UNIVERSIDAD DEL VALLE DE GUATEMALA

Carreras de autos formula 1

Andre Marroquin- 22266
Nelson García Bravatti - 22434
Joaquin Puente - 22296

Programacion de Mircroprocesadores

Guatemala, 2023

Descripción del problema:

Se necesita un programa que ejecute la simulación de carreras de autos. Para ello, al iniciar el programa se le solicitará al usuario que ingrese datos tales como la duración de las llantas, la duración de la gasolina y la cantidad de vueltas totales que tendrá la carrera. La velocidad de cada auto se representa abstractamente como la velocidad a la que cada hilo ejecutará cada iteración del bucle. Como funcionalidades adicionales se tendrán diferentes pistas las cuales definirán un sleep universal para todos los autos representando que la vuelta era más larga. También, se tendrá la funcionalidad de probabilidad de lluvia la cual definirá la probabilidad de accidentes para los autos. Si un auto llega a tener un accidente, se termina la ejecución de ese hilo y se mostrará un mensaje diciendo que este ha sido descalificado. Al final de cada vuelta se mostrará el tiempo que cada auto se hizo en la vuelta y se mostrará al final de la carrera una tabla de resultados mostrando el tiempo que se hizo cada auto. El problema que debemos solucionar es el de hacer la carrera "justa" es decir que todos los autos deben correr al mismo tiempo. No podemos darnos a la tarea de ejecutar cada auto individualmente y luego comprar su tiempo, sino que debemos ejecutar cada auto como un hilo paralelo, para que todos corran al mismo tiempo, cada uno haga su vuelta a su tiempo y cada uno tenga el mismo riesgo de tener un accidente.

Solución Planteada:

1. Descripción de la solución propuesta por los estudiantes:

La solución del problema planteado anteriormente será implementar lo siguiente en un programa, este trabajará con variables que incluyen el tiempo de inicio de cada auto, el tiempo de parada de cada auto y el tiempo de completación de la carrera de cada vehículo. El usuario deberá proporcionar la cantidad de vueltas deseadas, así como el número de vueltas que durarán las llantas y la gasolina de cada auto. El programa realizará cálculos para determinar cuántas vueltas ha dado cada auto, cuántas paradas ha realizado cada uno, el tiempo que tarda en cada vuelta y el tiempo total que cada auto se tomó en la carrera. Además de las funcionalidades básicas, se planea agregar mecánicas adicionales para hacer el programa más interesante. Esto incluirá la introducción de un factor climático que aumentará el riesgo de accidentes cuando llueve, lo que añadirá un elemento de

imprevisibilidad a la carrera. También se considerará la inclusión de diferentes pistas, el usuario podrá elegir en cual correrán los autos, cada una tiene un tiempo diferente de recorrido. Los accidentes y desperfectos mecánicos serán parte del juego, lo que podría retrasar o eliminar a un auto de la carrera, con un mayor riesgo de accidentes en condiciones de lluvia. Al final de cada carrera se imprimirán las posiciones de los autos y se indicará si quedaron descalificados o terminaron la carrera y en qué tiempo la terminaron.

2. Fortalezas y debilidades:

Fortalezas:

- Es una carrera justa ya que se realiza en paralelo.
- Se ve el estado de todos los autos desde principio a fin de la carrera.

Debilidades:

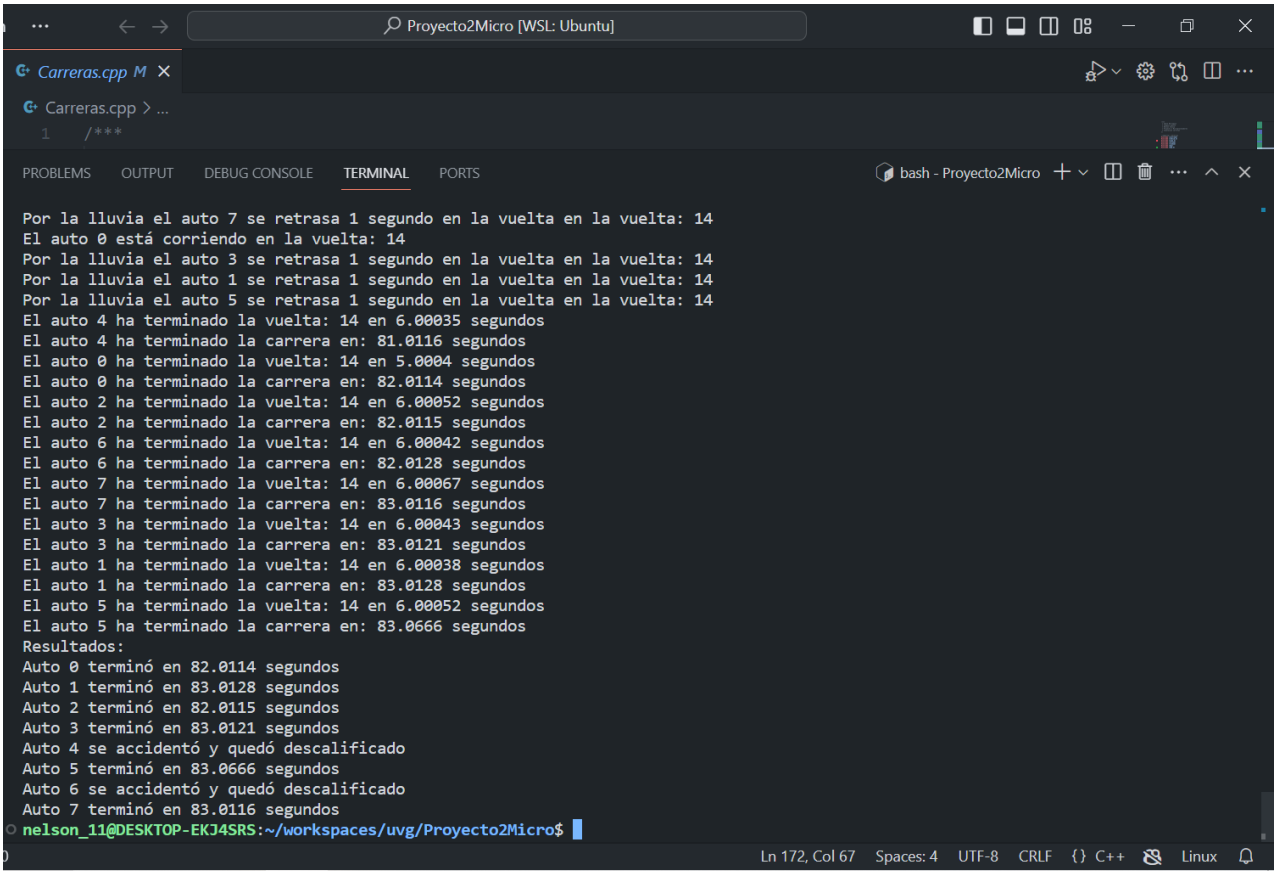
- Puede darse el caso en el que ningún auto termine la carrera.
- El método utilizado para comunicar datos entre main y la subrutina es por medio de variables globales y no de parámetros o returns.

