**PRÁCTICO DE HILOS**

**INF 323 SISTEMAS OPERATIVOS I. GESTIÓN 1/2013**

**1.** Mediante un grafo de planificación, verifique que en el siguiente esquema, con una variable de cierre c, se produce una condición de concurso, tomando en cuenta que en cada quantum asignado a un proceso se ejecutan **al menos dos líneas** de código (no tome en cuenta en su grafo a la línea “*while (true)*”)

|  |  |
| --- | --- |
| int **c**=0; //Variable global.  **Proceso P0**  while (true){  ➀ if (c==0){  ➁ c=1;  ➂ SC();  ➃ c=0;  }  } | **Proceso P1**  while (true){  ➄ if (c==0){  ➅ c=1;  ➆ SC();  ➇ c=0;  }  } |

**2.** Usando un Grafo de Planificación, muestre que en el siguiente esquema se genera una condición de concurso:

|  |  |
| --- | --- |
| int Turno=0, C=0; //Variables globales.  **Proceso P0**  while(true){ if (C==0)**{**  C=1;  if (Turno==0)  SC();  C=1-C;  Turno = C;  **}**  } | **Proceso P1**  while(true){  if (C==0)**{**  C=1;  if (Turno==1)  SC();  C=1-C;  Turno=1-C;  **}**  } |

*En su Grafo,* ***no*** *tome en cuenta a las líneas de los “while´s”.*

**3.** Se tiene un vector global de enteros V[ ] y un entero X. Con la ayuda de 2 hilos h y t busque la existencia de X en V[ ] **alternadamente**. Es decir, h busca X en V[0], t busca X en V[1], h busca X en V[2], etc. Sin embargo, si algún hilo encuentra X en V[ ], muestra un mensaje y ambos hilos finalizan; pero si X no se encuentra en V[ ] el último hilo en salir del ejecutor mostrará el mensaje en consola “Dato no encontrado” (verifique que éste mensaje solo sea mostrado una vez).

**4.** Se tiene un entero N y una matriz entera M[0..N][0..N], ambos globales. Escriba un programa que con la ayuda de los hilos h[0], h[1], …, h[N] y **sin alternación estricta** calculen la suma de todos los elementos de la matriz, donde el hilo h[k] (cuando le toca su “Turno”) subcalcula la suma de la fila k de M[ ][ ].

El último hilo en salir del ejecutor, mostrará en consola (print) la suma total de la matriz.

**5.** Se tienen los hilos h[0], h[1], ..., h[N] y un vector de enteros V[ ] global. Cada uno de éstos hilos busca un valor entero X en particular en el vector V[ ] (i.e. X es un campo de la class Hilo).

Escriba un programa que lance los hilos y **alternadamente** cada hilo busque su X en V[ ] casilla por casilla (revisa **una** casilla cuando le toca el “Turno”). Si algún hilo encuentra su X, mostrará el mensaje “Gané” y todos los demás hilos finalizarán.

*Hint: Use un campo int i en la class Hilo, para que así cada hilo sepa cual casilla le tocará inspeccionar ­ “if (V[i]==X)…”*

**6.** Rescriba el ejercicio anterior, pero sin alternación estricta. Es decir, los hilos no esperarán un “Turno” para revisar la siguiente casilla de V[ ], sino que correrán libremente como si fuese una competencia por quien encuentra primero su valor X.

**7.** Se tiene un vector global de enteros V[ ] y dos hilos h y t de una misma class Hilo. Sin alternación estricta hacer que los hilos h y t hagan lo siguiente: El hilo h asigna ceros a V[ ] desde su parte izquierda (i.e. V[0]=0, V[1]=0, …), mientras que t asigna unos desde la parte derecha de V[ ] (i.e. V[UltimaCasilla]=1, V[PenúltimaCasilla]=1, …). Un hilo (h o t) finaliza si en la próxima casilla a asignar se encuentra el otro hilo; es decir, los hilos finalizan si se encuentran procesando la misma casilla. (Al final de todo, V[ ] tendrá 0´s en la parte izquierda y 1´s en su parte derecha).

**8.** *(Hilos con un Planificador)* Se tienen globalmente un vector global de enteros V[0..N], una variable entera X, los hilos H y T de la class Hilo y un hilo P de la class Planificador.

Realizar un programa que cuente la cantidad de veces que aparece X en V[ ] de la siguiente manera:

El hilo P toma el control y planifica en Round-Robin a los hilos H y T.

El hilo H busca en la casilla V[0] la existencia de X y luego cede el control a P. P cede el control a T, T verifica la existencia de X en V[1] y cede el control a P…, y así sucesivamente hasta agotar todas las casilla de V[ ].

Cuando esto ocurre el **planificador P**, muestra un mensaje (print) con la cantidad de ocurrencias de X en V[ ].