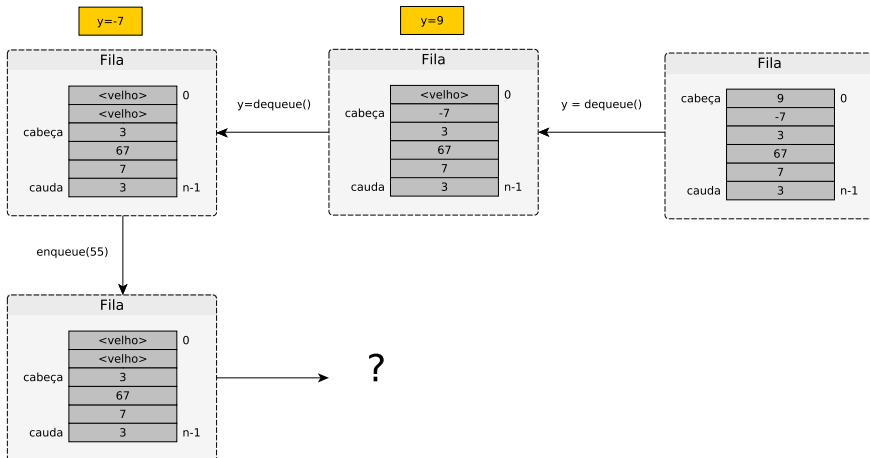
Funções básicas:

- *enqueue*: adiciona um elemento
- *dequeue*: remove um elemento

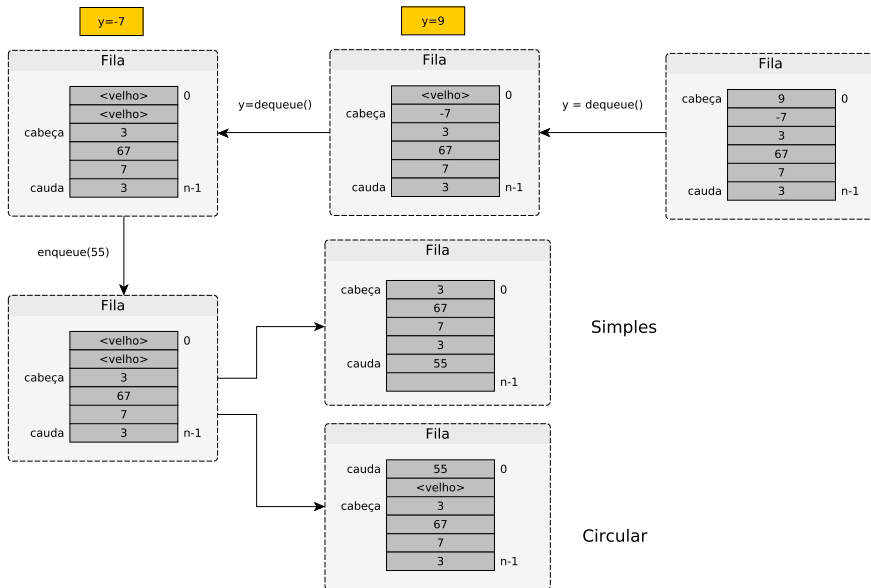
Filas



Filas



Filas



Considerações com implementação com *array*:

- A região de elementos contínuos não necessariamente ocupará as posições iniciais do *array*
 - Em algumas situações as posições “farão a volta”.
 - Ou devemos **deslocar todos os elementos**.
- Dado um *array* de tamanho n , então:
 - $0 \leq \text{cabeca} < n$
 - $0 \leq \text{cauda} < n$
 - $0 \leq \text{cabeca} - \text{cauda} \leq n - 1$
 - Como há apenas n diferenças possíveis, poderá haver $0, 1, \dots, n - 1$ comprimentos de fila.
- Dado estas considerações, é impossível descobrir o tamanho da fila observando somente os valores da cabeça e cauda.
- Quando há somente um elemento: $\text{cauda} = \text{cabeça}$

Funções adicionais:

- *inicio*: retorna o valor do elemento da cabeça da fila
- *tamanho*: retorna tamanho da fila
- *vazia*: retorna se fila está vazia

1 Introdução

2 Filas

3 Exemplo

4 Exercícios

Exemplo – fila.h

```
#ifndef FILA_H_INCLUDED
#define FILA_H_INCLUDED

typedef struct filas fila_t;

int dequeue(fila_t* fila);
void enqueue(int x, fila_t* fila);

fila_t* cria_fila(int tamanho);
void libera_fila(fila_t* fila);

#endif // FILA_H_INCLUDED
```

Exemplo – fila.c: struct fila

```
#include <stdio.h>
#include <stdlib.h>

#include "fila.h"

struct filas {
    int cabeca;
    int cauda;
    int tamanho;
    int tamanho_max;
    int *dados;
};
```

Exemplo – fila.c: cria_fila

```
//cria uma nova fila
fila_t* cria_fila(int tamanho)
{
    fila_t *p = NULL;

    //aloca memoria
    p = (fila_t*)malloc(sizeof(fila_t));

    //variaveis de controle
    p->cabeca = 0;
    p->cauda = tamanho-1;
    p->tamanho = 0;
    p->tamanho_max = tamanho;

    p->data = (int) malloc(sizeof(int) * tamanho);

    return p;
}
```

Exemplo – fila.c: libera

```
//libera memoria
void libera_fila(fila_t* fila)
{
    fila_t *p = fila;

    if (p == NULL)
    {
        fprintf(stderr, "Erro desalocando memoria da fila, ponteiro invalido!");
        exit(EXIT_FAILURE);
    }

    free(p->dados);
    free(p);
}
```


Exemplo – fila.c: main

```
int main()
{
    fila_t* fila = NULL;

    fila = cria_fila(10);

    enqueue(fila, 10)
    enqueue(fila, 45)
    enqueue(fila, 33)

    x = dequeue(fila)
    printf("%d\n", x)
    x = dequeue(fila)
    x = dequeue(fila)

    //(...)

    libera_fila(fila);

    return 0;
}
```

Tópico

1 Introdução

2 Filas

3 Exemplo

4 Exercícios

Baseando-se na implementação exemplo desta aula, implemente uma fila através de uma lista simplesmente encadeada com as seguintes interfaces:

- *enqueue*: enfileirar
- *dequeue*: desenfileirar
- *inicio*: retorna o valor do elemento do inicio da fila
- *tamanho*: retorna tamanho da fila
- *vazia*: retorna se a fila está vazia

Exercício

Baseando-se na implementação exemplo desta aula, implemente uma fila **através de uma lista simplesmente encadeada** com as seguintes interfaces:

- *fila_t * cria_fila()*: cria uma fila genérica (aceita qualquer ponteiro de dados) vazia
- *enqueue (fila_t * fila, void *dado)*: enfileira um elemento
- *void * dequeue(fila_t *fila)*: desenfileira um elemento
- *void * inicio (fila_t *fila)* :retorna o valor do elemento do topo da fila
- *int tamanho()*: retorna tamanho da fila
- *int vazia()*: retorna se a fila está vazia
- Você deve construir sua fila utilizando o conceito de dados abstratos:
 - typedef struct filas fila_t
 - fila.c
 - fila.h

Após a implementação da fila genérica através de uma lista encadeada:

- Modifique o exemplo, implementado com lista encadeada e pilha, para usar sua fila.
 - A função de leitura de arquivo deve agora retornar uma fila