Autores:

* Nava Benítez David Emilio
* Tavera Castillo David Emmanuel

Problema: Impresoras

Descripción del problema: En el día a día, en diferentes lugares el uso de impresoras es tan común y necesario que es de suma importancia mantener un control de dichas impresoras para poder satisfacer las necesidades de aquellos que la ocupan y que no haya inconformidades por un mal funcionamiento. En estos meses, al estar en el Instituto de Investigaciones Jurídicas (David Nava) y en el servicio social en la Facultad de Ingeniería (David Tavera), se ha visto varios problemas en las dichosas impresoras. Y con un análisis (no tan exhaustivo hay que decirlo), realmente la concurrencia ahí es un problema que te deja canas verdes. Debido a que no se tiene la posibilidad de que cada persona tenga una impresora asignada a ellas, en ciertos espacios una gran cantidad de personas puede mandar a imprimir a una sola impresora, pero no conforme a ello, también pueden escanear y sacar copias. Entonces tenemos tres procesos en que en ciertas ocasiones puede llegar a ser un problema y más cuando todos quieren trabajar sobre la impresora.

Ahora bien, luego tenemos que se tienen 3 o 4 impresoras asignadas cada una a un área específica y que la comparten las personas de esa área, pero sucede (y es lo que pasa actualmente en el IIJ y en la FI) que una impresora se descompone y en lo que el fabricante le da mantenimiento, la impresora (o impresoras) que funcionan se sobrecargan con las personas de la susodicha impresora.

Entonces identificamos varios puntos de concurrencia en este aspecto:

1.- Varias personas requieren imprimir a la vez

2.- Varias personas requieren sacar copias a la vez

3.- Varias personas requieren escanear a la vez

¡Y por si fuera poco! ¡Una combinación de ellas!

Pero aún peor, ¡¿y si una falla?!

Así pues, ¡aquí presentamos la solución!

**Para resumir el programa realizado en C**

Este programa simula un entorno de múltiples usuarios accediendo a un conjunto limitado de impresoras, las cuales pueden presentar fallas y ser reparadas. Se emplean técnicas de programación concurrente utilizando hilos, semáforos y mutex para controlar el acceso compartido a los recursos.

**Mecanismos de sincronización empleados:**

* **Semáforos:** Son los que controlan cuántos usuarios pueden acceder a una impresora al mismo tiempo.
* **Mutex:** Protege las variables compartidas.

**Variables Compartidas:**

* ‘impresoras\_disponibles’: Impresoras funcionales en el sistema.
* ‘usuarios\_activos’: número de usuarios restantes.
* ‘estado\_impresoras[]’: estado de cada impresora que puede ser 1 = operativa o 0 = dañada.
* ‘en\_uso[]’: indica que una impresora esta siendo usada actualmente
* ‘ultima\_impresora\_usada’: Usada para repartir equitativamente el uso de impresoras.

**Flujo de operación:**

1.- Se puede elegir cuantos usuarios participarán (máximo 100)

2.- Se inicializan las 4 impresoras

3.- Se lanzan:

* Un hilo de mantenimiento que simula las fallas en impresoras
* Un hilo de reparación (de impresoras)
* Un hilo por cada usuario participante

4.- Cada usuario espera su turno, accede a una impresora disponible, realiza una acción aleatoria (imprimir, copiar o escanear), y libera la impresora (la deja de usar)

5.- Una vez que todos los usuarios hayan terminado su acción, el programa finaliza.

**Descripción algorítmica del avance de cada hilo**

* **Hilo principal**

Solicita el número de usuarios.

Inicializa semáforo y mutex.

Lanza hilos:

De mantenimiento (descompone impresoras aleatoriamente).

De reparación.

Uno por cada usuario.

Espera a que todos los hilos terminen con *pthread\_join*.

* **Hilos de usuario (tarea)**

Espera turno mediante *sem\_wait(&impresora)*

Busca una impresora disponible con *obtenerImpresoraDisponible()*

Realiza una acción aleatoria

Libera la impresora(*sem\_post*) y disminuye los usuarios con *usuarios\_activos*

Cada hilo de usuario trabaja de forma autónoma y sincronizada

* **Hilo de mantenimiento**

Espera un tiempo aleatorio y descompone una impresora si hay más de una disponible y ninguna descompuesta.

* **Hilo de reparación**

Verifica cada 10 segundos si hay alguna impresora descompuesta.

Si la hay, la repara y aumenta el contador de impresoras disponibles.

**Descripción de la interacción entre hilos**

* Usuarios y semáforo: Solo pueden operar tantas impresoras como haya disponibles (el semáforo controla eso).
* Usuarios y mutex: Al asignar impresoras y modificar variables globales, se requiere protección contra acceso simultáneo a dichas variables globales.
* Mantenimiento y reparación: Actúan como eventos externos que modifican el sistema. Ambos también requieren mutex para modificar los estados de impresoras.
* Todos dependen del mutex para garantizar consistencia al modificar las estructuras compartidas.

Observaciones:

* Las impresoras pueden fallar aleatoriamente.
* El sistema permite observar concurrencia, espera y reparación.
* El comportamiento puede cambiar en cada ejecución debido al uso de números aleatorios.

**Descripción del entorno de desarrollo**

Lenguaje empleado: C

Se usan bibliotecas estándar POSIX.

