

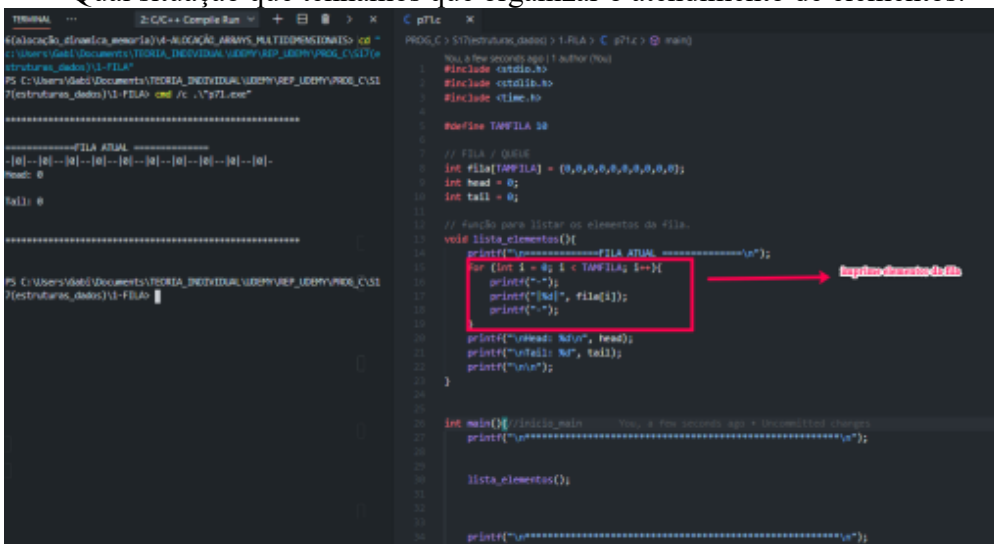
# 1-FILA

FILA = QUEUE

- Estrutura de dados que se parece com uma fila de banco.
  - Sempre que um elemento é adicionado na fila, ele ira para ao final da fila.
  - Sempre que removemos um elemento, o primeiro é removido.
  - As filas trabalham com um recurso da computação chamado FIFO = FIRST IN / FIRST OUT.
- enqueue() // adiciona o elemento na fila  
dequeue() // remove o elemento da fila  
clear() // limpa a fila
- A ESTRUTURA da FILA possui 3 elementos principais:  
FILA[10] - [0][1][2][3][4][5][6][7][8][9]  
HEAD - Cabeça da fila, indica o primeiro elemento da fila.  
TAIL - Calda da fila, indica quantos elementos tem na fila.

