

Nome: João Paulo de Oliveira

11611BCC046

10º Aula prática

Uberlândia

2016

1.Código fonte:

1.1 Parte 1:

- **Main.c:**

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>

#define MAXTAM 4

typedef int TipoApontador;

typedef int TipoChave;
typedef struct {
    TipoChave Chave;
    /* outros componentes */
} TipoItem;
typedef struct {
    TipoItem Item[10];
    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 % 10 + 1 == Fila->Frente)
    printf(" Erro  fila est a cheia\n");
  else { Fila->Item[Fila->Tras - 1] = x;
        Fila->Tras = Fila->Tras % 10 + 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 % 10 + 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[10];
  int i, j, k, n, max;

  gettimeofday(&t, NULL);
  srand((unsigned int)t.tv_usec);
  max = 9;
  FFVazia(&fila);

  /*Gera uma permutacao aleatoria de chaves entre 1 e max*/
  for(i = 0; i < max+1; i++) vetor[i] = i + 1;
  for(i = 0; i < max; i++){
    k = (int) (10.0 * rand()/(RAND_MAX + 1.0));
    j = (int) (10.0 * rand()/(RAND_MAX + 1.0));
    n = vetor[k];
    vetor[k] = vetor[j];
    vetor[j] = n;
  }
  /*Enfileira cada chave */
  for (i = 0; i < max; i++){
    item.Chave = vetor[i];
    Enfileira(item, &fila);
    printf("Enfileirou: %d \n", item.Chave);
  }
  /*Desenfileira cada chave */
  for(i = 0; i < max; i++)
  { Desenfileira(&fila, &item);
    printf("Desenfileirou: %d\n", item.Chave);
  }
}

```

```

    return(0);
}

```

1.2 Parte 2:

- **Main.c:**

```

#include <stdio.h>
#include <stdlib.h>
#include "lista.h"
int main(){
    Fila* fila;
    fila = cria_Fila();
    int el=0,i;
    for(i=0;i<5;i++){
        scanf("%d",&el);
        if (i==0)fila->inicio = el;
        else insere_Fila(fila,el);
        printf("inserido: %d\n",el);
        if(i==3){
            printf("removido: %d\n",fila->inicio);
            remove_Fila(fila);
        }
    }

    for(i=0;i<4;i++){
        printf("removido: %d\n",fila->inicio);
        remove_Fila(fila);
    }
    libera_Fila(fila);
    return 0;
}

```

- **fila.c:**

```

#include "lista.h"
#include <stdio.h>
#include <stdlib.h>
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, int elemento){
        if(fi == NULL)return 0;
        if(Fila_cheia(fi))return 0;
        fi->elemento = elemento;
        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, int elemento){
        if(fi == NULL||Fila_vazia(fi))return 0;
        elemento = fi->elemento;
        return 1;
    }
}

```

- **fila.h:**
#ifndef LISTA_H_INCLUDED

```

#define LISTA_H_INCLUDED
#define MAX 100

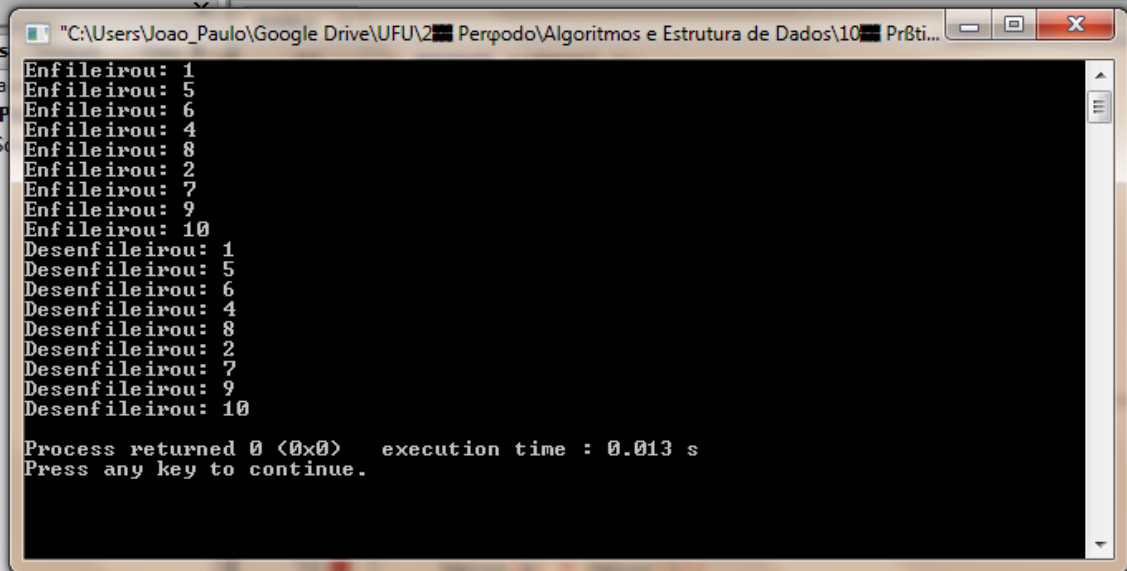
struct fila{
    int inicio,final,qtd;
    int elemento;
};
typedef struct fila Fila;

