Curso: Ciência da Computação

Disciplina: Estrutura de Dados 1

Professor: Clayton Zambon

centro universitário unifacvest

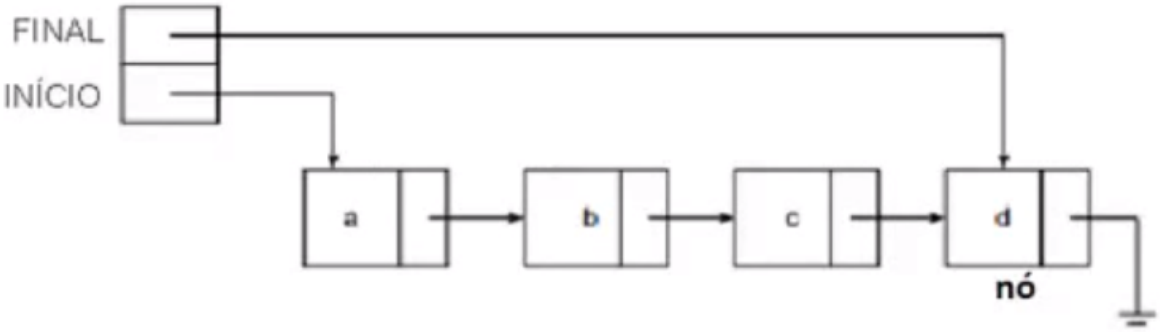
5. Fila (Queue)

5.3. Alocação Dinâmica;

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

- Fila Dinâmica:

- Tipo de Fila onde cada elemento aponta para o seu sucessor na Fila;
- Usa um Nó descritor para representar o início e final da fila e uma indicação de final de Fila



5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando uma “Fila Dinâmica”:

- FilaDin.h: definir

- Protótipos das funções;
- O tipo de dado armazenado na Fila;
- O ponteiro Fila;

- FilaDin.c: definir

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

- main.c: definir

- Fazer as chamadas das funções;

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

- Definindo os Tipos, Structs e Ponteiro;

```c
//Arquivo FilaDin.h
struct aluno{
    int matricula;
    char nome[30];
    float n1,n2,n3;
};

typedef struct fila Fila;
```

```c
//Arquivo FilaDin.c
#include <stdio.h>
#include <stdlib.h>
#include "FilaDin.h" //inclui os Protótipos
//Definição do tipo Fila
struct elemento{
    struct aluno dados;
    struct elemento *prox;
};
typedef struct elemento Elem;
//Definição do Nó Descritor da Fila
struct fila{
    struct elemento *inicio;
    struct elemento *final;
    int qtd;
};
```

```c
//Arquivo main.c
#include <stdio.h>
#include <stdlib.h>
#include "FilaDin.h"
int main(){
    Fila *fi;
}
```

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "cria_Fila":

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

```
17  //Arquivo FilaDin.c
18  Fila* cria_Fila(){
19      Fila* fi = (Fila*) malloc(sizeof(Fila));
20      if(fi != NULL){
21          fi->final = NULL;
22          fi->inicio = NULL;
23          fi->qtd = 0;
24      }
25      return fi;
26  }
```

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "libera_Fila":

```c
//Arquivo FilaDin.h
Fila* cria_Fila();
void libera_Fila(Fila* fi);
int tamanho_Fila(Fila* fi);
int Fila_cheia(Fila* fi);
int Fila_vazia(Fila* fi);
int insere_Fila(Fila* fi, struct aluno al);
int remove_Fila(Fila* fi);
int consulta_Fila(Fila* fi, struct aluno *al);
void imprime_Fila(Fila* fi);
```

```c
//Arquivo FilaDin.c
void libera_Fila(Fila* fi){
    if(fi != NULL){
        Elem* no;
        while(fi->inicio != NULL){
            no = fi->inicio;
            fi->inicio = fi->inicio->prox;
            free(no);
        }
        free(fi);
    }
}
```

```c
//Arquivo main.c
#include <stdio.h>
#include <stdlib.h>
#include "FilaDin.h"
int main(){
    Fila *fi;
    fi = cria_Fila();
    libera_Fila(fi);
}
```

