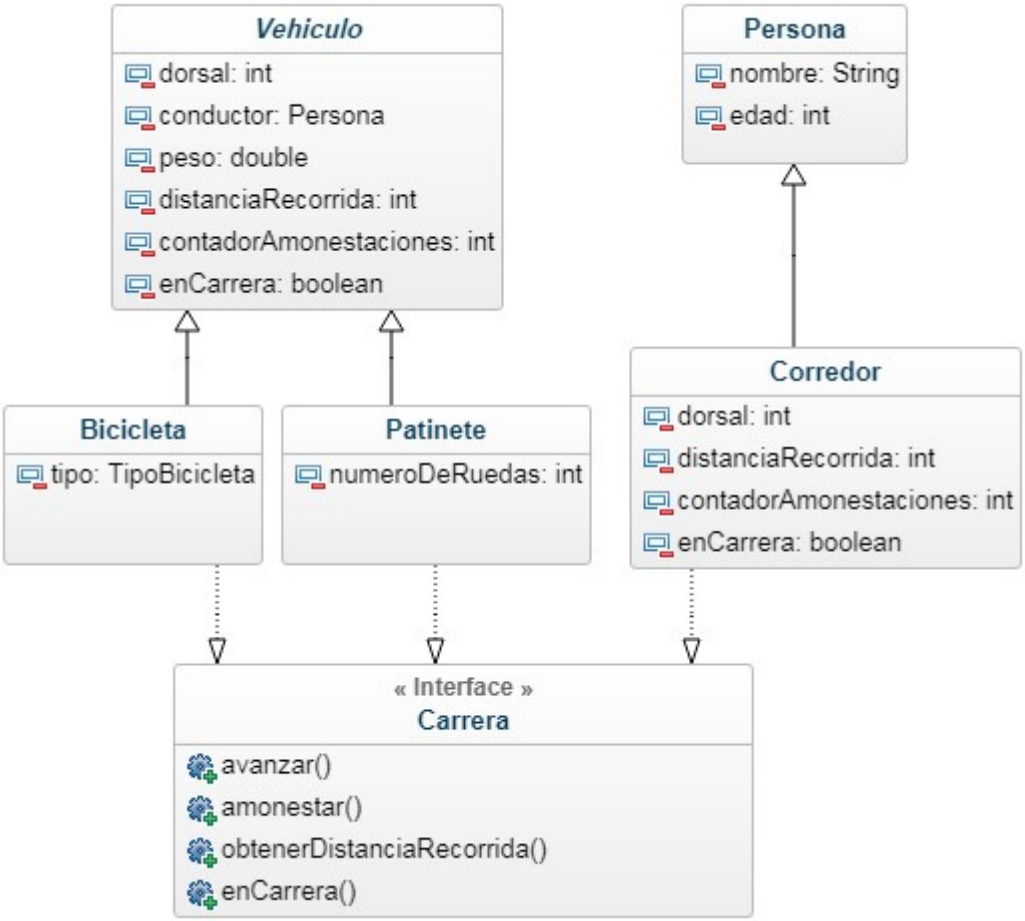
Crea un proyecto llamado CarreraPopular que simulará el desarrollo de una carrera. En la carrera solo pueden participar corredores, bicicletas y patinetes.

Mediante un menú se simulará el desarrollo de la carrera: mostrar posiciones, hacer avanzar a los participantes, amonestar, mostrar ganadores, etc.

Como estructura de datos se utilizará un **ArrayList** que contendrá todos los participantes en la carrera.

El diagrama de clases de la aplicación es el siguiente:



Clase Vehículo

Clase abstracta.

Contiene los atributos:

- dorsal: el dorsal será el número de orden en el que se ha dado de alta en la carrera.
- conductor: Contiene los datos del conductor del vehículo.
- peso: peso del vehículo en Kg.
- distanciaRecorrida: distancia que lleva recorrida el vehículo en la carrera.
- enCarrera: indica si el vehículo sigue o no en la carrera.
- contadorAmonestaciones: contiene el número de amonestaciones que ha recibido el participante durante la carrera.

Clase Bicicleta

Contiene el atributo:

- tipo: del tipo TipoBicicleta. TipoBicicleta es un **enum** con los valores {PASEO, MONTAÑA}

Clase Patinete

Contiene el atributo numeroDeRuedas: número de ruedas del Patinete.

Clase Persona

Los atributos de esta clase son:

- nombre: nombre del participante
- edad: edad del participante.

Clase Corredor

Los atributos de esta clase son:

- dorsal: será el número de orden en el que se ha dado de alta en la carrera.
- distanciaRecorrida: distancia recorrida por el participante en la carrera.
- enCarrera: indica si el participante sigue o no en la carrera.
- contadorAmonestaciones: contiene el número de amonestaciones que ha recibido el participante durante la carrera.

