4. Fila Circular com Prioridades em C - Estrutura de Dados

4.1 Conceito

- O conceito continua o mesmo de uma **fila com prioridades**, sendo assim o elemento é inserido no final e deslocado conforme sua prioridade;
- Uma fila circular com prioridades é basicamente a junção das formulações de uma fila com prioridades e uma fila circular;
- As Estruturas de Dados e as Operações Associadas devem ser adaptadas em relação à **fila circular** sem prioridades e o **inserir** deve ser remodelado.

4.2 Exemplos de Aplicação

- Fila de banco no qual existe prioridade dedicada à gestantes e idosos;
- Fila de uma lotérica;
- Supermercado com caixa dedicado à idosos;
- Fórum;
- Cinema;
- Entre outros.

4.3 Estrutura da FILA CIRCULAR COM PRIORIDADES

TAD (Tipo Abstrato de Dados) da FILA CIRCULAR COM PRIORIDADES:

```
#define MAXFILA 10

struct TpElemento{
    char Elemento;
    int Prioridade;
};
struct TpFilaCircPri{
    int inicio, fim, cont;
    TpElemento fila[MAXFILA];
};
```

4.4 Operações Associadas

- void FCPInicializar (TpFilaCircPri &FCP)
- void FCPInserir (TpFilaCircPri &FCP, TpElemento Elemento)
- TpElemento FCPRetirar (TpFilaCircPri &FCP)
- TpElemento FCPElementoInicio (TpFilaCircPri FCP)

- TpElemento FCPElementoFim (TpFilaCircPri FCP)
- char FCPVazia (int cont)
- char FCPCheia (int cont)
- FCPExibir (TpFilaCircPri FCP)

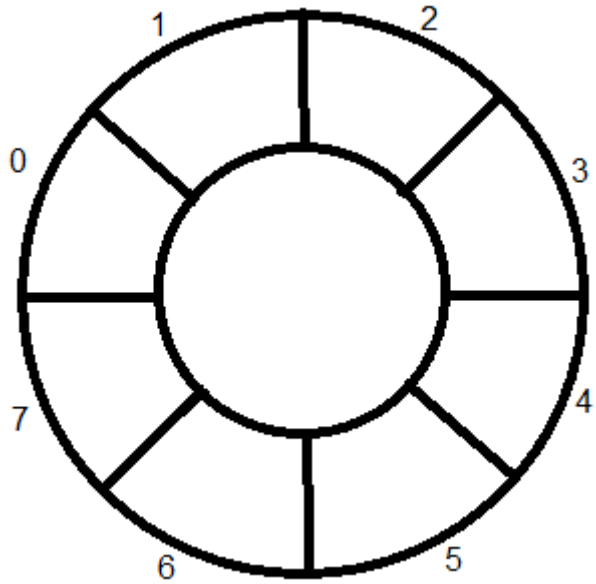
4.5 Implementação em C

```
#define MAXFILA 10

struct TpElemento{
    char Elemento;
    int Prioridade;
};
struct TpFilaCircPri{
    int inicio, fim, cont;
    TpElemento fila[MAXFILA];
};

void FCPInicializar(TpFilaCircPri &FCP){
    FCP.inicio = 0;
    FCP.fim = -1;
    FCP.cont = 0;
}

void FCPInserir(TpFilaCircPri &FCP, TpElemento Elemento){
    TpElemento aux;
    int i;

    if(FCP.fim == MAXFILA -1)
        FCP.fim = -1;

    FCP.fila[++FCP.fim] = Elemento;
    i = FCP.fim;

    while(i>FCP.inicio && FCP.fila[i].Prioridade < FCP.fila[i-1].Prioridade){
        aux = FCP.fila[i];
        FCP.fila[i] = FCP.fila[i-1];
        FCP.fila[i-1] = aux;
        i--;
    }

    FCP.cont ++;
}
TpElemento FCPRetirar (TpFilaCircPri &FCP){
    TpElemento aux;

    aux = FCP.fila[FCP.inicio++];

    if(FCP.inicio == MAXFILA)     // se o inicio for == MAXFILA ele recebe 0,
ele "reseta", volta pro início
        FCP.inicio = 0;

    FCP.cont --;  //decrementa o contador, ja que foi retirado um elemento
```

```
    return aux;
}

TpElemento FCPElementoInicio(TpFilaCircPri FCP){
    return FCP.fila[FCP.inicio];
}
TpElemento FCPElementoFIM(TpFilaCircPri FCP){
    return FCP.fila[FCP.fim];
}

char FCPVazia(int cont){
    return cont == 0;
}
char FCPCheia(int cont){
    return cont == MAXFILA;
}

FCPExibir(TpFilaCircPri FCP){
    TpElemento aux;
    while(!FCPVazia(FCP.cont)){
        aux = FCPRetirar(FCP);
        printf("Elemento: %c - Prioridade: %d\n", aux.Elemento, aux.Prioridade);
    }
}
```

