# Monte Carlo y Value Iteration

El multimillonario Nylon Mask, como parte del proyecto que tiene de conducción automática, nos asignó la tarea de manejar las revoluciones del motor del auto. Cómo somos expertos en el área, en seguida nos damos cuenta que esto lo podemos representar como un MDP. Dependiendo de la temperatura del motor, las acciones que tomaríamos sería aumentar o disminuir las revoluciones del motor, y la recompensa sería en base a la distancia recorrida, o una penalización si se rompe el motor. El problema es que no sabemos cómo son las transiciones del MDP. Por suerte, como Nylon Mask es tan excéntrico, nos dejó probar y romper la cantidad de motores que nos dé la gana.

## Ejercicio 1 - Monte Carlo

Lo primero que tenemos que hacer entones es aprender el MDP

En el archivo Engine.py se encuentra la clase Engine la cual representa el MDP del motor, que desconocemos. Como cumple con gym.Env podemos interactuar con el ambiente mediante las funciones reset y step. Para ambos casos en el diccionario info (el último elemento de la tupla) es un diccionario con una única llave ‘actions’ que contiene la lista de acciones posibles para ese estado.

Por ejemplo si corremos las siguientes lineas de codigo:

```
eng = Engine()
print(eng.reset())
```

El output sera el siguiente

```
('Cool', {'actions': ['slow', 'fast']})
```

‘Cool’ siendo el estado inicial, y [‘slow’,’fast’] siendo las acciones posibles

Se pide:

1. Estimar el MDP utilizando el método Monte Carlo y guardarlo en un pickle (.pkl).
2. Crear una policy cualquiera y estimar el valor de la misma mediante el método Monte Carlo.

## Ejercicio 2 - Value Iteration

Ahora que ya contamos con el MDP, podemos hacer la tarea que nos pidieron

Se pide:
Con el MDP calculado en el Ejercicio 1.1, implementar el algoritmo de Value Iteration para encontrar la policy óptima.