Todos los atributos de las clases son privados

Las clases contendrán los métodos que consideres necesarios: constructores, get/set, toString, etc.

Interface Carrera

Contiene los métodos abstractos:

avanzar()

amonestar()

obtenerDistanciaRecorrida()

enCarrera()

No se pueden añadir más métodos a la interface.

Método avanzar()

El método avanzar debe actuar de la siguiente forma:

Solo pueden avanzar los participantes que siguen en carrera.

Avanzar significa aumentar la distancia recorrida por el participante.

Para las bicicletas: si el peso es inferior a 10Kg se aumenta la distancia en 20 metros, en caso contrario se avanza 10 metros.

Para los patinetes: si el peso es inferior a 1Kg se aumenta la distancia en 5 metros, en caso contrario se avanza 3 metros.

Para los corredores: si el corredor es menor de 40 años se aumenta la distancia en 15 metros, en caso contrario se avanza 10 metros.

El método avanzar devuelve la distancia total recorrida por el participante hasta el momento.

El método devuelve -1 si el participante no sigue en la carrera.

Método amonestar()

El método amonestar debe actuar de la siguiente forma:

Solo se puede amonestar a los participantes que siguen en carrera.

Amonestar significa sumar una amonestación al participante.

Cuando un participante tiene dos amonestaciones queda eliminado de la carrera.

El método devuelve true si se ha realizado la amonestación o false si el participante ya no estaba en carrera.

Método obtenerDistanciaRecorrida()

Devuelve la distancia recorrida por el participante hasta este momento.

Método enCarrera()

Devuelve si el participante sigue o no en la carrera.

Clase Menu

El programa utilizará la siguiente clase Menu:

```
public class Menu {
    private int opcion;
    public void mostrar() {
        System.out.println("1. Mostrar carrera");
        System.out.println("2. Avanzar");
        System.out.println("3. Amonestar");
        System.out.println("4. Mostrar ganadores");
        System.out.println("5. Mostrar ganadores por categorías");
        System.out.println("6. Mostrar posición final del participante más joven");
        System.out.println("0. FIN");
    }
    public int leerOpcion() {
        Scanner sc = new Scanner(System.in);
        do {
            System.out.print("Introduzca opción: ");
            opcion = sc.nextInt();
        } while (opcion < 0 || opcion > 6);
        sc.nextLine(); //limpiar el intro
        return opcion;
    }
}
```

Clase Principal

Desde la clase principal se gestionarán las operaciones a realizar con los participantes en la carrera.

Lo primero que debe hacer el programa es crear la carrera mediante el método crearCarrera()). Una vez creada se muestra el menú y se ejecuta la opción elegida.

El programa termina cuando se introduce 0 como opción de menú.

Método crearCarrera()

Primero se pide la distancia total a recorrer.

Después se pide el número de participantes de la carrera.

A continuación se leen los datos de cada participante y se guardan en un **ArrayList**.

>> Los datos dorsal, distancia recorrida, número de amonestaciones y enCarrera **NO se leen**.

> Cuando se lee un participante el valor del dorsal es el número de orden, el campo *distanciaRecorrida* inicialmente será cero, el campo *contadorAmonestaciones* será cero y el campo *enCarrera* será true.

Un ejemplo de ejecución de este método puede ser:

```
Introduzca distancia (en metros y >=100): 2000   /////----> se lee por teclado
Introduzca número de participantes en la carrera: 5 ///----> se lee por teclado

Dorsal 1:
Tipo de participante 1->Bicicleta 2->Patinete 3-> Corredor --> 2
Nombre del conductor: Penelope Centella
Edad: 23
Peso del vehículo: 1
Número de ruedas: 4

Dorsal 2:
Tipo de participante 1->Bicicleta 2->Patinete 3-> Corredor --> 3
Nombre del corredor: Lucas Gervasio
edad: 22
```

```
Dorsal 3:
Tipo de participante 1->Bicicleta 2->Patinete 3-> Corredor --> 1
Nombre del conductor: Juan Reyes
Edad: 18
Peso del vehículo: 10
Tipo de bicicleta 1-> Paseo 2-> Montaña: 1

Dorsal 4:
Tipo de participante 1->Bicicleta 2->Patinete 3-> Corredor --> 3
Nombre del corredor: Felipe Juan
edad: 20

Dorsal 5:
Tipo de participante 1->Bicicleta 2->Patinete 3-> Corredor --> 1
Nombre del conductor: Aitor Montes
Edad: 19
Peso del vehículo: 15
Tipo de bicicleta 1-> Paseo 2-> Montaña: 2
```

Resto de métodos que se deben ejecutar correspondientes a cada opción del menú:

Opción 1: método mostrarCarrera() muestra todos los participantes ordenados por la posición que ocupan en la carrera, de mayor a menor. La primera posición será la del participante que haya recorrido más distancia hasta el momento y el último será el que haya recorrido menos distancia.