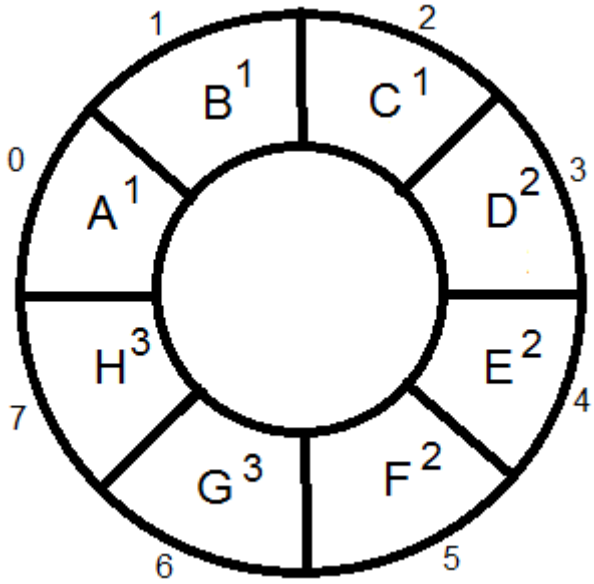
4.6 Funcionamento

1. Inicializar:
    - Início = 0
    - Fim = -1
    - Contador = 0
    - Fila =



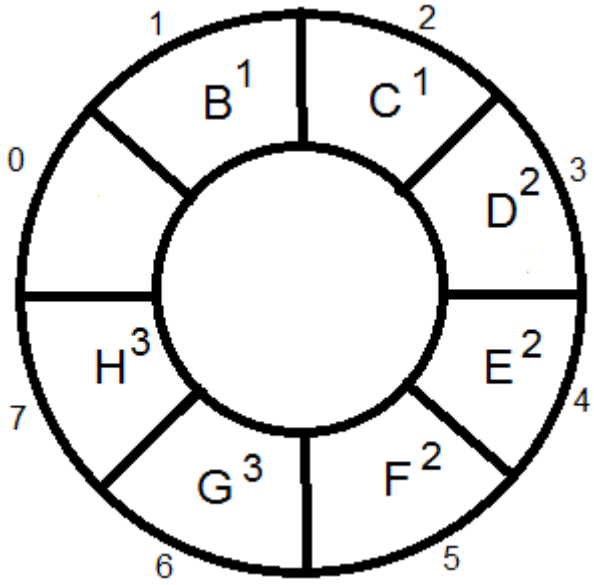
2. Inserir 'A' [1]/ Inserir 'B'[1] / Inserir 'C'[1] / Inserir 'D'[2] / Inserir 'E'[2] / Inserir 'F'[2] / Inserir 'G'[3] / Inserir 'H'[3]:
    - Início = 0
    - Fim = 7
    - Contador = 8