Catálogo de las funciones desarrolladas para implementar el algoritmo de solución:

Nombre de la Función	Tipo de Retorno	Función
simularCarrera	void	Simula una carrera basada en los parámetros dados, como gasolina, llantas, vueltas, tiempo de circuito, probabilidad de lluvia y circuito seleccionado. En esta función, cada thread simula un vehículo compitiendo en la carrera y considera diferentes eventos como cambio de llantas, recarga de gasolina, accidentes y retrasos por lluvia.

randomEntre0Y1	double	Función que genera un número decimal aleatorio entre 0 y 1 utilizando la función rand(). Utilizada para determinar la probabilidad de lluvia y accidentes.
main	int	Función principal del programa que gestiona la configuración de la carrera, la creación de hilos, la espera de la finalización de los hilos y la presentación de los resultados.

Resultados obtenidos:



Durante toda la carrera se dará el estado de cada uno de los autos, si cambian llantas, necesitan gasolina, terminan vueltas de la carrera y en qué tiempo, también por qué vuelta van. Al finalizar el programa, este imprimirá el récord de cada uno de los autos, para verificar quién fue el que ganó.

Conclusiones:

- Al utilizar OpenMP para aplicar paralelismo al programa este permite que el código distribuya la simulación de múltiples autos en la carrera entre diferentes threads, maximizando así la utilización de recursos y potencialmente reduciendo el tiempo de ejecución.
- Cada auto de la carrera es representado por un thread individual y como se ejecutan en paralelo, es posible simular una verdadera carrera en la que todos los coches están en la pista al mismo tiempo y compiten entre sí.
- En cuestión de la simulación de la carrera, los resultados dependen mucho de lo que devuelve la función randomEntre0Y1, la cual utiliza la librería random, ya que, esta genera los números aleatorios para simular la probabilidad de lluvia y accidentes durante la carrera, esto añade un elemento de incertidumbre y realismo a la simulación.

Bibliografía:

Kerrisk, M. K. (2023, 24 junio). *PThreads(7) - Linux manual page*. man7.org. https://man7.org/linux/man-pages/man7/pthreads.7.html

cplusplus.com. (2000). C++ language. https://cplusplus.com/. https://cplusplus.com/doc/tutorial/

LearnCpp.com. (2007). Learn C++. Learn C++. https://www.learncpp.com/

Anexo:

Fase 1: https://docs.google.com/document/d/1K7m4Z4-yRuvzRsKeJ9OF4Y9mM82r3XJPcA_tXv_28Hk/edit?usp=sharing

Link git: https://github.com/mar22266/Proyecto2Micro.git

Variables utilizadas:

Nombre	**Tipo**	**Para qué sirve**
NTHREADS	Macro (int)	Define el número de threads a utilizar
resultados	Arreglo (double)	Almacena los resultados de cada auto en la simulación
gasolina	int	Duración del tanque de gasolina en vueltas
llantas	int	Duración de las llantas en vueltas
vueltas	int	Número de vueltas en la carrera
circuitoTiempo	int	Duración en segundos de cada vuelta
probabilidadLluvia	float	Probabilidad de que llueva
circuito	map<int, int>	Asocia un circuito con su respectivo tiempo
i	int	Variable para for
circuitoSeleccionado	int	Circuito elegido por el usuario
carName	int	Nombre/número del auto basado en el número de thread
startTimeG	chrono::time_point	Marca el tiempo inicial global para el auto
endTime	chrono::time_point	Marca el tiempo final para el auto
elapsedSeconds	duration<double>	Duración que ha pasado desde startTimeG hasta endTime
j	int	Variable para for en simularCarrera

startTime	chrono::time_point	Marca el tiempo inicial para una vuelta específica
isRain	float	Determina si está lloviendo en una vuelta
isAccident	float	Determina si hay un accidente en una vuelta