Relatório Projeto SO

Para guardar os carros, optei por colocá-los num array na memória partilhada e tratei o array como fosse uma matriz, ou seja, uma “linha” por equipa e as “colunas” para colocar os carros. Exemplo com 3 equipas e 2 carros:

Equipa 0 carro 0	Equipa 0 carro 1	Equipa 1 carro 0	Equipa 1 carro 1	Equipa 2 carro 0	Equipa 2 carro 1

Quando o sinal SIGUSR1 é recebido optei por deixar os carros chegar à meta e ficarem em espera para que possam continuar de onde ficaram, ou seja, todos os atributos são conservados e permanecem na pista.

Quando o sinal SIGINT é recebido, se não houver nenhuma corrida, optei por mandar o sinal SIGUSR2 para os processos gestor de equipas porque estão há espera de um semáforo, terminando e limpando os recursos de imediato. Caso uma corrida esteja ativa, tudo acontece normalmente (segundo o enunciado).

Para gestão da corrida, sempre que um carro é adicionado é verificado se a equipa já existe, caso exista, tenta adicionar o carro à equipa, caso não exista, cria um novo gestor de equipas e é lhe atribuído a equipa do carro. Os gestores de equipas ficam num semáforo à espera que a corrida seja iniciada. Quando o comando para iniciar a corrida é recebido, o gestor de corridas abre o semáforo para todos os gestores de equipas, os gestores de equipas iniciam as threads dos carros. Sempre que um carro muda de estado, o gestor de corrida é notificado através do unnamed pipe e o gestor de equipa é notificado através de um semáforo, para verificar se é necessário alterar o estado da box. Caso o estado da box mude para ocupada, o carro na box é atestado e reparado caso esteja estragado, de notar que todo este processo acontece no gestor de equipa. Sempre que um carro termina a corrida, notifica o gestor de corrida e fica a aguardar que todos os carros terminem, quando todos os carros terminarem, o gestor de corrida liberta o semáforo para as threads de todos os carros serem terminadas e para o gestor de equipas sair do modo corrida e aguardar pelo novo início de corrida.

Mecanismos de sincronização:
mutexBox – semáforo para verificar se a box já está a ser acedida por outra thread.
pipeSem – semáforo para verificar se outra thread está a escrever no unnamed pipe.
mudouEstadoCarro – semáforo para alertar gestor de equipa sempre que um carro muda de estado.
naBox – semáforo para prender carro na box enquanto é atestado e reparado.
sem – semáforo que liberta os processos gestor de equipas quando a corrida começa e que liberta as threads para terminarem quando a corrida termina.
mutexLog – semáforo para escrever uma string de cada vez no sdout e no ficheiro de log.

Optei por receber o input do utilizador no processo principal, simulador de corrida e redirecioná-lo para a named pipe para não ser necessário abrir um terminal à parte para enviar os comandos para o named pipe.