5] sem_t m(1), full(0), empty(0),
int caldero = 0;

Salvajes

orden entre los salvajes

```
while ( 1 ) }
    sem_wait(&m);
    if(caldero == 0)}
        sem_post(&empty);
        sem_wait(&full);
    {
    getServingsFromPot();
    sem_post(&m);
    eat();
{
```

Cocinero

```
while(1) }
    sem_wait(&empty);
    putServingsInPot(M);
    sem_post(&full);
{
```

```
(6) int surtidor[2]={0,0}; //0:libre, 1: ocupado
    mutex_t m;
    cond_t acceso;
    int ticket = 0;
    int turno = 0;
    void cliente (int dinero)
    {
            int myticket, mysurtidor;

            lock(&m);                                   post.incr.
            myticket = ticket++;
            while ((myticket != turno) || (surtidor[0] && surtidor[1]))
                    cond_wait(&acceso, &m);

 Secc       if (surtidor[0])
 Critica            mysurtidor = 1;                     coger el surtidor
            else                                               libre
                    mysurtidor = 0;
            surtidor[mysurtidor] = 1;
            turno++;
            cond_broadcast(&acceso);
            unlock(&m);
            ServirCombustible(mysurtidor, dinero);      fuera de sección
                                                        critica
                                                        ↳ Concurrencia
             lock(&m);
 Secc.       surtidor[mysurtidor] = 0;
 Critica     cond_broadcast(&acceso);
             unlock(&m);

    }
    int surtidor[2] = }0,0{;
    sem_t m(1), surlibre(2);

void cliente(int dinero)
{       int s;
        sem_wait(&surlibre);
        sem_wait(&m);
         if (surtidor[0])
               s = 1;
         else
               s = 0;
         surtidor[s] = 1
         sem_post(&m);
         servirCombustible(s, dinero);
         sem_wait(&m);
         surtidor[s] = 0;
         sem_post(&surlibre);
         sem_post(&m);
}
```

Secc
Critica