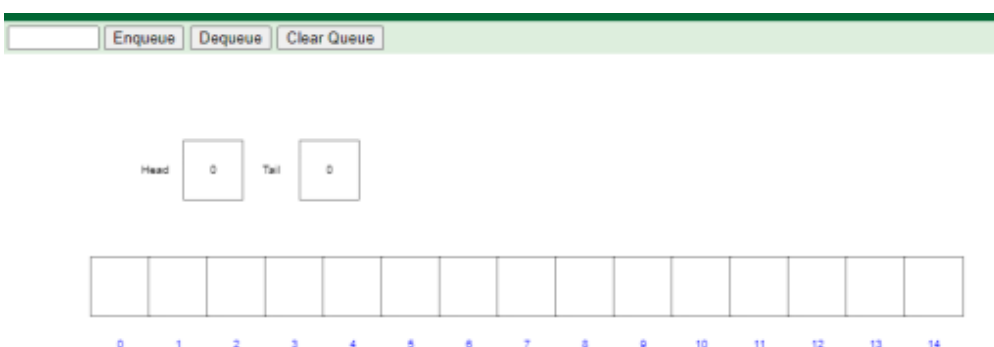
## - APLICAÇÃO

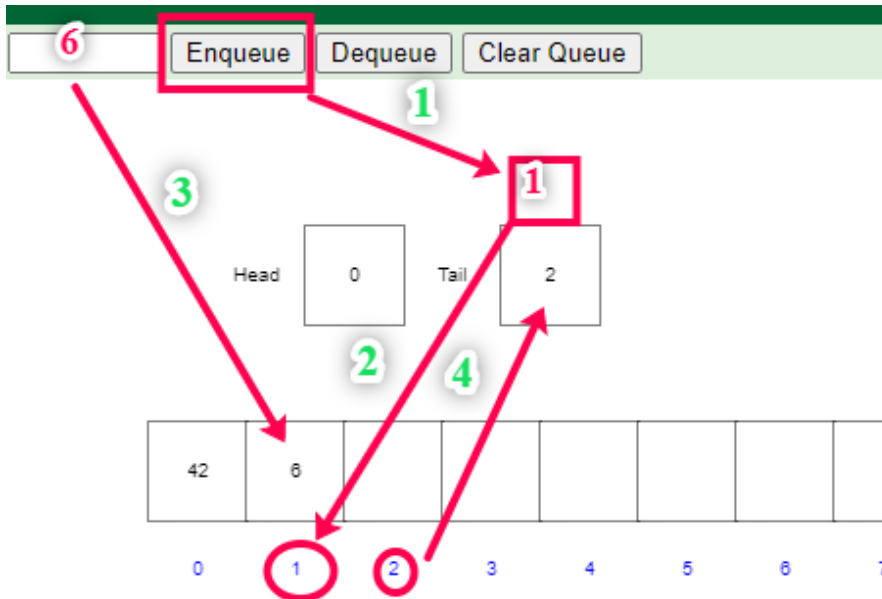
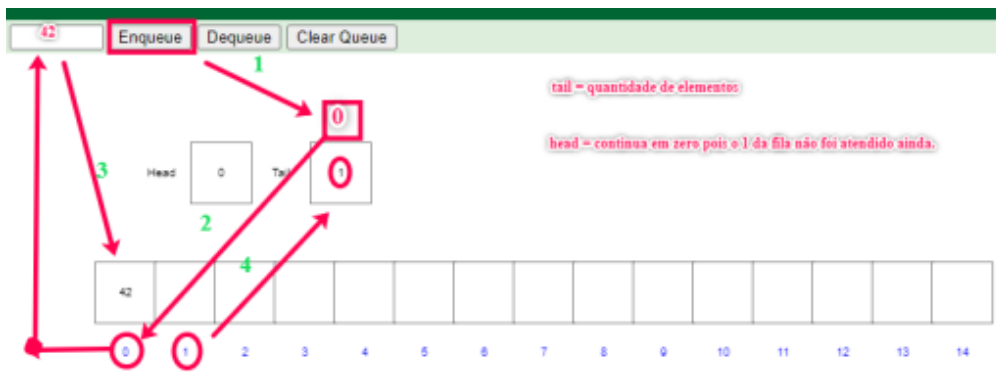
- Qual situação que tenhamos que organizar o atendimento de elementos.



