Memoria Fumadores
Práctica 1

Jesús Manuel García Palma

DESCRIPCIÓN DE VARIABLES Y SEMÁFOROS

En la parte superior del .cpp tenemos la creación de los semaforos: sem_t fuma1,fuma2 y fuma3 que serán los semáforos de los respectivos fumadores y sem_t estanco, que será el semáforo del estanquero. Después tenemos una variable booleana que declaramos con el valor true. ¿Para qué? A la hora de llamar a la función de cada fumador, la primera vez que la ejecutemos el fumador fumará, pero si se repite un ingrediente al llamar por segunda vez a un fumador, el programa se queda pillado, por eso utilizamos un while al principio de cada función de fumador, para que cada vez que se llama al fumador para que fume, este lo imprima en pantalla sin ningún problema.

En la función fumar he añadido una variable static booleana para saber si es la primera vez que se ejecuta, en caso de serlo, el programa llama a srand(time(NULL)), posteriormente calcula un numero aleatorio y el retraso bloqueado durante milisegundos.

En estanquero creamos una variable inicializada por defecto a 1. Después creamos un for que se ejecutará infinitamente, para eso hemos creado variable, porque para acabar el for, i tiene que ser igual que variable, y esta va también aumentando a la vez.

Una vez dentro del for creamos una variable r_number,cuyo valor será random entre 1 y 3 en cada iteración del for, así en cada iteración el for utiliza al azar con esta variable un producto. 1 corresponde a tabaco, 2 a papel y tres a cerillas. Llamamos a sem_wait(&estanco) para comprobar si es posible continuar.Si r_number es 1, desbloqueamos el semáforo del fumador dos para que este se ejecute, una vez dentro del fumador dos, llamamos a sem_wait(&fuma2) para que nos diga si es posible fumar, nos da el visto bueno, y a la vez que se ejecuta fumar, indicando que el fumador 2 fuma, se llama a sem_post(&estanco) para desbloquear el semáforo del estanquero y que este pueda seguir creando productos. Al final del bucle, aumentamos variable y así el bucle se hará infinitas veces, hasta que el usuario pulse ctrl+c.
El resto de los fumadores se utilizan de forma similar al explicado, únicamente se cambian los semáforos.

En el main es similar a lo realizado en productor-consumidor: creamos las hebras de los fumadores y del estanquero con pthread_t, inicializamos los semáforos con sem_init, ejecutamos las hebras con pthread_create, finalizamos proceso con pthread_join(es,NUll) y por último destruimos los semáforos con sem_destroy.

CÓDIGO

```
#include <iostream>
#include <cassert>
#include <pthread.h>
#include <semaphore.h>
#include <time.h>      // incluye "time(....)"
#include <unistd.h>    // incluye "usleep(...)"
#include <stdlib.h>    // incluye "rand(...)" y "srand"

using namespace std;
// ---------------------------------------------------------------------------
//semaforos
sem_t fuma1,fuma2,fuma3,estanco;
bool esCierto=true;
// ---------------------------------------------------------------------------
void fumar() // función que simula la acción de fumar  como un retardo aleatorio de la hebra
{
static bool primera_vez = true ;
if ( primera_vez )
{   primera_vez = false ;
srand( time(NULL) );
}
// calcular un numero aleatorio de milisegundos (entre 1/10 y 2 segundos)
const unsigned miliseg = 100U + (rand() % 1900U) ;

// retraso bloqueado durante 'miliseg' milisegundos
usleep( 1000U*miliseg );
}
// ---------------------------------------------------------------------------
void * estanquero(void* p)
{
int variable=1;
for(int i=0;i<variable;i++){
int r_number=0;
sem_wait(&estanco);
r_number= rand() % 3 + 1; //1 == tabaco , 2 == papel, 3 == cerillas
if(r_number == 1){
cout << "el estanquero produce tabaco\n";
sem_post(&fuma2);
}else if(r_number == 2){
cout << "el estanquero produce papel\n";
sem_post(&fuma3);
}else if(r_number == 3){
cout << "el estanquero produce cerillas\n";
sem_post(&fuma1);
}
variable++;
}
}
// ---------------------------------------------------------------------------
```

```
void * fumador1(void* p)
 {          while(esCierto){
          sem_wait(&fuma1);
 cout << "El fumador 1 fuma\n";
                fumar();
          sem_post(&estanco);
                }
                }
// -------------------------------------------------------------------------
                void * fumador2(void* p)
 {          while(esCierto){
          sem_wait(&fuma2);
 cout << "El fumador 2 fuma\n";
                fumar();
          sem_post(&estanco);
                }
                }
// -------------------------------------------------------------------------
                void * fumador3(void* p)
 {          while(esCierto){
          sem_wait(&fuma3);
 cout << "El fumador 3 fuma\n";
                fumar();
          sem_post(&estanco);
                }
                }
// -------------------------------------------------------------------------
                        int main()
                        {
 srand( time(NULL) );
   pthread_t f1,f2,f3,es;

  sem_init(&fuma1,0,0);
  sem_init(&fuma2,0,0);
  sem_init(&fuma3,0,0);
 sem_init(&estanco,0,1);

pthread_create(&es,NULL,estanquero,NULL);
 pthread_create(&f1,NULL,fumador1,NULL);
 pthread_create(&f2,NULL,fumador2,NULL);
 pthread_create(&f3,NULL,fumador3,NULL);

  pthread_join(es,NULL);


 sem_destroy(&estanco);
  sem_destroy(&fuma1);
  sem_destroy(&fuma2);
  sem_destroy(&fuma3);

     cout << "FIN\n";
                }
```