Ejercicio 1.- Se solicita crear una clase, la vamos a llamar Vehículo que va a tener un enumerado) MarcaDeVehiculo) para las distintas marcas. Los valores de las marcas van a ser BMW, MERCEDES, AVENSIS, TOYOTA, SEAT, NISSAN y AUDI. La clase va a tener como atributos la matrícula (tipo String) y la marca (tipo MarcaDeVehiculo) y los métodos getters y setters de los atributos. Escribe un pequeño programa de prueba donde se solicite por consola la matrícula y la marca y verifiques que los métodos funcionan correctamente.
El método Setter valorará antes de hacer el Set que la marca introducida es válida y se corresponde con un valor del enumerado

Ejercicio 2.- Modificación del anterior

1. Cambiar el nombre del método getMarca y llamarlo getMarcaString

2. Crea un método getMarca que devuelva un objeto de tipo MarcaDeVehiculo

3. Cambia el nombre del método setMarca y llamarlo setMarcaString

4. Crea un método setMarca que reciba un objeto de tipo MarcaDeVehiculo y en base al valor recibido establezca el atributo correspondiente

5. En el main, tras mostrar los datos del vehiculo con los métodos anteriores, modifica la marca operando con valores de MarcaDeVehiculo directamente.

El objetivo de todo esto es ver cómo se puede trabajar sin String usando directamente objetos del tipo enumerado

Ejercicio 3.- Crear un enumerado que contenga estos valores {COCHE, CAMION, BARCO, TREN, AVION} y muestre su número de orden y haga comparaciones indicando quién puede ser más rápido que otro. Por ejemplo: “BARCO es más rápido que COCHE” o “COCHE es más lento que TREN” o “CAMIÓN es igual de rápido que CAMIÓN”

Ejercicio 4.- Diseñar la clase Texto que gestiona una cadena de caracteres con algunas características:

- La cadena de caracteres tendrá una longitud máxima que se indicará en el constructor
- Se podrá añadir un carácter al principio o al final siempre que exista espacio disponible
- Se podrá añadir una cadena al principio o al final siempre que exista espacio disponible
- Se debe contar cuántas vocales hay en el texto.
- Mediante los métodos iremos creando el texto.

Ayuda:

- Para saber la longitud de la cadena de caracteres se usará el método length()
- Para buscar en una cadena de caracteres un carácter en concreto usar indexOf(carácter), devuelve -1 si no lo encuentra

Ejercicio 5.- Se quiere definir una clase (SintonizadorFM) que permita controlar un sintonizador digital de emisoras FM; concretamente, se desea dotar al controlador de una interfaz que permita subir (up) o bajar (down) la frecuencia, en saltos de 0,5 MHz y mostrar la frecuencia en un momento dado Display. El rango de frecuencias va a oscilar entre 80 y 108 MHz, y al inicio, el controlador sintonice 80MHz. Si durante una operación de subida o bajada, se sobrepasan los límites, la frecuencia sintonizada debe pasar al extremo contrario. Escribir un programa principal básico para probar su funcionamiento.

Ejercicio 6.- Se desea obtener la fecha del día y pedir por teclado un número entero de segundos. Se solicita que se muestre la hora del día y las n siguientes horas que se diferencian en un segundo.

Habrá que crearse la clase Hora que se seteará con la hora del sistema y que dispondrá de los atributos hora, minutos y segundos

Usar Try/catch para validar que los segundos es un valor válido

Ejercicio 7.- Crea la clase Fracción. Los atributos, en principio, serán numerador y denominador.
El numerador y denominador se introducirán por pantalla. No es necesario usar Try pero sí controlar que el denominador no sea negativo. Se crearán métodos para simplificar la fracción, multiplicarla y dividirla por un número y por una fracción e invertirle el signo.
Para simplificar la fracción ayudaros del método Math.min que devuelve el mínimo de dos números (el tipo devuelto es el mismo tipo de los números que compara)

Ejercicio 8.- Implementar una clase *Punto* cuyos datos miembros sean las coordenadas de un punto del plano. Estos datos han de ser privados. Para esta clase se piden los siguientes métodos y constructores:

- El constructor Punto() recibe como argumento dos números reales, a y b, y construye un nuevo punto de la clase Punto cuyas coordenadas son a y b.
- Los métodos de acceso getX() y getY(), sin argumentos, que devuelven las coordenadas de un objeto Punto.
- Los métodos modificadores setX() y setY(), que reciben un argumento y modifican el valor de la correspondiente coordenada de un objeto Punto.
- El método igual(), que comprueba si un objeto de la clase Punto es igual a otro dado que se pasa como argumento.
- El método distancia(), sin argumentos, que calcula la distancia de un objeto de la clase Punto al origen de coordenadas.
- El método distancia(), que calcula la distancia de un objeto de la clase Punto a otro que se proporciona como argumento.

Ejercicio 9.- Implementar una clase PersonaC con las siguientes características:

- Atributos: String Nombre, int edad, String DNI y PesoAlturaC pesoAltura (atributo del tipo PesoAlturaC);

- Tanto la clase PersonaC como la clase PesoAlturaC tendrán los constructores, setter y getter necesarios;
- Ambas clases implementarán la interfaz cloneable;
- La clase Persona escribirá dos métodos clone() uno superficial y otro profundo
- Haced un pequeño programa principal para clonar objetos y modificarlos de modo que se vea que cuando se clona superficialmente, los cambios realizados a los atributos objetos se reflejan tanto en el objeto original como en el copiado, pero con la copia profunda no ocurre eso.

Ejercicio 10.- Implementar una clase Gato con las siguientes características:

- Atributos (String) nombre, color, raza, (Int) edad
- Constructores necesarios
- Que implemente dos método CompareTo para compararlos por nombre y por edad
- Haced un programa principal sencillo que cree dos gatos y pruebe los métodos de comparación.