

# **Barbeiro Dorminhoco**



## **Problema existe:**

- ✓ 1 Barbeiro;
  - ✓ 1 Cadeira de barbeiro;
  - ✓ N cadeiras de espera;
- 
- Se não houver clientes o barbeiro senta em sua cadeira e dorme;
  - Quando o cliente chega:
    - ✓ Ele acorda o barbeiro, caso esteja a dormir;
    - ✓ Se o barbeiro está trabalhando, o cliente senta para esperar ou sai da barbearia, dependendo das vagas nas cadeiras de espera.

# Implementação

```
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

```
#define N_CLIENTES 10
#define N_CADEIRAS 5
```

```
sem_t sem_cadeiras; // semaforo para cadeira ocupada
sem_t sem_cad_barbeiro; // semaforo para cadeira do barbeiro
sem_t sem_cabelo_cortado; // semaforo barbeiro ocupado ou nao
sem_t sem_cliente_cadeira; // semaforo para se existem clientes
```

**// criar o barbeiro**

```
void * f_barbeiro( void * v){
    while (1){

        sem_wait(&sem_cliente_cadeira);
        printf("barbeiro cortou o cabelo de um cliente");
        sem_post(&sem_cabelo_cortado);
    }
    return NULL;
}
```

**void \* f\_cliente( void \* v){ // procedimento que diz se o barbeiro está dormindo ou atendendo**

```
    int id = * (int *) v;
    sleep (id%3);
```

```

if(sem_trywait(&sem_cadeiras)==0){

    printf("cliente %d entrou na barbearia.\n", id);
    sem_wait(&sem_cad_barbeiro); //aguarda a cadeira do barbeiro ser liberado
    printf("cliente %d sentou na cadeira do barbeiro.\n", id);
    sem_post(&sem_cliente_cadeira); // cliente levanta da cadeira de espera
    sem_post(&sem_cadeiras); // libera cadeira de espera
    sem_wait(&sem_cabelo_cortado); // cliente está cortando o cabelo
    sem_post(&sem_cad_barbeiro); // barbeiro está liberado para dormir ou
atender
    printf("cliente %d deixou a barbearia.\n", id);

}
else {
    printf("cliente % não entrou na barbearia.\n", id);
    return NULL;
}
}

```

```

int main (){
    pthread_t thr_clientes[N_CLIENTES]; // vetor de threads indicando as cadeiras de
espera
    pthread_t thr_barbeiro; // thread do barbeiro
    int i; //contador auxiliar
    int id[N_CLIENTES]; //vetor de identificação de cada clientes

```

**// inicialização dos semaforos**

```

sem_init(&sem_cadeiras, 0,5);
sem_init(&sem_cad_barbeiro,0,1);
sem_init(&sem_cliente_cadeira, 0,0);
sem_init(&sem_cabelo_cortado, 0,0);

```

**//criar threads clientes**

```
for(i = 0; i<N_CLIENTES; i++){  
    id[i] = i;  
    pthread_create(&thr_clientes[i],NULL,f_cliente,(void *)&id[i]);  
}
```

**// criar thread barbeiro**

```
pthread_create(&thr_barbeiro,NULL,f_barbeiro,NULL);  
//chamada para as threads clientes  
for(i=0; i<N_CLIENTES; i++){  
    pthread_join(thr_clientes[i],NULL);  
  
    }  
  
    return 0;  
  
}
```