

Curso: Ciência da Computação

Disciplina: Estrutura de Dados 1

Professor: Clayton Zambon

5. Fila (Queue)

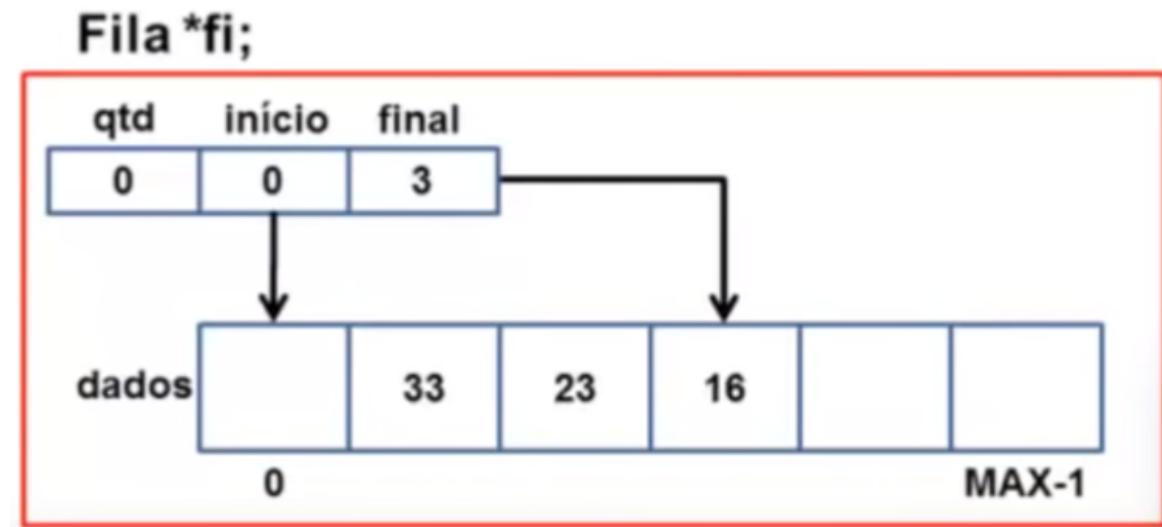
5.2. Alocação Estática;

5. Fila (Queue)

5.2. Fila com Alocação Estática

- Fila Estática:

- Tipo de Fila onde o sucessor de um elemento ocupa a posição física seguinte do mesmo (uso de array);



5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando uma “Fila Estática”:

- **FilaSequencial.h: definir**

- Protótipos das funções;
- O tipo de dado armazenado na Fila;
- O ponteiro da Fila;
- Tamanho do vetor usado na Fila;

- **FilaSequencial.c: definir**

- O tipo de dados Fila;
- Implementar as funções;

- **main.c: definir**

- Fazer as chamadas das funções;

5. Fila (Queue)

5.2. Fila com Alocação Estática

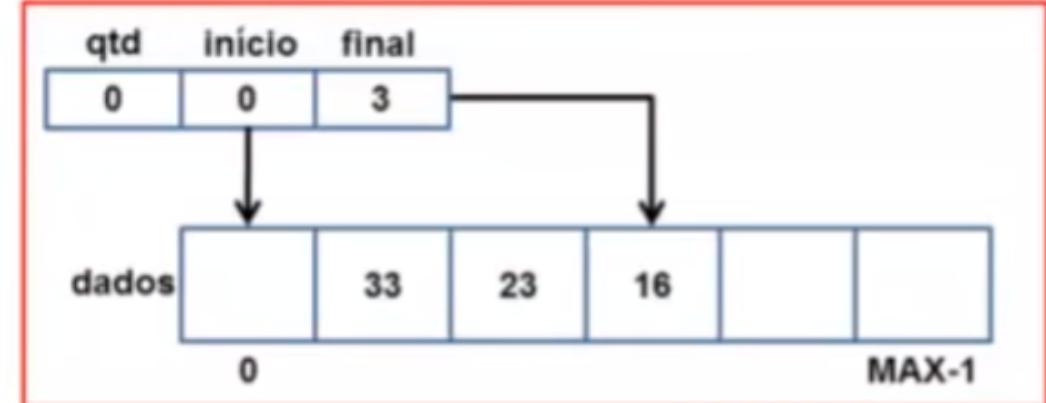
- Definindo os Tipos, Structs e Ponteiro;

```
1 //Arquivo FilaEstatica.h
2 #define MAX 100
3 struct aluno{
4     int matricula;
5     char nome[30];
6     float nl,n2,n3;
7 };
8 typedef struct fila Fila;
9 //*****
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "FilaEstatica.h" //inclui os Protótipos
4
5 //Definição do tipo Fila
6 struct fila{
7     int inicio, final;
8     struct aluno dados[MAX];
9 };
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "FilaEstatica.h"
4 int main(){
5     Fila *fi;
6 }
7 }
```

