ADSOF
Memoria Práctica 4

Fecha: 29/04/2021
Grupo: 2262

David Teófilo Garitagoitia Romero
Javier Fernández de Alarcón Gervás

Apartado 1:

Para este apartado lo primero fue la creación de la interfaz IVehicle y su implementación en la clase Vehículo, los métodos comunes se definieron en la propia clase, pero métodos como updatePosition() son abstractos, ya que depende del vehículo en concreto por lo que se implementaron en las clases especificas, a continuación se crea la clase RaceReader con el método read que permite cargar una carrera a través de un fichero, para ello lee la primera línea donde se especifica la longitud de la carrera, a continuación mientras queden por leer, guarda el número de vehículos con las características indicadas a continuación en la línea (tipo y velocidad máxima) y llama a una función auxiliar para crear y añadir el vehículo tantas veces como se indicó al inicio de la línea y se crea una nueva excepción que emplearemos cuando se desconozca el tipo de vehículo pasado por argumento, por último se crea la clase Race que contará con una longitud y una lista de vehículos participantes en la carrera y se implementa el método toString que mostrará la situación actual de la carrera, para ello primero se imprime la longitud de la carrera y después por cada vehículo participante se imprimirá, la información aislada del vehículo y la información del vehículo con respecto al resto de participantes de la carrera

Apartado 2:

Para este apartado se solicita la implementación del método para simular la carrera, para ello lo primero será llevar a cabo las modificaciones del método updatePosition() siguiendo las instrucciones del pdf.

Para la simulación se imprime el inicio de turno, la situación de la carrera mediante el toString() del anterior apartado, luego se pasa a la fase de movimiento que consiste en hacer un updatePosition() de todos los participantes de la carrera, a su vez en la fase se comprueba si hay ganador, en cuyo caso retornará el vehículo ganador.

Respuesta a la pregunta: Si se mantienen a la misma velocidad, el 90% de la velocidad del coche debería ser equivalente a la velocidad máxima de la moto para así garantizar que siempre va como mínimo a igual velocidad que la motocicleta, en este caso debería ir a velocidad máxima = 10

Apartado 3:

En este apartado hay que implementar las clases de los componentes, al igual que su abstracta correspondiente (debido a que hay funcionalidad en común), y también los métodos definidos en la interfaz ya definida.

En cuanto al daño de los componentes y sus correspondientes características, se ha implementado la función repair para la fase de reparación de modo que compruebe los turnos que lleva dañado el componente y el coste de reparación del mismo, después, continuando la fase de reparación se comprueba si el componente se consiguió reparar completamente, en caso contrario, si ese componente es crítico se pone un flag a true

para así mantener que no se puede mover, y para el ataque se comprueba si el vehículo puede o no atacar (si cuenta con algún componente no dañado que pueda atacar), después se comprueba si tiene alguien a quien atacar y por último se intenta realizar el ataque con las probabilidades especificadas en el pdf, para ellos se ha hecho además del método de attackPhase(), el método getDelante(IVehicle vehiculo) que retorna en caso de existir el vehículo inmediatamente por delante del que se pasa por argumento, por último se llama a la función de atacar pasándole el objetivo y con un 50% de probabilidades realizará un ataque a un componente aleatorio del objetivo, al implementar los componentes es necesario reestructurar las funciones toString() para que ahora también impriman la información sobre el estado de los componentes, además el método read de RaceReader también debera ser modificado de forma que se tenga en cuenta los componentes a añadir en los participantes.

Apartado 4:

Los power ups se han implementado de forma similar a los componentes, con interfaz y clase abstracta, pero en lugar de estar relacionados con el vehículo como los componentes, estos son acciones en la carrera más que propiedades.

Igual que en las otras dos fases de la carrera, hay dos métodos para realizarse. En primer lugar se establece la probabilidad de suceder dicha fase y en el otro método, la probabilidad de cada power up.

El tercer power up se ha definido como una habilidad que haga que un vehículo repare todos sus componentes automáticamente. Se itera por todos los componentes y se emplea el método de reparación usado en la fase de reparación.