Instrucciones Miniproyecto 2

Descripción

En este miniproyecto deberás construir una simulación del funcionamiento de vehículos utilizando programación orientada a objetos.

Archivos entregados

Para este trabajo, se entregará una carpeta *MP2.zip* que contiene dos archivos:

- *main.py*: Corresponde al archivo principal del programa en el que deberás completar lo solicitado. Este es el único archivo que debes modificar.
- *parametros.py*: Corresponde a un archivo que contiene valores estáticos a utilizar dentro del código.

Código entregado

En el archivo *main.py* se encuentra completamente implementada la clase Rueda, por lo que **no** debes modificarla. A continuación, se describe la clase:

Esta clase posee los siguientes atributos:

- *resistencia_actual*: Corresponde a la resistencia actual de la rueda. Esta se inicializa como un *int* aleatorio entre los números almacenados en la lista RESISTENCIA que se encuentra en el archivo *parametros.py*.
- *resistencia_total*: Corresponde a la resistencia total de la rueda. Esta se inicializa con el valor inicial del atributo *resistencia_actual*.
- *estado*: Es el estado en el que se encuentra la rueda. Se inicializa en el *string* “Perfecto”.

Además, esta clase posee los siguientes métodos:

- *gastar(acción, tipo)*: Recibe el argumento acción, que corresponde a un *string* que puede tener el valor “acelerar” o “frenar” y el argumento tipo, que corresponde a un *string* que puede ser “automovil” o “moto”. Dependiendo de la acción y tipo, el valor del atributo resistencia_actual de la rueda disminuye. Este método no retorna nada.

- *actualizar_estado()*: Este método no recibe argumentos. Se encarga de actualizar el estado de la rueda según el nuevo valor de la *resistencial_actual* en comparación a la resistencia_total de la rueda. Este método no retorna nada.

Trabajo a realizar

Utilizando programación orientada a objetos se deben realizar las siguientes tareas en el archivo *main.py*:

Parte 1: Definición de funciones y clases.

Antes de definir tus clases, definirás una función que usarás dentro de tus clases. Esta función es:

- *avanzar(velocidad, tiempo)*: Recibe como argumento un *float* que corresponde a la velocidad del vehículo en m/s y un *int* que corresponde al tiempo expresado en segundos. Entrega la cantidad de kilómetros avanzados en un rango de tiempo dado de acuerdo con la siguiente fórmula:

  kilometraje = velocidad en m/s * tiempo

  Esta función retorna el kilometraje avanzado durante el tiempo ingresado si se viaja a la velocidad indicada.

1. **Clase Automóvil.**

Debes definir una clase Automóvil que contenga los siguientes atributos, cuyos valores deben ser inicializados mediante el método *init*:

- *kilometraje*: Corresponde al kilometraje del vehículo medido en km. Este se recibe como parámetro al inicializar la clase.

- *ano*: Es el año de fabricación del vehículo. Se recibe como parámetro al inicializar la clase.

- *ruedas*: Es una lista de objetos de tipo Rueda. Se inicializa como una lista de 4 objetos de clase Rueda.

- *aceleracion*: Corresponde a la aceleración del vehículo medida en km/h^2. Se inicializa en 0.

- *velocidad*: Corresponde a la velocidad del vehículo medida en km/h. Se inicializa en 0.

Además, la clase debe poseer los siguientes métodos:

- *avanzar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en segundos. Incrementa el atributo kilometraje de acuerdo con el resultado de la función *avanzar* definida previamente.

  Este método no retorna nada. Recuerda que la velocidad almacenada en el atributo velocidad de la clase está medida en km/h, por lo que deberás primero convertirla antes de entregar la velocidad a la función *avanzar.* Para convertir un valor de km/h a m/s, debes multiplicarlo por $\frac{1000}{3600}$.

- *acelerar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en horas. Primero debes agregar *tiempo*0.5* al atributo aceleración. Luego incrementa la velocidad del vehículo según el nuevo atributo aceleración de acuerdo con la fórmula *velocidad += aceleración * tiempo * 3,6*. Después llama al método *avanzar(tiempo)*, entregándole como atributo el tiempo recibido anteriormente y luego, ejecuta el método *gastar()* de cada uno de los objetos guardados en la lista ruedas entregándole como argumento el *string* “acelerar” y el tipo “automovil". Finalmente, devuelve el atributo aceleración a 0.

  Este método no retorna nada.

- *frenar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en horas. Primero resta *tiempo*0.5* al atributo aceleración. Luego, disminuye la velocidad del vehículo de acuerdo con el nuevo atributo aceleración, con la fórmula *velocidad += aceleración * tiempo * 3,6* (dado que aceleración tendrá un valor negativo en ese momento, realizar esta suma debería disminuir la velocidad). Si la velocidad queda negativa, se tiene que dejar en 0. Después llama al método *avanzar(tiempo)* entregándole como atributo el tiempo recibido anteriormente y luego, ejecuta el método *gastar()* de cada uno de los objetos guardados en la lista ruedas entregándole como

argumento el *string* “frenar” y el *string* “automovil". Finalmente devuelve el atributo aceleración a 0. Este método no retorna nada.

- *obtener_kilometraje()*: Este método no recibe argumentos y sólo retorna el valor del atributo kilometraje.

- *reemplazar_rueda()*: Este método no recibe argumentos. Debe buscar dentro de la lista ruedas la rueda con menor resistencia y eliminarla de la lista. Luego debe instanciar un nuevo objeto de la clase Rueda y añadir este a la lista ruedas. No retorna nada. En caso de que haya dos o más ruedas con el valor mínimo queda a su criterio cuál reemplazar.