Fila *fi;



5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “cria_Fila”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila(); ←
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
16 //Arquivo FilaEstatica.c
17 Fila* cria_Fila(){
18     Fila *fi;
19     fi = (Fila*) malloc(sizeof(struct fila));
20     if(fi != NULL){
21         fi->inicio = 0;
22         fi->final = 0;
23     }
24     return fi;
25 }
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "FilaEstatica.h"
4 int main(){
5     Fila *fi;
6     fi = cria_Fila();
7 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “libera_Fila”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi); ←
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
26 //Arquivo FilaEstatica.c
27 void libera_Fila(Fila* fi) {
28     free(fi);
29 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main() {
6     Fila *fi;
7     fi = cria_Fila();
8     libera_Fila(fi);
9 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “tamanho_Fila”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi); ←
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
30 //Arquivo FilaEstatica.c
31 int tamanho_Fila(Fila* fi){
32     if(fi == NULL)
33         return -1;
34     int qtd = abs(fi->final - fi->inicio);
35     return qtd;
36 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main(){
6     Fila *fi;
7     fi = cria_Fila();
8     int x = tamanho_Fila(fi);
9     libera_Fila(fi);
10 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “Fila_cheia”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi); ←
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
32 //Arquivo FilaEstatica.c
33 int Fila_cheia(Fila* fi){
34     if(fi == NULL)
35         return -1;
36     if (fi->inicio == (fi->final+1)%MAX)
37         return 1;
38     else
39         return 0;
40 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main(){
6     Fila *fi;
7     fi = cria_Fila();
8     int x = tamanho_Fila(fi);
9     Fila_cheia(fi); ←
10    libera_Fila(fi);
11 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “Fila_vazia”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi); ←
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
41 //Arquivo FilaEstatica.c
42 int Fila_vazia(Fila* fi) {
43     if(fi == NULL)
44         return -1;
45     return (fi->inicio == fi->final);
46 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main() {
6     Fila *fi;
7     fi = cria_Fila();
8     int x = tamanho_Fila(fi);
9     Fila_cheia(fi);
10    Fila_vazia(fi); ←
11    libera_Fila(fi);
12 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “`insere_Fila`”: (Enqueue)

- Lembre-se que em uma Fila a inserção é sempre feita no final;
- Em uma Fila estática não podemos inserir quando está cheia;



5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “insere_Fila”: (Enqueue)

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al); ←
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
47 //Arquivo FilaEstatica.c
48 int insere_Fila(Fila* fi, struct aluno al){
49     if(fi == NULL)
50         return 0;
51     if(Fila_cheia(fi))
52         return 0;
53     fi->final = (fi->final+1)%MAX;
54     fi->dados[fi->final] = al;
55 }
56 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main(){
6     Fila *fi;
7     struct aluno dados_aluno;
8     fi = cria_Fila();
9     int x = tamanho_Fila(fi);
10    Fila_cheia(fi);
11    Fila_vazia(fi);
12    x = insere_Fila(fi, dados_aluno); ←
13    libera_Fila(fi);
14 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “remove_Fila”: (Dequeue)

- Em uma Fila a remoção é sempre no seu início;
- Não é possível remover de uma Fila vazia;



5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “remove_Fila”: (Dequeue)

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi); ←
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
```

```
57 //Arquivo FilaEstatica.c
58 int remove_Fila(Fila* fi){
59     if(fi == NULL || Fila_vazia(fi))
60         return 0;
61     fi->inicio = (fi->inicio+1)%MAX;
62     return 1;
63 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main(){
6     Fila *fi;
7     struct aluno dados_aluno;
8     fi = cria_Fila();
9     int x = tamanho_Fila(fi);
10    Fila_cheia(fi);
11    Fila_vazia(fi);
12    x = insere_Fila(fi, dados_aluno);
13    x = remove_Fila(fi); ←
14    libera_Fila(fi);
15 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “consulta_Fila”:

- Em uma Fila a consulta se dá apenas ao elemento do início;