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, int elemento);
int remove_Fila(Fila *fi);
int consulta_Fila(Fila* fi, int elemento);
#endif // LISTA_H_INCLUDED

```

2.Print do funcionamento:

2.1 Parte 1:

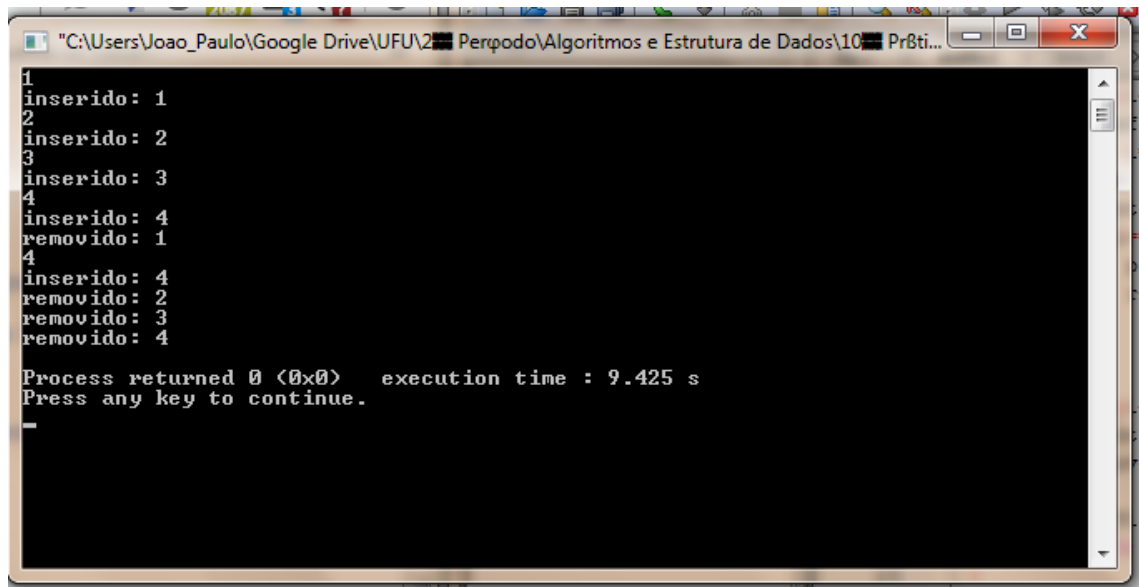


```

"C:\Users\Joao_Paulo\Google Drive\UFU\2º Período\Algoritmos e Estrutura de Dados\10º Prática..."
Enfileirou: 1
Enfileirou: 5
Enfileirou: 6
Enfileirou: 4
Enfileirou: 8
Enfileirou: 2
Enfileirou: 7
Enfileirou: 9
Enfileirou: 10
Desenfileirou: 1
Desenfileirou: 5
Desenfileirou: 6
Desenfileirou: 4
Desenfileirou: 8
Desenfileirou: 2
Desenfileirou: 7
Desenfileirou: 9
Desenfileirou: 10
Process returned 0 (0x0)   execution time : 0.013 s
Press any key to continue.

```

2.2 Parte 2:



A screenshot of a Windows command prompt window. The title bar shows the file path: "C:\Users\Joao_Paulo\Google Drive\UFU\2 Peripodo\Algoritmos e Estrutura de Dados\10 PrBti...". The window contains the following text:

```
1
inserido: 1
2
inserido: 2
3
inserido: 3
4
inserido: 4
removido: 1
4
inserido: 4
removido: 2
removido: 3
removido: 4

Process returned 0 (0x0)   execution time : 9.425 s
Press any key to continue.
```

The text is displayed in a monospaced font on a black background. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.