Projeto de Sistemas Operativos

Tomás Mendes – 2019232272  
Joel Oliveira – djashdasdoij

**Introdução**

Este projeto tinha como objetivo implementar um simulador de corridas, usando todas as estruturas lecionadas e trabalhadas nas aulas PL. Assim, foram usados alguns métodos de sincronizção entre processos e *threads,* tais como semáfros, variáveis de condição e sinais. Foram também usados *named* e *unnamed pipes* bem como *message queues.*

**Explicação de funcionalidades**

***Race Simulator***

O processo principal, *Race\_Simulator*, gere todos os sinais recebidos, reencaminhando-os para os outros processos. Este processo trata dos seguintes sinais:

* SIGINT – envia para *o race manager* *e malfunction manager*, o sinal SIGTERM para acabarem a sua execução.
* SIGUSR1 – reencaminha o sinal para *o race manager* e *malfunction manger* para parar a corrida após todos os carros terem chegado á meta.
* SIGTSTP – chama a função que da *print* as estatisticas da corrida.

É também responsável por da *print* ás estatisticas da corrida. Para tal acede á memória partilhada, faz uma cópia do *array* com todos os carros e ordena os carros com base na sua posição em pista. Este acesso à memória parilhada é sincronizado por uma pequena pausa da corrida de modo a garantir a integridade dos dados. Esta é a única sincronização necessária para o acesso á memória partilhada porque os carros apenas acedem ao seu bloco da memória, não havendo assim conflitos de acesso.

***Race Manager***

Este processo é criado pelo *Race\_Simulator,* é responsável por ler todos os comandos enviados para o *named pipe – race\_pipe* e pelos *unnamed pipes* de todas as equipas, depois de a corrida começar. Como tal, verifica todos os comandos recebidos pelo *named pipe* e adiciona os carros à memória partilhada para depois serem criados os carros pelos equipas. Quando recebe o comando “ADDCAR” faz todas as verificações necessárias para a adição de um carro a uma certa equipa. As equipas são também criadas quando é inserido um carro numa equipa que ainda não exista. Após receber o comando “START RACE”, cria os processos das equipas e verifica, na memória partilhada, se todas as equipas têm pelo menos um carro e envia o sinal SIGUSR2 para todos os processos equipa e *malfunction manager* para informar do inicio da corrida.

***Team Manager***

Este processo é criado pelo *race\_manager.* Acede á sua zona da memória partilhada, que corresponde ao seu índice + Nº max de carros de cada equipa e cria as *thread* carro e inicializa a estrutura do carro que está em memória partilhada. As *threads* carro correm uma função com um loop onde são feitas todas as operações relacionadas com o carro: incrementar a posição na pista, decrementar o combustivel, alterar a sua velocidade e consumo com base no mode em que está, verificar se acabou um volta, se vai entrar na box, etc. Após receber o sinal SIGTERM, enviado pelo *Race\_Simulator* depois de receber o SIGINT, reencaminha-o para as *threads* que acabam quando o carro passar na meta, mesmo que não seja a sua última volta da corrida. Cada equipa tem também um *pthread\_mutex* para controlar o acesso á variável com o estado da box da equipa.