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi);
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 //Arquivo FilaEstatica.c
65 int consulta_Fila(Fila* fi, struct aluno *al) {
66     if(fi == NULL || Fila_vazia(fi))
67         return 0;
68     int pos = (fi->inicio+1)%MAX;
69     *al = fi->dados[pos];
70     return 1;
71 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main() {
6     Fila *fi;
7     struct aluno dados_aluno;
8     fi = cria_Fila();
9     int x = tamanho_Fila(fi);
10    Fila_cheia(fi);
11    Fila_vazia(fi);
12    x = insere_Fila(fi, dados_aluno);
13    x = consulta_Fila(fi, &dados_aluno);
14    x = remove_Fila(fi);
15    libera_Fila(fi);
16 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “`consulta_Fila`”:

- Em uma Fila a consulta se dá apenas ao elemento do início;



5. Fila (Queue)

5.2. Fila com Alocação Estática

Implementando a função “imprime_Fila”:

```
10 //Arquivo FilaEstatica.h
11 Fila* cria_Fila();
12 void libera_Fila(Fila* fi);
13 int tamanho_Fila(Fila* fi);
14 int Fila_cheia(Fila* fi);
15 int Fila_vazia(Fila* fi);
16 int insere_Fila(Fila* fi, struct aluno al);
17 int remove_Fila(Fila* fi);
18 int consulta_Fila(Fila* fi, struct aluno *al);
19 void imprime_Fila(Fila* fi); ←
```

```
72 //Arquivo FilaEstatica.c
73 void imprime_Fila(Fila* fi){
74     if(fi == NULL)
75         return;
76     int i = fi->inicio;
77     while(i != fi->final){
78         i = (i + 1) % MAX;
79         printf("Matricula: %d\n", fi->dados[i].matricula);
80         printf("Nome: %s\n", fi->dados[i].nome);
81         printf("Notas: %f %f %f\n", fi->dados[i].n1,
82                                         fi->dados[i].n2,
83                                         fi->dados[i].n3);
84         printf("-----\n");
85     }
86 }
```

```
1 //Arquivo main.c
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include "FilaEstatica.h"
5 int main(){
6     Fila *fi;
7     struct aluno dados_aluno;
8     fi = cria_Fila();
9     int x = tamanho_Fila(fi);
10    Fila_cheia(fi);
11    Fila_vazia(fi);
12    x = insere_Fila(fi, dados_aluno);
13    x = consulta_Fila(fi, &dados_aluno);
14    imprime_Fila(fi); ←
15    x = remove_Fila(fi);
16    libera_Fila(fi);
17 }
```

5. Fila (Queue)

5.2. Fila com Alocação Estática

EXERCÍCIO:

- Fazer um menu para acessar as funções;
- Fazer as implementações descritas anteriormente;
- Popular a fila com informações do aluno;
- Implementar mensagens para informar o usuário as ações que foram realizadas;
- Ao consultar pela posição ou matrícula, exibir as informações do elemento;
- Quando não encontrar um elemento exibir uma mensagem para o usuário;
- Quando não for possível realizar a operação, exibir a mensagem de fila vazia ou fila cheia conforme o caso.
- Quando inserir ou remover um elemento informar ao usuário se a operação ocorreu com sucesso ou não.

Referências

- EDELWEISS, Nina; GALANTE, Renata. Estruturas de Dados. Porto Alegre, BOOKMAN, 2009.
- HEINZLE, Roberto. Estruturas de Dados: implementações com C e Pascal. Blumenau, DIRETIVA, 2006.
- TENENBAUM, Aron M. Estrutura de Dados usando C. São Paulo, Makron Books, 1995.
- FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. Lógica de Programação: a construção de algoritmos e estruturas de dados. 3. ed. São Paulo, PRENTICE HALL, 2005.
- KOFFMAN, Elliot B.; WOLFGANG, Paul A. T. Objetos, abstração, estruturas de dados e projeto usando C++. Rio de Janeiro, LTC, 2008.
- PEREIRA, Silvio do Iago. Estruturas de dados fundamentais: conceitos e aplicações. São Paulo, Érica, 1996.
- VILLAS, Marcos Viana et al. Estruturas de dados – Conceitos e técnicas de implementação. Rio de Janeiro, Campus, 1993.
- VELOSO, Paulo et al. Estrutura de dados. Rio de Janeiro, Campus, 1996.
- Canal do Youtube: Linguagem C Programação Descomplicada
- Site: <https://programacaodescomplicada.wordpress.com/>

Obrigado!