

Simulación de carreras de coches

Fecha de entrega de la versión 2: **20 de noviembre de 2023**



(Imagen extraída de <http://motorsportsimulator.com>)

El objetivo del proyecto es realizar un programa que permita simular carreras de coches. La idea es desarrollar el proyecto, de forma incremental, para ir incluyendo en cada versión los distintos conocimientos que se imparten en la asignatura.

1. La simulación en detalle

Una carrera de coches está compuesta por una carretera con varios carriles, donde en cada carril hay un coche. Es importante matizar que **en esta versión nos centraremos en una carretera con un solo carril**, pero en la versión posterior podremos simular carreras con varios carriles y varios coches. Todos los coches empiezan en la *posición inicial* 0 y en cada turno, si es posible, avanzan. Cada posición de un carril puede ser de tres tipos:

- ☐ **Posición normal.** Este tipo de posiciones no tiene ningún efecto sobre el coche.
- ☐ **Posición con *clavo*.** Las posiciones con clavo hacen que el coche pinche y por tanto permanezca parado un número determinado de turnos, mientras se repara el pinchazo.
- ☐ **Posición *sorpresa*.** Las posiciones sorpresa hacen que el coche se mueva a otra posición sorpresa. La forma de calcular a qué posición se traslada el coche es la siguiente: generamos un número aleatorio entre 0 y 1 (ambos incluidos). Si sale 0, se avanza a la siguiente posición sorpresa. Si sale 1, se retrocede hasta la posición sorpresa anterior. En caso de no existir dicha posición sorpresa, el coche se coloca en la posición 0 (posición inicial).

La posición inicial en cualquier carril nunca puede contener ni un clavo ni una sorpresa. Además, cada carril puede tener una configuración distinta, es decir, los clavos y sorpresas de carriles

diferentes, pueden estar en posiciones distintas. Otro aspecto importante es que, en una posición de un carril determinado, no puede haber un clavo y una sorpresa simultáneamente.

La simulación termina cuando todos los coches han llegado a la meta, en cuyo caso se muestra el pódium (orden de llegada de los coches), y se permite, si el usuario lo desea, repetir la simulación. El avance de los coches se puede hacer en dos modos distintos: *normal* y *depuración*. En el modo normal, un coche avanza un número aleatorio de pasos, que oscilará entre 1 y una constante `MAX_PASOS`. En el modo depuración será el usuario quien introducirá el número de pasos a avanzar. En este caso no hay límite y se pueden introducir valores negativos que harán que el coche retroceda. Hay que tener cuidado de no mandar el coche a posiciones negativas. Si esto ocurriera, habrá que colocar el coche en la posición inicial. En secciones posteriores explicaremos cómo generar números aleatorios.

2. Carretera con un único carril

En esta versión simularemos la lógica de una carrera en la que participa un único coche que circula por una carretera de un único carril. La configuración de la carretera se cargará de un archivo de texto. Por ejemplo, el archivo `"carriles.txt"` que suministramos con el código, es de la forma:

```
CLAVO 3 2 6 9
SORPRESA 3 3 5 7
XX
```

donde puedes observar que, en cada línea, se especifican las posiciones en las que hay clavo o sorpresa. El fichero termina con el centinela `"XX"`. Concretamente, en este archivo:

- ☐ La primera línea contiene información sobre posiciones con clavo. Aparece el identificador `CLAVO` seguido de 3, 2, 6 y 9. El primer entero 3 indica que habrá tres clavos, colocados en las posiciones 2, 6 y 9.
- ☐ La segunda línea hace referencia a posiciones sorpresa (identificador `SORPRESA`). En este caso hay 3 sorpresas en las posiciones 3, 5 y 7.
- ☐ El resto de posiciones del carril son posiciones normales.

Ten en cuenta que el fichero puede tener más de dos líneas, de ahí el uso del centinela `"XX"`. Por ejemplo, el archivo anterior podría contener también la línea `CLAVO 2 4 8`, que indicaría que además de las posiciones `CLAVO` de la primera línea, hay otros dos clavos en las posiciones 4 y 8. Asumiremos que el archivo es correcto, es decir que las posiciones están dentro de la longitud del carril y que en una misma posición no puede haber un clavo y una sorpresa a la vez. La posición de salida (posición cero) no podrá tener ni clavo ni sorpresa.

