I.T.C.R. Sistemas Operativos

Escuela de Computación — Laberinto

Prof: Ing. Erika Marín Schumann. — II Semestre del 2021

Objetivos del proyecto

- Desarrollar un programa donde se utilicen threads y forks

Descripción

Este primer proyecto tiene como finalidad desarrollar una aplicación que simule un Laberinto. Cada laberinto que se ejecute tendrá 2 versiones. Una con hilos y otra con foks. En ambas se aplicará la misma lógica y al finalizar se evaluará cual de las 2 dio mejores resultados en cuanto a eficiencia.

La Tarea debe ser programada en C para Linux.

LABERINTO: El laberinto se leerá de un archivo de texto donde se podrá obtener las dimensiones y los campos donde haya paredes.

Es importante, que ya sea cada uno de los hilos/procesos o un hilo aparte muestre en pantalla, aunque sea con caracteres el movimiento que han tenido los hilos/procesos y los espacios por los que ya se ha pasado.

Siempre se saldrá o iniciará de la posición (0, 0). El laberinto se verá así

	*				*	*	*
	*		*				
	*		*		*	*	*
	*		*	*	*	*	*
			*				*
*	*		*		*		
	*		*		*		*
					*	/	*

Los asteriscos significa que son paredes y el / indica la salida del laberinto. En este ejemplo hay una sola salida, sin embargo pudiera haber laberintos con más de una salida.

Utilice cualquier representación del Laberinto en memoria. Lo importante es que lleven control de la dirección en la que se ha recorrido cada espacio. Es Recomendable que el laberinto y su información sean compartidos por los hilos/procesos. En ese caso puede utilizar un semáforo (o algún medio de sincronización) para todo el laberinto. No se complique a sincronizar cada espacio del laberinto.

Ejecución del Laberinto.

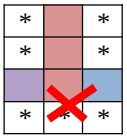
La ejecución del Laberinto aplica igual para la versión de hilos y la versión de procesos.

La idea general es que se cuente con un hilo/proceso inicial que irá "caminando" de campo en campo en el laberinto. Cuando llegue a un espacio donde tenga más de una opción para avanzar creará un nuevo hilo/proceso para cada uno de los otros caminos diferentes a la dirección que lleva el hilo. Al finalizar cada hilo/proceso debe decir cuántos espacios recorrió esa ruta y si tuvo éxito de llegar a la salida del laberinto.

DIRECCION DEL HILO/PROCESO: Es necesario saber la dirección que tienen los hilos/procesos para asegurarnos que no se enciclará la tarea. De manera que un hilo/proceso muere o no puede seguir si ya algún otro hilo/proceso pasó en la misma dirección por ese espacio. Hay cuatro direcciones: Abajo, Arriba, Derecha e Izquierda.

CREACION DE NUEVOS HILOS/PROCESOS: Cada vez que un hilo/proceso llega a un espacio donde tiene más de un camino a seguir creará un nuevo hilo/proceso definiéndole la nueva dirección y "pasándole" la cuenta de espacios que lleva hasta el momento. No se crea un hilo/proceso para devolverse por el camino en que venían.

Hay un caso donde el hilo/proceso puede no seguir su dirección pero tiene otras opciones. En ese caso puede crear los hilos/procesos para las otras direcciones y después finalizar



FINALIZACION DE HILOS/PROCESO: Un Hilo/Proceso finaliza cuando:

1. Llega a la salida
2. Cuando topa con pared en la dirección en la que iba.
3. Cuando el siguiente espacio ya fue recorrido en su misma dirección

Cuando esto ocurre, debe desplegar que finalizo, los espacios recorridos y si fue exitoso.

DESPLIEGUE: Ya sea un hilo aparte o los mismos hilos, deben desplegar en consola, por medio de caracteres los espacios que ya han sido recorridos. Puede utilizar un carácter o color diferente para cada dirección. Si un espacio ha sido recorrido en varias direcciones puede presentarse solo una. Para efectos de presentación, puede hacer que los hilos duren cierto tiempo en cada espacio. De esta manera se verá mejor el despliegue y no se resolverá el problema en milisegundos.

COMPARACION y FINALIZACION DE LA SIMULACION: Una vez cargado el laberinto, la aplicación procederá a ejecutar usando hilos y al finalizar volverá a ejecutar usando procesos fork. La idea es que contabilicen el tiempo que dura cada solución y presenten los dos tiempos al final.

En ambos casos puede usar hilos para el despliegue y para llevar tiempo, si lo consideran necesario.

Documentación

Se espera que sea un documento donde especifique lo siguiente:

a. Portada, índice, introducción
b. Estrategia de Solución
c. Análisis de Resultados: Deberá elaborar un listado de todas y cada una de las actividades y tareas que deben cubrirse a nivel funcional, para cada una de ellas debe aportar el porcentaje de realización y en caso de no ser el 100% debe justificarse.
d. Lecciones aprendidas: Debe prepararse un listado de las lecciones aprendidas producto del desarrollo de la tarea programada. Las lecciones aprendidas pueden ser de carácter personal y/o técnico que involucre aspectos que han logrado un aprendizaje en temas de investigación, desarrollo de habilidades técnicas y habilidades blandas como trabajo en equipo, comunicación, forma de expresar ideas, entre otros.
e. Casos de pruebas: se espera que definan claramente cada prueba, cuáles son los resultados esperados y cuáles fueron los resultados obtenidos. No es necesario que sean grandes pero deben evaluar la funcionalidad completa del programa.
f. Comparación: Según los resultados, la experiencia obtenida y una pequeña investigación(complementaria) debe comparar el trabajar con Threads y con procesos. La comparación debe incluir usabilidad, detalles técnicos y rendimiento.
g. Manual de usuario: especificar como compilar y correr su tarea.
h. Bitácora de trabajo durante las tres semanas de trabajo, incluyendo verificaciones realizadas (si existieran) de consultas realizadas con el profesor o asistente.
i. Bibliografía y fuentes digitales utilizadas

Aspectos Administrativos

- El desarrollo de este programa debe de realizarse en grupos de tres o cuatro personas.
- El trabajo se debe de entregar antes del 17 de setiembre de 2021 a media noche, por medio del TecDigital
- Deben entregar el código fuente junto con el ejecutable.
- Recuerde que los trabajos que se entreguen de forma tardía serán evaluados en base a 60.