2. Clase Moto.

Esta clase posee los mismos atributos que la clase automóvil, y además un atributo adicional a los que ya tiene de la clase Automovil:

- *cilindrada*: Se recibe como parámetro al inicializar la clase, es número *int* no negativo (puede ser 0).

Además, su inicializador inicializa el atributo ruedas como una lista de dos objetos de clase Rueda en vez de 4.

Además, debe contener los siguientes métodos:

- *avanzar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en segundos. Incrementa el atributo kilometraje de acuerdo con el resultado de la función *avanzar* definida previamente.

  Este método no retorna nada. Recuerda que la velocidad almacenada en el atributo velocidad de la clase está medida en km/h, por lo que deberás primero convertirla antes de entregar la velocidad a la función *avanzar.* Para convertir un valor de km/h a m/s, debes multiplicarlo por $\frac{1000}{3600}$.

- *acelerar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en horas. Primero debes agregar *tiempo*0.8* **+** *cilindrada*0.2* al atributo aceleración. Luego incrementa la velocidad del vehículo según el nuevo atributo aceleración de acuerdo con la fórmula *velocidad* += *aceleración* * *tiempo* * *3*. Después llama al método *avanzar(tiempo)*, entregándole como atributo el tiempo recibido anteriormente y luego, ejecuta el método *gastar()* de cada uno de los objetos guardados en la lista ruedas

entregándole como argumento el *string* “acelerar” y el tipo “moto". Finalmente, devuelve el atributo aceleración a 0.

Este método no retorna nada.

- *frenar(tiempo)*: Recibe como argumento un *int* que corresponde al tiempo expresado en segundos. Primero resta *tiempo*0.8 + cilindrada*0.2* al atributo aceleración. Luego, disminuye la velocidad del vehículo de acuerdo con el nuevo atributo aceleración, con la fórmula *velocidad* += *aceleración* * *tiempo* * *3*. Si la velocidad queda negativa, se tiene que dejar en 0. Después llama al método *avanzar(tiempo)* entregándole como atributo el tiempo recibido anteriormente y luego, ejecuta el método *gastar()* de cada uno de los objetos guardados en la lista ruedas entregándole como argumento el *string* “frenar” y el tipo “moto". Finalmente devuelve el atributo aceleración a 0. Este método no retorna nada.

- *obtener_kilometraje()*: Este método no recibe argumentos y sólo retorna el valor del atributo kilometraje.

- *reemplazar_rueda()*: Este método no recibe argumentos. Debe buscar dentro de la lista ruedas todas las ruedas cuya resistencia actual sea estrictamente menor a la mitad de su resistencia total y eliminarlas de la lista. Luego, por cada rueda eliminada debe instanciar un nuevo objeto de la clase Rueda y añadir este a la lista ruedas. No retorna nada.

Parte 2: Completar acciones.

En esta parte, debes completar la función *accion(vehiculo, opcion)* que se encuentra en el archivo main.py. Esta función recibe como argumentos una instancia de un vehículo y un *int* que corresponde a la acción a realizar. Debes completar cada condición según lo indicado a continuación.

- **Acelerar (opción 2):** Primero debes permitir al usuario que escoja un tiempo en horas para acelerar el vehículo, el cual puede ser decimal. Luego debes llamar al método *acelerar(tiempo)* del vehículo, entregándole como argumento el tiempo ingresado por el usuario. Finalmente, se debe imprimir en consola un mensaje que diga “Se ha acelerado por X horas, llegando a una velocidad de Y km/h”, donde X corresponde al tiempo e Y corresponde a la velocidad actual del vehículo.

- **Frenar (opción 3):** Primero debes permitir al usuario que escoja un tiempo en horas para frenar el vehículo, el cual puede ser decimal. Luego debes

llamar al método *frenar(tiempo*) del vehículo, entregándole como argumento el tiempo ingresado por el usuario. Finalmente, debes imprimir en consola un mensaje que diga “Se ha frenado por X horas, llegando a una velocidad de Y km/h”, donde X corresponde al tiempo e Y corresponde a la velocidad actual del vehículo.

- **Avanzar (opción 4):** Primero debes permitir al usuario que escoja un tiempo en segundos para avanzar el vehículo, el cual puede ser decimal. Luego debes llamar al método *avanzar(tiempo)* del vehículo, entregándole como argumento el tiempo ingresado por el usuario. Finalmente, debes imprimir en consola un mensaje que diga “Se ha avanzado por X segundos a una velocidad de Y km/h”, donde X corresponde al tiempo e Y corresponde a la velocidad actual del vehículo.

- **Cambiar rueda (opción 5):** Debes llamar al método *reemplazar_rueda()* del vehículo actual. Finalmente se debe imprimir un mensaje que diga “Se han reemplazado las ruedas con éxito”.

- **Mostrar estado (opción 6):** Debes imprimir la información del vehículo actual en el siguiente orden: año, velocidad, kilometraje. Además, por cada rueda del vehículo debes imprimir su estado actual.

Parte 3: Completar código principal.

En esta parte deberás instanciar vehículos en la función main.py. En particular, deberás crear dos objetos: uno de clase Moto y uno de clase Automóvil. Los atributos iniciales con los que se completen ambos objetos deben ser valores definidos por ti, pero respetando los tipos especificados en el enunciado. Luego debes agregar ambos objetos a la lista de vehículos existente, la cual comienza vacía.

Archivos a entregar

Debes entregar el archivo *main.py* con el código que hace funcionar tu programa y el archivo *parámetros.py* con los parámetros que hayas utilizado.