

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 --- */
} Tipoltem;

typedef struct {
    Tipoltem Item[MAXTAM];
    TipoApontador Topo;
} TipoPilha;

void FpVazia(TipoPilha *Pilha)
{ Pilha -> Topo = 0; }

int Vazia(TipoPilha Pilha)
{ return (Pilha.Topo == 0); }

void Empilha(Tipoltem x, TipoPilha *Pilha)
{ if (Pilha -> Topo == MAXTAM)
    printf(" Erro pilha esta cheia\n");
  else { Pilha->Topo++; Pilha->Item[Pilha->Topo - 1] = x; }
}

void Desempilha(TipoPilha *Pilha, Tipoltem *Item)
{ if (Vazia(*Pilha))
    printf(" Erro pilha esta vazia\n");
  else { *Item = Pilha->Item[Pilha->Topo - 1]; Pilha->Topo--; }
}

int Tamanho(TipoPilha Pilha)
{ return (Pilha.Topo); }

int main(int argc, char *argv[])
{ struct timeval t;
  TipoPilha pilha;
  Tipoltem item;
  int vetor[10];
```

```

int i, j, k, n, max;
gettimeofday(&t, NULL);
srand((unsigned int)t.tv_usec);

max = 4;
FPVazia(&pilha);

/*Gera uma permutacao aleatoria de chaves entre 1 e max*/
for(i = 0; i < 10; 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;
}
/*Insere cada chave na lista */
for (i = 0; i < max; i++)
{ item.Chave = vetor[i];
  Empilha(item, &pilha);
  printf("Empilhou: %d \n", item.Chave);
}
printf("Tamanho da pilha: %d\n", Tamanho(pilha));

/*Desempilha cada chave */
for(i = 0; i < max; i++)
{ Desempilha(&pilha, &item);
  printf ("Desempilhou: %d\n", item.Chave);
}
printf ("Tamanho da pilha: %d \n", Tamanho(pilha));
return(0);
} /* pilha */

```

## 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 3

```

```

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

```

```

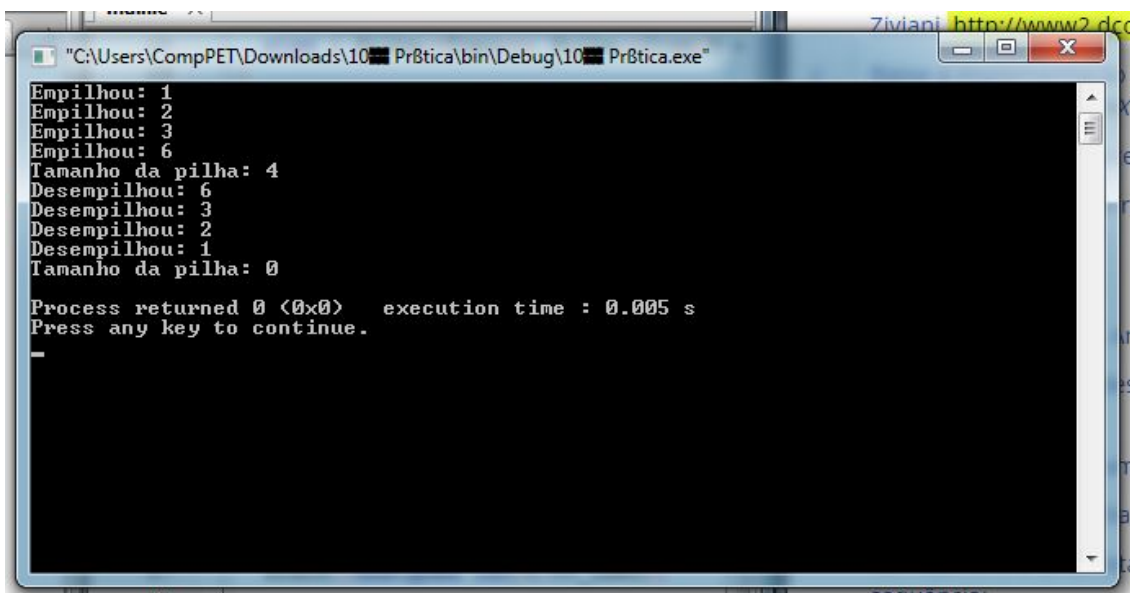
Fila* cria_Fila();
void libera_Fila(Fila* fi);

```

```
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
```

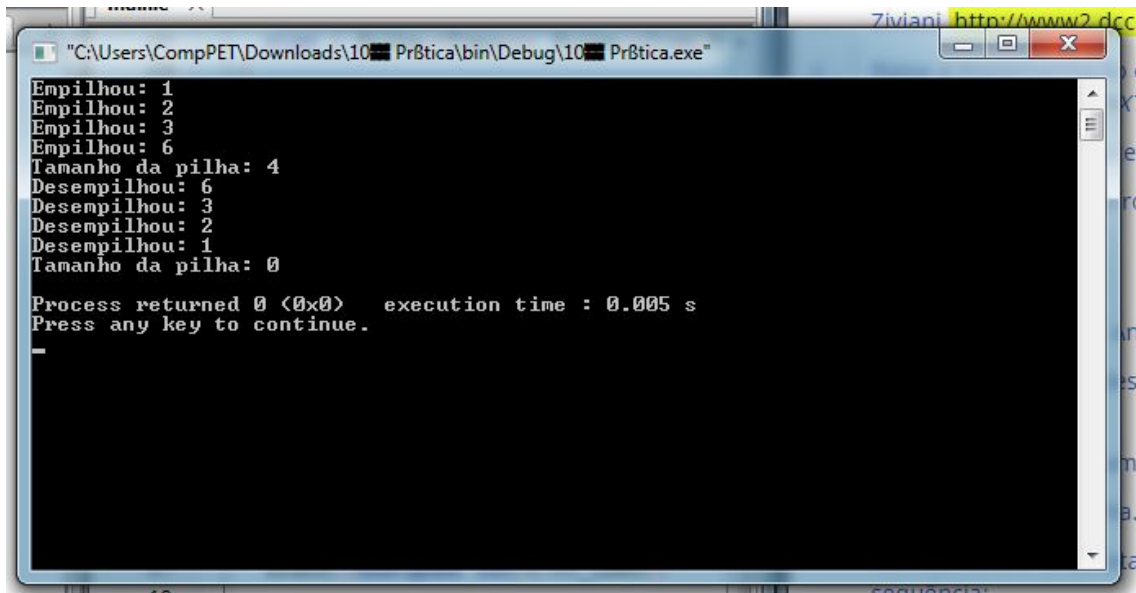
## 2.Print do funcionamento:

### 2.1 Parte 1:



```
"C:\Users\CompPET\Downloads\10 Pratica\bin\Debug\10 Pratica.exe"  
Empilhou: 1  
Empilhou: 2  
Empilhou: 3  
Empilhou: 6  
Tamanho da pilha: 4  
Desempilhou: 6  
Desempilhou: 3  
Desempilhou: 2  
Desempilhou: 1  
Tamanho da pilha: 0  
Process returned 0 (0x0)   execution time : 0.005 s  
Press any key to continue.  
-
```

## 2.2 Parte 2:



```
"C:\Users\CompPET\Downloads\10 PrBtica\bin\Debug\10 PrBtica.exe"
Empilhou: 1
Empilhou: 2
Empilhou: 3
Empilhou: 6
Tamanho da pilha: 4
Desempilhou: 6
Desempilhou: 3
Desempilhou: 2
Desempilhou: 1
Tamanho da pilha: 0
Process returned 0 (0x0)   execution time : 0.005 s
Press any key to continue.
-
```