SISTEMAS CONCURRENTES Y DISTRIBUIDOS
Examen práctica 1 (28-10-2016)

Complete **EN PRIMER LUGAR todos los datos** que se piden a continuación:

APELLIDOS: _____
NOMBRE: _____ e-mail : _____

Lea atentamente las siguientes NORMAS DE ESTE EXAMEN:

- *En la identificación inicial se debe incluir el código:* **ubu14es**
- *Si hay copia entre exámenes se anularán los dos exámenes implicados.*
- *Tiempo para este examen: 75 minutos.*
- *Los ejercicios se deberán subir a la plataforma prado en el trabajo del examen*

*1.- (5 ptos.) Modificar el problema de los fumadores de la siguiente manera (el archivo se debe llamar "**fuma_ex.cpp**"):*

- *Se debe crear una nueva hebra estanquero, que se alternará con la otra hebra estanquero para el ciclo normal de producir ingrediente y esperar a ser recogido por los fumadores. En le hebra main se decidirá de forma aleatoria cuál es la hebra estanquero que comienza poniendo el primer ingrediente.*
- *Cada vez que un estanquero genere el mismo ingrediente que generó en su ciclo anterior, se deberá mostrar el siguiente mensaje por pantalla indicándolo: "El estanquero N ha producido dos veces seguidas el mismo ingrediente: INGR", sustituyendo N por el número de estanquero (1 ó 2) e INGR por el ingrediente generado.*

2.- (5 ptos.) Tenemos tres procesos concurrentes que comparten la variables enteras **x** e **y**. Los procesos *P1 y P2 van recorriendo cada uno un vector de tipo entero de 9 elementos (***v1** y **v2**), cuyos valores son *escogidos de forma aleatoria entre los valores 1 y 7 (* rand()%7+1 *). El proceso P2 debe esperar al cálculo de* **x** *por parte de P1 para calcular* **y**. *Cuando está calculado el valor de* **y**, *si dicho valor es par, el proceso P3 actualiza* **z** *y si es impar no se actualiza* **z**.

Nota: *Como no se puede saber a priori cuántos valores pares o impares va a calcular el proceso P2, no se puede saber cuantas iteraciones debe hacer el proceso P3. Dicho proceso debe terminar cuando se hayan generado todos los datos (no se puede utilizar la función* *pthread_cancel*). *A continuación se muestra el seudocódigo:*

variables globales: *v1[9], v2[9]* x, y, z: integer

```
P1
for i =0 to 8 do
   x := v1[i] ;
   <liberar proceso P2>
   <esperar a que x sea leida>
```

```
P2
for i =0 to 8 do
   <esperar valor x>
   y:= v2[i] + x ;
   <liberar proceso P1>
   if (y es par)
      then <liberar proceso P3>
           <esperar a que y sea leida>
```

```
P3
z:= 0
<mientras haya valores repetir>
   <esperar valor y>
   z:= z + y
   <liberar proceso P2>
```

*Construir un programa con tres hebras que cumpla esos requisitos (el archivo se debe llamar "**ejercicio2.cpp**"). La hebra main debe inicializar al principio los dos vectores y al final debe imprimir el resultado de la variable* **z**.