2.1. Detalles de implementación

Definiremos las siguientes constantes:

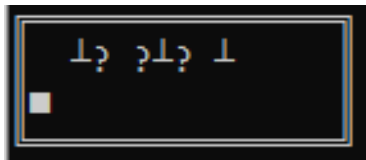
- ☐ `LONG_CARRETERA`: Representa la longitud del carril por el que circula el coche, es decir, el número de posiciones que tiene. El carril lo forman las posiciones comprendidas entre 0 y `LONG_CARRETERA-1`. La posición 0 es la salida de la carrera y no se pueden colocar ni clavos ni sorpresas en esta posición. Inicializarla al valor 10.

- **MAX_PASOS**: Define el número máximo de posiciones que puede avanzar un coche cuando la simulación se lleva a cabo en modo normal. En modo depuración no existe límite para indicar el avance (o retroceso) del coche, pero hay que controlar que el coche no se vaya a posiciones negativas. Inicializarla al valor 2.
- **TIEMPO_PARADO**: Representa el número de turnos que un coche permanece parado cuando cae en una posición clavo (ha sufrido un pinchazo). Inicializarla al valor 2.
- **DEBUG**: Define el modo de ejecución, normal o depuración. La constante será de tipo *bool* de tal forma que si vale *true* ejecutaremos el modo depuración. En otro caso se ejecutará el modo normal.

Definiremos también constantes para representar los valores ASCII de los caracteres que utilizaremos para dibujar la carretera.

- `CHAR_LINEA_HORIZONTAL = 205;` // `=`
- `CHAR_ESQUINA_SUPERIOR_IZQUIERDA = 201;` // `┌`
- `CHAR_ESQUINA_SUPERIOR_DERECHA = 187;` // `┐`
- `CHAR_ESQUINA_INFERIOR_IZQUIERDA = 200;` // `└`
- `CHAR_ESQUINA_INFERIOR_DERECHA = 188;` // `┘`
- `CHAR_LINEA_VERTICAL = 186;` // `||`
- `CHAR_COCHE = 254;` // `■`
- `CHAR_CLAVO = 193;` // `⊥`
- `CHAR_SORPRESA = 63;` // `?`
- `CHAR_NORMAL = 32;` // `' '`

Por ejemplo, la situación inicial para la carretera representada en el fichero “`carriles.txt`” sería:



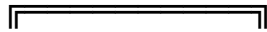
Para representar las posiciones de la carretera, definiremos un tipo enumerado `tTipoPosicion` con los valores `NORMAL`, `CLAVO` y `SORPRESA`. La carretera estará representada por el tipo `tCarretera`, un array de elementos del tipo `tTipoPosicion` y longitud `LONG_CARRETERA`. Implementaremos, entre otras, las funciones que se indican a continuación.

1) funciones para leer del archivo de texto y cargar la carretera.

- `tTipoPosicion stringToEnum(string s)`: transforma el string `s` en su correspondiente elemento del tipo enumerado `tTipoPosicion`.
- `bool cargaCarretera(tCarretera carretera)`: pide el nombre del archivo al usuario. Lo abre y carga los datos en `carretera`. Devuelve `true` sí y sólo sí la apertura del archivo ha sido correcta.

2) funciones de visualización de la carretera:

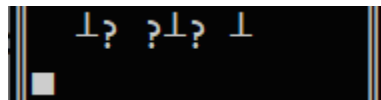
- ❑ `void dibujaLineaHorizontalSuperior()`: dibuja la línea superior de la carretera, teniendo en cuenta la longitud de la misma.



- ❑ `void dibujaLineaHorizontalInferior()`: dibuja la línea inferior de la carretera, teniendo en cuenta la longitud de la misma.



- ❑ `void dibujaCarril(const tCarretera carretera, int posCoche)`: dibuja el contenido del carril, es decir los clavos (\perp), las sorpresas (?) y el coche (■), que está en la posición `posCoche`. El carril realmente ocupa dos líneas. En la primera se colocan las posiciones clavo y sorpresa, y en la segunda línea se coloca el coche. De esta forma la visualización es más representativa. Observa además que cada línea empieza y acaba con `||`. El siguiente dibujo muestra el carril del archivo “`carriles.txt`”, donde el coche está en la posición inicial.



- ❑ `void dibujaCarretera(const tCarretera carretera, int posCoche)`: utiliza las funciones anteriores para dibujar la carretera completa.

2.1.1. La función main

Coloca el coche en el inicio (posición 0) e inicializa su tiempo parado a 0. Posteriormente se carga la configuración de la carretera de un archivo de texto, cuyo nombre se pide por consola. Se llama a la función `dibujaCarretera` para probar las funciones de carga de datos y mostrar la carretera.