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "tamanho_Fila":

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

```
39  //Arquivo FilaDin.c
40  int tamanho_Fila(Fila* fi){
41      if(fi == NULL)
42          return 0;
43      return fi->qtd;
44  }
```

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "Fila_cheia":

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

```
45  //Arquivo FilaDin.c
46  int Fila_cheia(Fila* fi){
47      return 0;
48  }
```

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "Fila_vazia":

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

```
49  //Arquivo FilaDin.c
50  int Fila_vazia(Fila* fi){
51      if(fi == NULL)
52          return 1;
53      if(fi->inicio == NULL)
54          return 1;
55      return 0;
56  }
```

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "insere_Fila": (Enqueue)

- Lembre-se que em uma Fila a inserção é sempre feita no final;

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "insere_Fila": (Enqueue)

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

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

```
57  //Arquivo FilaDin.c
58  int insere_Fila(Fila* fi, struct aluno al){
59      if(fi == NULL)
60          return 0;
61      Elem *no = (Elem*) malloc(sizeof(Elem));
62      if(no == NULL)
63          return 0;
64      no->dados = al;
65      no->prox = NULL;
66      if(fi->final == NULL)//fila vazia
67          fi->inicio = no;
68      else
69          fi->final->prox = no;
70      fi->final = no;
71      fi->qtd++;
72      return 1;
73  }
```

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

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)
5.3. Fila com Alocação Dinâmica

Implementando a função "remove_Fila": (Dequeue)

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

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

```
74  //Arquivo FilaDin.c
75  int remove_Fila(Fila* fi){
76      if(fi == NULL)
77          return 0;
78      if(fi->inicio == NULL)//fila vazia
79          return 0;
80      Elem *no = fi->inicio;
81      fi->inicio = fi->inicio->prox;
82      if(fi->inicio == NULL)//fila ficou vazia
83          fi->final = NULL;
84      free(no);
85      fi->qtd--;
86      return 1;
87  }
```

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "consulta_Fila":

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "consulta_Fila":

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

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

```
88  //Arquivo FilaDin.c
89  int consulta_Fila(Fila* fi, struct aluno *al){
90      if(fi == NULL)
91          return 0;
92      if(fi->inicio == NULL)//fila vazia
93          return 0;
94      *al = fi->inicio->dados;
95      return 1;
96  }
```

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

5. Fila (Queue)
5.3. Fila com Alocação Dinâmica

Implementando a função "imprime_Fila":

```c
//Arquivo FilaDin.h
Fila* cria_Fila();
void libera_Fila(Fila* fi);
int tamanho_Fila(Fila* fi);
int Fila_cheia(Fila* fi);
int Fila_vazia(Fila* fi);
int insere_Fila(Fila* fi, struct aluno al);
int remove_Fila(Fila* fi);
int consulta_Fila(Fila* fi, struct aluno *al);
void imprime_Fila(Fila* fi);
```

```c
//Arquivo main.c
#include <stdio.h>
#include <stdlib.h>
#include "FilaDin.h"
int main(){
    Fila *fi;
    struct aluno dados_aluno;
    fi = cria_Fila();
    tamanho_Fila(fi);
    int x = Fila_cheia(fi);
    x = Fila_vazia(fi);
    x = insere_Fila(fi, dados_aluno);
    consulta_Fila(fi, dados_aluno);
    imprime_Fila(fi);
    x = remove_Fila(fi);
    libera_Fila(fi);
}
```

```c
//Arquivo FilaDin.c
void imprime_Fila(Fila* fi){
    if(fi == NULL)
        return;
    Elem* no = fi->inicio;
    while(no != NULL){
        printf("Matricula: %d\n",no->dados.matricula);
        printf("Nome: %s\n",no->dados.nome);
        printf("Notas: %f %f %f\n",no->dados.n1,
                                   no->dados.n2,
                                   no->dados.n3);
        printf("-------------------------------\n");
        no = no->prox;
    }
}
```

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 lago. 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!