- Fila =



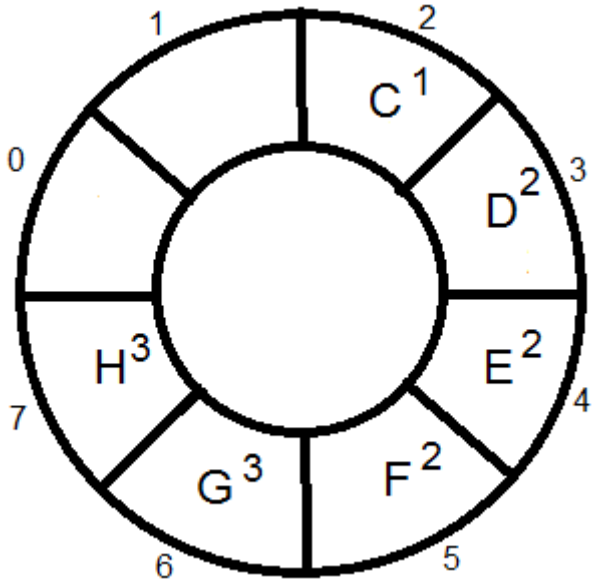
3. Retirar:

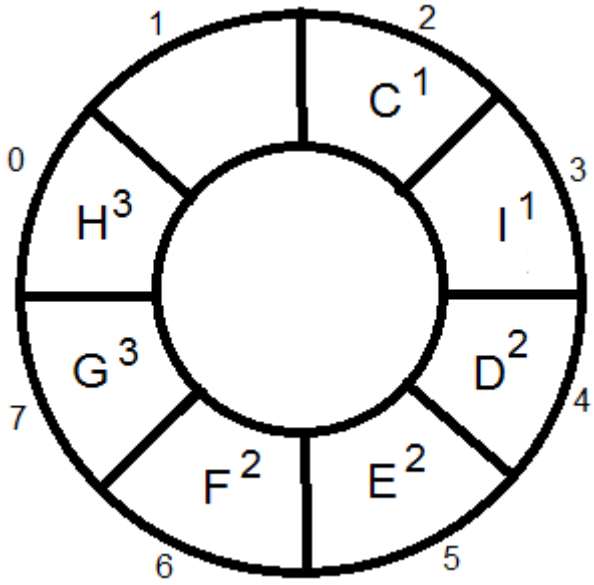
- Início = 1
- Fim = 7
- Contador = 7
- Fila =



4. Retirar:

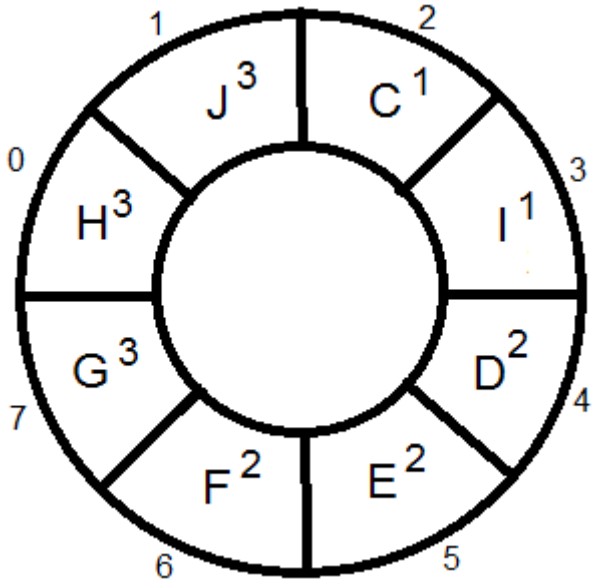
- Início = 2
- Fim = 7
- Contador = 6

- Fila = 

5. Inserir 'I'¹:
   - Início = 2
   - Fim = 0
   - Contador = 7
   - Fila = 

6. Inserir 'J'³:
   - Início = 2
   - Fim = 1
   - Contador = 8

- Fila =



7. Retirar:
    - Início = 3
    - Fim = 1
    - Contador = 7
    - Fila =