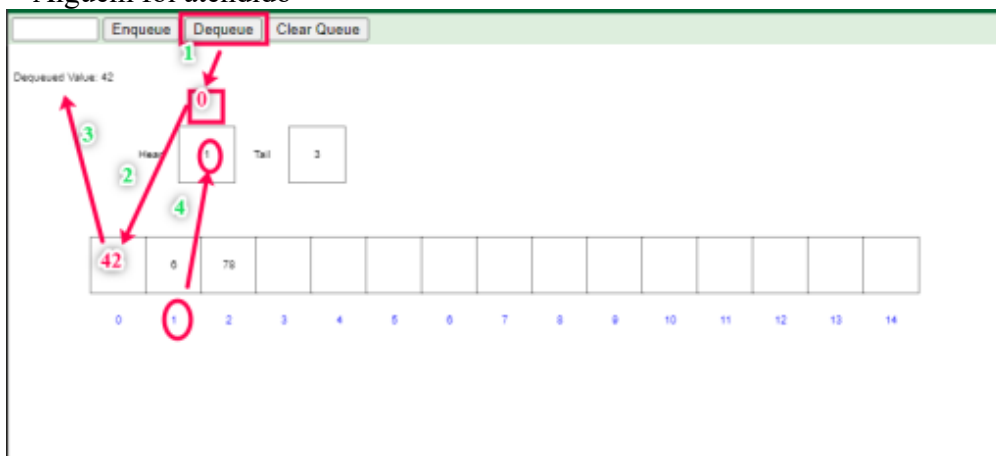
```
PROG.C: S:\Estrutura_dados\1-FILA > C g++ > main)
1 // No, a few seconds ago: 1 author (You)
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <time.h>
5
6 #define TAMFILA 10
7
8 // FILA / QUEUE
9 int fila[TAMFILA] = {0,0,0,0,0,0,0,0,0,0};
10 int head = 0;
11 int tail = 0;
12
13 // Função para listar os elementos da fila.
14 void lista_elementos()
15 {
16     printf("=====FILA ATUAL =====\n");
17     for (int i = 0; i < TAMFILA; i++)
18     {
19         printf("%d", fila[i]);
20         printf("\n");
21     }
22     printf("=====FILA ATUAL =====\n");
23     printf("head: %d", head);
24     printf("tail: %d", tail);
25     printf("\n");
26 }
27
28 int main()
29 {
30     printf("=====FILA ATUAL =====\n");
31     lista_elementos();
32     printf("=====FILA ATUAL =====\n");
33 }
```

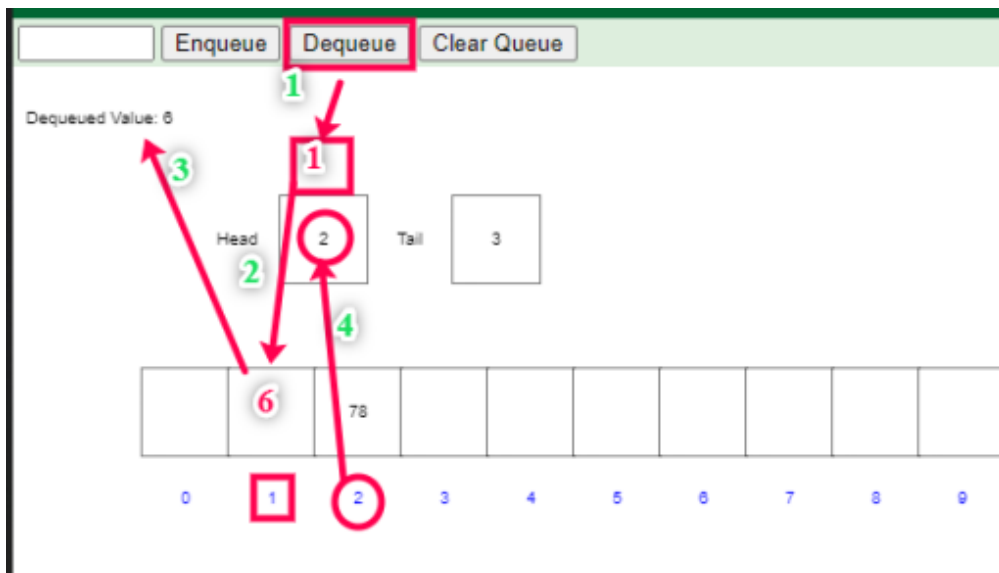
[filas](#) - Site para visualizar a implementação de uma fila.



