UNIVERSIDAD DEL BÍO-BÍO

EJERCICIO DE LABORATORIO 5

Sistemas Operativos – Otoño 2018

Profesor Luis Gajardo

ICI

1. Objetivo

Implementar la solución del productor-consumidor para un proceso productor y uno consumidor utilizando la librería nSystem.

2. Ejercicio de Laboratorio

Considerando la siguiente solución al problema del productor-consumidor, implemente una solución con el mecanismo de sincronización de semáforos que permita mostrar por pantalla el contenido del buffer a medida que cambia.

Esta solución es válida solo para un productor y consumidor simultáneo.

```
#define N 10
Item buff[N];
int e=0, f=0;

Sem full= makeSem(0);
Sem empty= makeSem(N);

void put(Item it) {
  wait(empty);
  buff[e]= it;
  e= (e+1)%N;
  signal(full);
}
```

```
Item get() {
  Item it;
  wait(full);
  it= buff[f];
  f= (f+1)%N;
  signal(empty);
  return it;
}
```

Debe implementar la función nMain(), lanzar las tareas, e imprimir las palabras “prod” y “cons” cuando de invoca alguna de las dos funciones, además debe imprimir el contenido del buffer después de que cada procesos inserte o extraiga datos. El buffer representa una celda vacía con cero ‘0’ y una con datos con ‘p’.

A continuación se muestra un ejemplo de salida que usted debe respetar (sin los colores) (los números no son necesarios):

```
1.   prod 0000000000
2.   cons p000000000
3.   prod 0000000000
4.   prod 0p00000000
5.   prod 0pp0000000
6.   prod 0ppp000000
7.   prod 0pppp00000
8.   prod 0ppppp0000
9.   prod 0pppppp000
10.  prod 0ppppppp00
11.  prod 0pppppppp0
12.  prod 0ppppppppp
13.  cons pppppppppp
14.  prod p0pppppppp
```

La API de nSystem para usar semáforos es la siguiente

- ***nSem nMakeSem(int n);***
  Crea un semáforo de n tickets (si n=1, el semáforo es binario)
- ***void nWaitSem(nSem s);***
  Solicita un ticket al semáforo (sino hay disponible la tarea debe esperar)
- ***void nSignalSem(nSem s);***
  Aporta un ticket al semáforo y la tarea continúa su ejecución
- ***void nDestroySem(nSem s);***
  Destruye el semáforo

15. cons ppppppppp