UFU

Nome: Antonio Carlos Neto

Curso: Ciência da Computação

Disciplina: GBC024

PARTE 1

MAIN.C :

#include <stdio.h>

#include <stdlib.h>

#include <sys/time.h>

#define MAXTAM 1000

typedef int TipoApontador;

typedef int TipoChave;

typedef struct {

TipoChave Chave;

/\* outros componentes \*/

} TipoItem;

typedef struct {

TipoItem Item[MAXTAM];

TipoApontador Frente, Tras;

} TipoFila;

void FFVazia(TipoFila \*Fila)

{ Fila->Frente = 1;

Fila->Tras = Fila->Frente;

}

int Vazia(TipoFila Fila)

{ return (Fila.Frente == Fila.Tras); }

void Enfileira(TipoItem x, TipoFila \*Fila)

{ if (Fila->Tras % MAXTAM + 1 == Fila->Frente)

printf(" Erro fila est a cheia\n");

else { Fila->Item[Fila->Tras - 1] = x;

Fila->Tras = Fila->Tras % MAXTAM + 1;

}

}

void Desenfileira(TipoFila \*Fila, TipoItem \*Item)

{ if (Vazia(\*Fila))

printf("Erro fila esta vazia\n");

else { \*Item = Fila->Item[Fila->Frente - 1];

Fila->Frente = Fila->Frente % MAXTAM + 1;

}

}

void Imprime(TipoFila Fila)

{ int Aux;

for (Aux = Fila.Frente - 1; Aux <= (Fila.Tras - 2); Aux++)

printf("%12d\n", Fila.Item[Aux].Chave);

} /\* Imprime \*/

int main(int argc, char \*argv[])

{ struct timeval t;

TipoFila fila;

TipoItem item;

int vetor[MAXTAM];

int i, j, k, n, max;

max = 4;

FFVazia(&fila);

for(i = 0; i < max; i++) vetor[i] = i + 1;

for (i = 0; i < max; i++)

{ item.Chave = vetor[i];

Enfileira(item, &fila);

printf("Enfileirou: %d \n", item.Chave);

}

for(i = 0; i < max; i++)

{ Desenfileira(&fila, &item);

printf("Desenfileirou: %d\n", item.Chave);

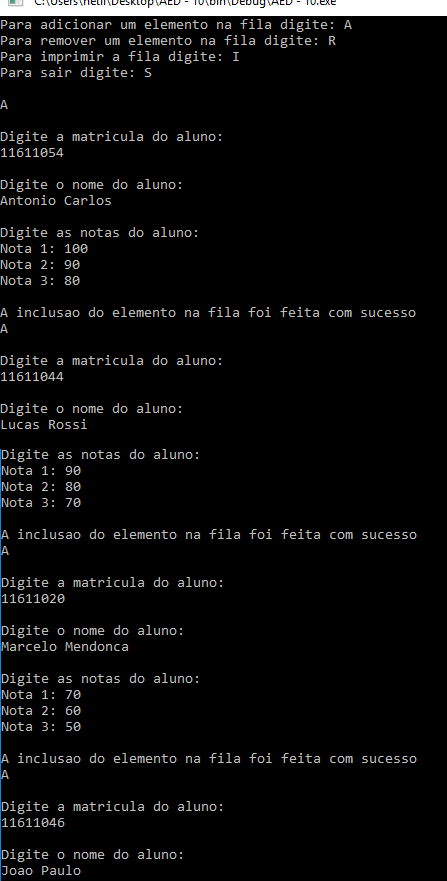
}

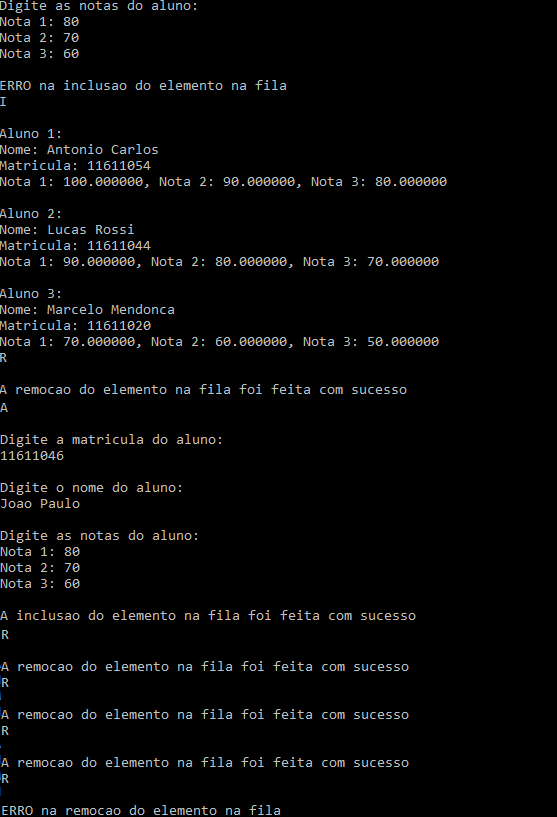
return(0);