Bibliotecas: pthread.h,

semaphore.h,

unistd.h,

stdlib.h,

stdio.h

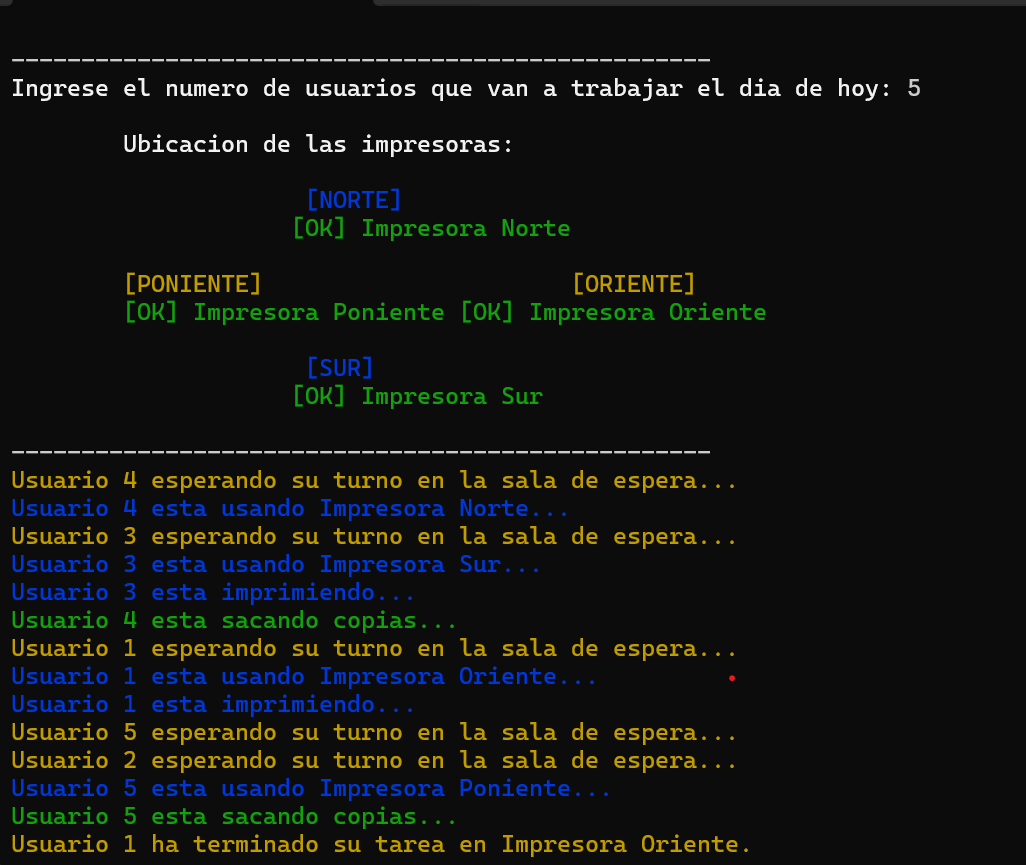
time.h

El proyecto se desarrollo bajo el sistema operativo Windows 11 y también se probó en Windows 10.

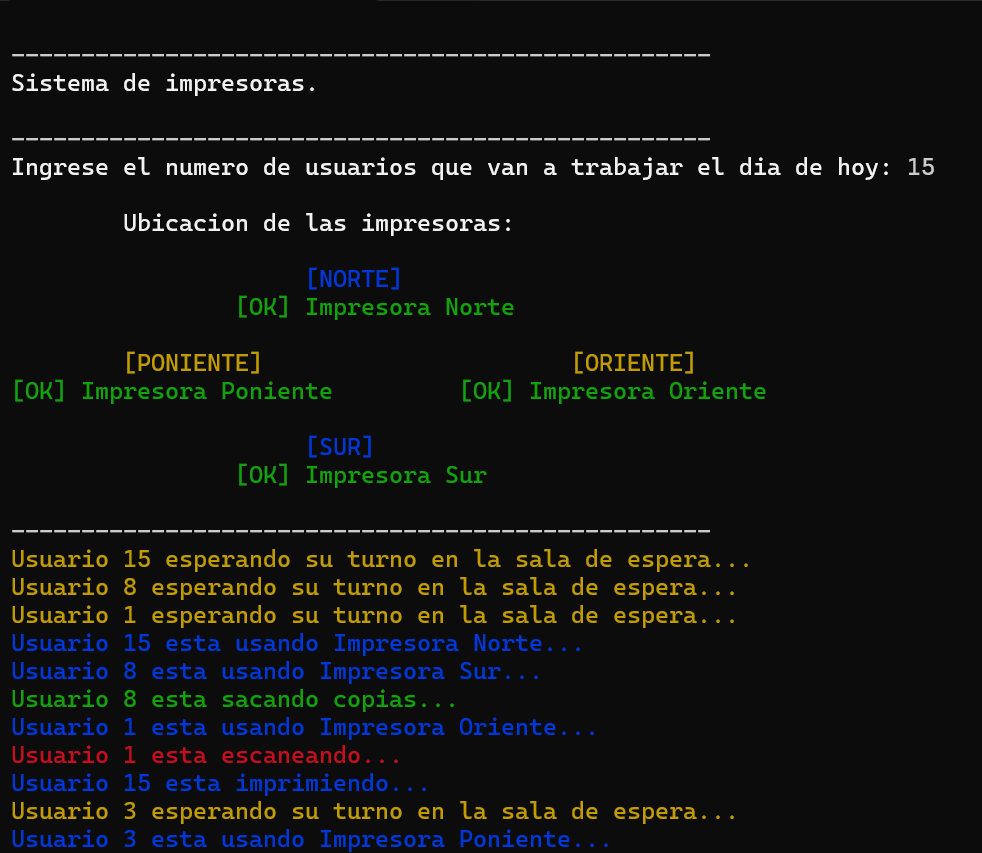
Importante contar con una terminal compatible con secuencias ANSI para colores.

**Ejemplos (pantallazos) de ejecución exitosa**

Con 5 usuarios.



Con 15 usuarios



Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Con 50 usuarios (antes de el mapa de impresoras)

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Texto, Aplicación

El contenido generado por IA puede ser incorrecto.