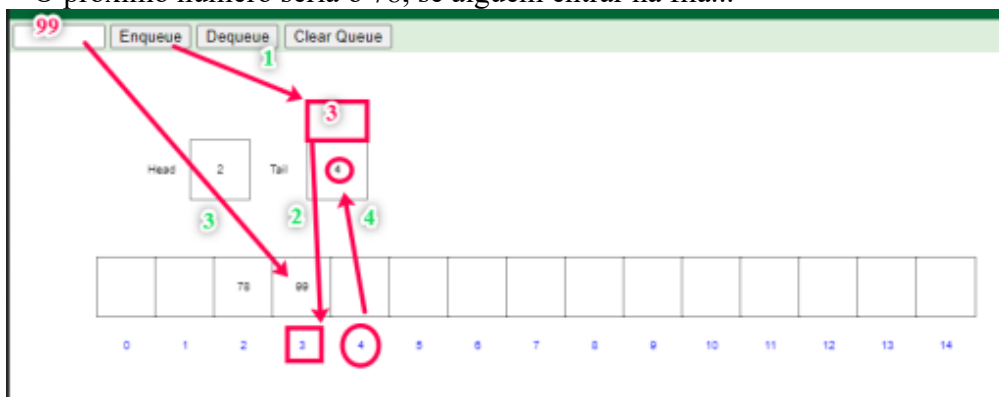


- Alguem foi atendido





- O proximo numero seria o 78, se alguem entrar na fila...



## FUNÇÕES DA FILA

```
#define TAMFILA 10
```

```
// FILA / QUEUE
```

```
int fila[TAMFILA] = {0,0,0,0,0,0,0,0,0,0};
```

```
int head = 0;//proximo a ser atendido
```

```
int tail = 0;// ultimo da fila
```

```
// função para listar os elementos da fila.
```

```
void lista_elementos(){
    printf("\n=====FILA ATUAL =====\n");
    for (int i = 0; i < TAMFILA; i++){
        printf("-");
        printf("|%d|", fila[i]);
        printf("-");
    }
    printf("\nHead: %d\n", head);
    printf("\nTail: %d", tail);
    printf("\n\n");
}
```

```
//Função para adicionar da fila.
```

```
void enqueue(){
```

```
    int val;
```

```

    if(tail < TAMFILA){//verificando se a fila esta cheia
        printf("Informe o elemento para adicionar na fila:\n");
        scanf("%d", &val);
        fila[tail] = val;
        tail++;
        lista_elementos();
    }else{
        printf("A fila esta cheia!\n")
    }
}

// Função para remover da fila..
void dequeue{//temos que fazer uma verificação aqui, ver se o head é menor que o tail
    if(head < tail){
        fila[head] = 0;
        head++;
        lista_elementos();
    }else{
        printf("Fila vazia!\n")
    }
}

//limpar fila clear, tbm temos que zerar o head e o tail se não eles continuam incrementados.
void clear(){
    for (int i = 0; i < TAMFILA; i++){
        fila[i] = 0;
    }
    head = 0;
    tail = 0;
}

```

```
[-1] - Sair:
```

```

int main(){//inicio_main
printf("\n*****\n");

int opcao = 0;

do{
printf("Selecione a opcao:\n");
printf("[1] - Inserir (enqueue):\n");
printf("[2] - Remover (dequeue):\n");
printf("[3] - Listar:\n");
printf("[4] - Limpar a fila:\n");
printf("[ -1] - Sair:\n");
printf("\n\n");
printf("Opcao: ");
scanf("%d", &opcao);

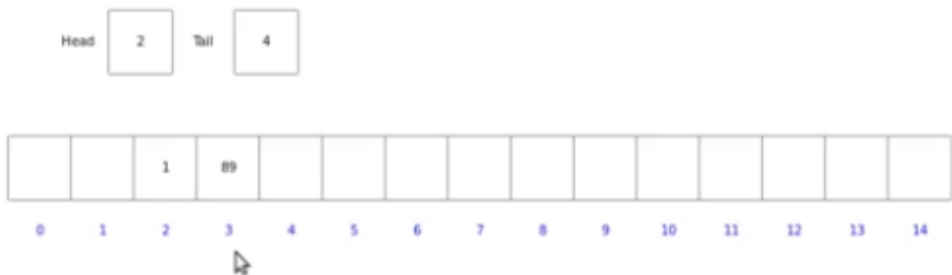
switch (opcao)
{
case 1:
enqueue();
break;
case 2:
dequeue();
break;
case 3:
lista_elementos();
break;
case 4:
clear();
break;

default:
printf("Opcao Invalida!\n");
}

}while(opcao != -1);

```

- Imagine esse caso



- Para essa fila ficar melhor, ou seja, quando chegar no catoze ter espaço ainda, temos que realocar os elementos em (2 e 3)
- Escreva uma função que quando um elemento for removido, os outros passam para frente.