}

PARTE 2

EXECUÇÃO:





MAIN.C :

#include <stdio.h>

#include <stdlib.h>

#include "Fila.h"

int main()

{

Fila \*fi;

int x;

char u;

fi = cria\_Fila();

printf("Para adicionar um elemento na fila digite: A");

printf("\nPara remover um elemento na fila digite: R");

printf("\nPara imprimir a fila digite: I");

printf("\nPara sair digite: S\n\n");

do

{

//printf("\n");

fflush(stdin);

scanf("%c",&u);

if(u == 'A')

{

struct aluno al;

printf("\nDigite a matricula do aluno:\n");

fflush(stdin);

scanf("%d",&al.matricula);

fflush(stdin);

printf("\nDigite o nome do aluno:\n");

fflush(stdin);

gets(al.nome);

fflush(stdin);

printf("\nDigite as notas do aluno:\nNota 1: ");

fflush(stdin);

scanf("%f",&al.n1);

fflush(stdin);

printf("Nota 2: ");

fflush(stdin);

scanf("%f",&al.n2);

fflush(stdin);

printf("Nota 3: ");

fflush(stdin);

scanf("%f",&al.n3);

fflush(stdin);

x = insere\_Fila(fi,al);

if(x == 0)

{

printf("\nERRO na inclusao do elemento na fila\n");

}

if(x == 1)

{

printf("\nA inclusao do elemento na fila foi feita com sucesso\n");

}

}else

if(u == 'R')

{

x = remove\_Fila(fi);

if(x == 0)

{

printf("\nERRO na remocao do elemento na fila\n");

}

if(x == 1)

{

printf("\nA remocao do elemento na fila foi feita com sucesso\n");

}

}else

if(u == 'I')

{

print\_Fila(fi);

}

}

while(u!='S');

libera\_Fila(fi);

return 0;

}

Fila.c :

#include <stdio.h>

#include <stdlib.h>

#include "Fila.h"

struct fila

{

int inicio, final, qtd;

struct aluno dados[MAX];

};

Fila\* cria\_Fila()

{

Fila \*fi = (Fila\*) malloc(sizeof(struct fila));

if(fi != NULL)

{

fi->inicio = 0;

fi->final = 0;

fi->qtd = 0;

}

return fi;

}

void libera\_Fila(Fila\* fi)

{

free(fi);

}

int tamanho\_Fila(Fila\* fi)

{

if(fi == NULL)

return -1;

return fi->qtd;

}

int Fila\_cheia(Fila\* fi)

{

if(fi == NULL) return -1;

if(fi->qtd == MAX)

return 1;

else

return 0;

}

int Fila\_vazia(Fila\* fi)

{

if(fi == NULL) return -1;

if(fi->qtd == 0)

return 1;

else

return 0;

}

int insere\_Fila(Fila\* fi, struct aluno al)

{

if(fi == NULL) return 0;

if(Fila\_cheia(fi)) return 0;

fi->dados[fi->final] = al;

fi->final = (fi->final+1)%MAX;

fi->qtd++;

return 1;

}

int remove\_Fila(Fila\* fi)

{

if(fi == NULL || Fila\_vazia(fi))

return 0;

fi->inicio = (fi->inicio+1)%MAX;

fi->qtd--;

return 1;

}

int consulta\_Fila(Fila\* fi, struct aluno \*al)

{

if(fi == NULL || Fila\_vazia(fi))

return 0;

\*al = fi->dados[fi->inicio];

return 1;

}

void print\_Fila(Fila\* fi)

{

if(fi == NULL || Fila\_vazia(fi))

printf("\nFila vazia ou inexistente.\n");

else

{

int i;

for(i=0; i<fi->qtd; i++)

{

printf("\nAluno %d:",i+1);

printf("\nNome: %s",fi->dados[(fi->inicio+i)%MAX].nome);

printf("\nMatricula: %d",fi->dados[(fi->inicio+i)%MAX].matricula);

printf("\nNota 1: %f, Nota 2: %f, Nota 3: %f\n",fi->dados[(fi->inicio+i)%MAX].n1,fi->dados[(fi->inicio+i)%MAX].n2,fi->dados[(fi->inicio+i)%MAX].n3);

}

}

}

Fila.h:

#ifndef FILA\_H\_INCLUDED

#define FILA\_H\_INCLUDED

#define MAX 3

struct aluno

{

int matricula;

char nome[30];

float n1,n2,n3;

};

typedef struct fila Fila;

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 print\_Fila(Fila\* fi);

#endif // FILA\_H\_INCLUDED