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:

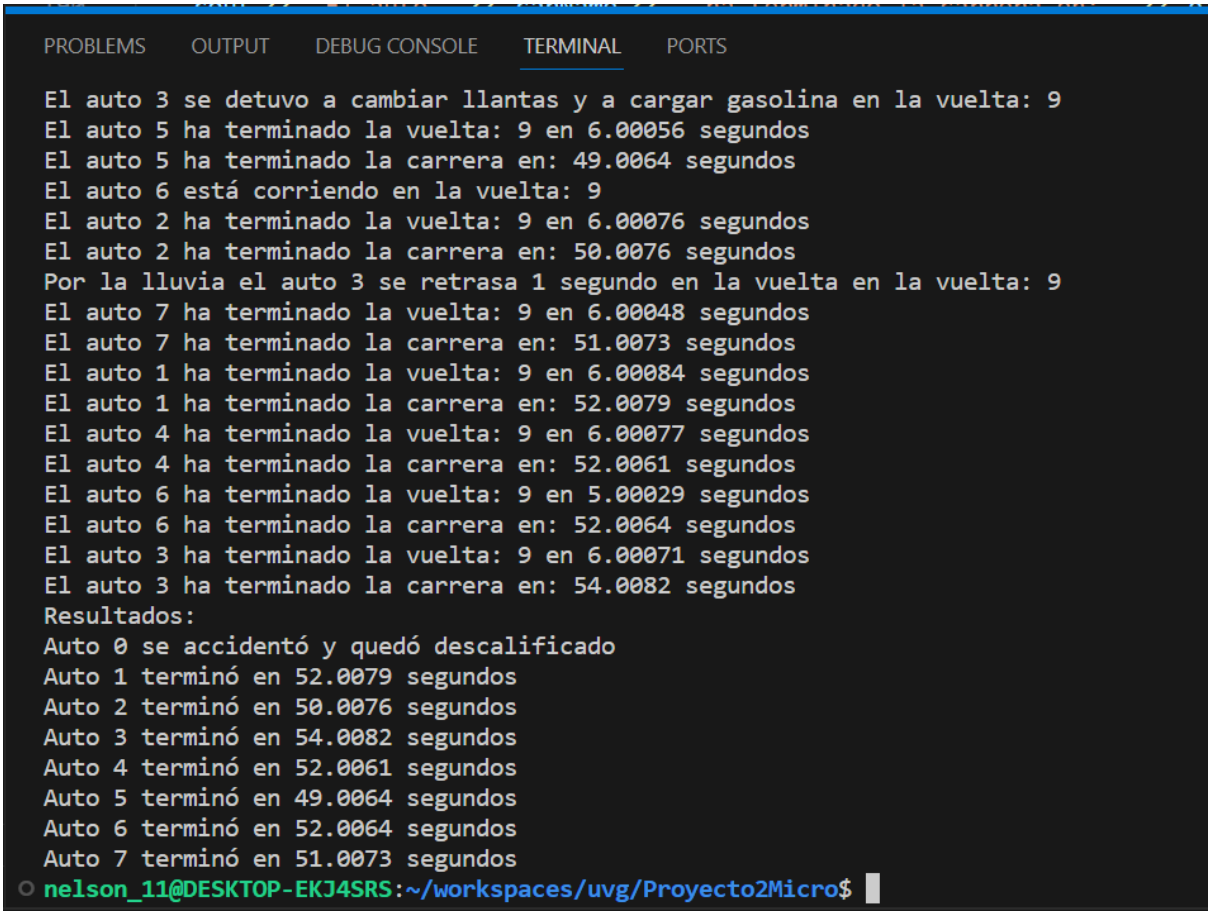
- Puede darse el caso en el que ningún auto termine la carrera.
- El método utilizado para comunicar datos entre main y la subrutina es por medio de variables globales y no de parámetros o returns.

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

Nombre de la Función	Tipo de Retorno	Función
simularCarrera	void *	Función que simula la carrera de un auto de Fórmula 1 como un hilo independiente. Controla las paradas en boxes, el riesgo de accidentes y registra el tiempo de finalización de la carrera.

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:

- El código implementa programación paralela utilizando la biblioteca pthread (POSIX Threads) para crear múltiples hilos de ejecución. Esto permite que todos los autos de carreras simulados compitan al mismo tiempo, lo que es esencial para lograr una carrera "justa". Cada hilo representa un auto y corre de manera concurrente, lo que aumenta la eficiencia y la capacidad de procesamiento del programa.
- El código utiliza punteros para pasar datos a los hilos creados. Por ejemplo, en la función pthread_create, se pasa un puntero a la función simularCarrera junto con el número de identificación del auto como un valor. Esto permite que cada hilo acceda a sus datos específicos y realice la simulación de carrera de manera individualizada.
- 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.

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**	**Función**
gasolina	int	Almacena la duración de las vueltas que dura la gasolina.
llantas	int	Almacena la duración de las vueltas que duran las llantas.
vueltas	int	Almacena la cantidad de vueltas totales para la carrera.
circuitoTiempo	int	Almacena el tiempo asignado para cada vuelta en el circuito seleccionado.
probabilidadLluvia	float	Almacena la probabilidad de lluvia para determinar el riesgo de accidentes.
resultados	double[NTHREADS]	Arreglo que almacena los tiempos de finalización de carrera de cada auto.
circuito	map<int, int>	Un mapa que asigna a cada circuito un tiempo de vuelta.
i	int	Variable de control para bucles y contadores.
rc	int	Código de retorno de funciones pthread.
circuitoSeleccionado	int	Almacena el número del circuito seleccionado por el usuario.

startTimeG	high_resolution_clock::time_point	Marca el tiempo de inicio global de la carrera.
endTime	high_resolution_clock::time_point	Marca el tiempo de finalización de una acción.
elapsedSeconds	duration<double>	Almacena la duración en segundos de un evento.
carName	int	Identificador del auto en la función simularCarrera.
startTime	high_resolution_clock::time_point	Marca el tiempo de inicio de una acción en la carrera.