Un ejemplo de ejecución del método antes de empezar a avanzar es el siguiente. En este caso los participantes se muestran en el orden en que se han introducido inicialmente con la distancia a cero.

```
Situación de la carrera:
Patinete
Dorsal: 1
Peso: 1.0
Conductor: Penélope Centella
Edad: 23
Amonestaciones: 0
Número de ruedas 4
Distancia Recorrida: 0
------------------------
Corredor
Dorsal: 2
Nombre: Lucas Gervasio
Edad: 22
Amonestaciones: 0
Distancia Recorrida: 0
------------------------
Bicicleta
Dorsal: 3
Peso: 10.0
Conductor: Juan Reyes
Edad: 18
Amonestaciones: 0
Bicicleta de PASEO
Distancia Recorrida: 0
------------------------
Corredor
Dorsal: 4
Nombre: Felipe Juan
Edad: 20
Amonestaciones: 0
Distancia Recorrida: 0
------------------------
Bicicleta
Dorsal: 5
Peso: 15.0
Conductor: Aitor Montes
Edad: 19
```

```
Amonestaciones: 0
Bicicleta de MONTAÑA
Distancia Recorrida: 0
------------------------
```

Un ejemplo de ejecución del método mostrarCarrera() después de ejecutar varias veces la opción de avanzar es el siguiente. Se muestran ordenados de mayor a menor distancia.

```
Situación de la carrera:

Bicicleta
Dorsal: 3
Peso: 10.0
Conductor: Juan Reyes
Edad: 18
Amonestaciones: 0
Bicicleta de PASEO
Distancia Recorrida: 70
------------------------
Corredor
Dorsal: 4
Nombre: Felipe Juan
Edad: 20
Distancia Recorrida: 45
------------------------
Corredor
Dorsal: 2
Nombre: Lucas Gervasio
Edad: 22
Distancia Recorrida: 30
------------------------
Patinete
Dorsal: 1
Peso: 1.0
Conductor: Penélope Centella
Edad: 23
Amonestaciones: 0
Número de ruedas 4
Distancia Recorrida: 30
------------------------
Bicicleta
Dorsal: 5
Peso: 15.0
Conductor: Aitor Montes
Edad: 19
Amonestaciones: 0
Bicicleta de MONTAÑA
Distancia Recorrida: 20
------------------------
```

Opción 2: método avanzarParticipantes(). Solo se puede avanzar si la carrera no ha terminado. El método obtiene un número al azar correspondiente a una posición del array y avanza el participante correspondiente. El participante que ocupa esa posición debe continuar en carrera, si no es así se elige otro. **Si después de avanzar, el participante supera la distancia total de la carrera, la carrera finaliza**.

Opción 3: método amonestarParticipantes(). Solo se puede amonestar si la carrera no ha terminado. Se pide por teclado el dorsal del participante y se realiza la amonestación. Si el participante ya había sido eliminado se mostrará un mensaje indicándolo y se vuelve al menú. Es posible que la carrera se quede sin participantes por las amonestaciones. En ese caso la carrera finaliza.

Opción 4: método mostrarGanadores(). Solo se podrá ejecutar si ha acabado la carrera. Se mostrará un mensaje si la carrera aún no ha acabado. Se mostrarán los tres primeros puestos de la carrera. Si quedan menos de tres se mostrarán los que queden. Si no queda ninguno se mostrará un mensaje indicándolo.

Por ejemplo:

```
Primer Puesto:
Bicicleta
Dorsal: 5
Peso: 15.0
Conductor: Aitor Montes
Edad: 19
Amonestaciones: 1
Bicicleta de MONTAÑA
Distancia Recorrida: 2000

Segundo Puesto:
Corredor
Dorsal: 4
Nombre: Felipe Juan
Edad: 20
Amonestaciones: 0
Distancia: 1815

Tercer Puesto:
Bicicleta
Dorsal: 3
Peso: 10.0
Conductor: Juan Reyes
Edad: 18
Amonestaciones: 0
Bicicleta de PASEO
Distancia: 1190
```

Opción 5: método mostrarGanadoresPorCategorias() Solo se podrá ejecutar si ha acabado la carrera. Se mostrará un mensaje si la carrera aún no ha acabado. Se mostrarán el primero de cada una de las categorías. Si de alguna categoría no hay participantes se mostrará un mensaje. Ejemplo:

```
Primera Bicicleta en la posición 1
Bicicleta
Dorsal: 5
Peso: 15.0
Conductor: Aitor Montes
Edad: 19
Amonestaciones: 1
Bicicleta de MONTAÑA
Distancia Recorrida: 2000

Primer Corredor en la posición 2
Corredor
Dorsal: 4
Nombre: Felipe Juan
Edad: 20
Amonestaciones: 0
Distancia: 1815

Primer Patinete en la posición 5
Patinete
Dorsal: 1
Peso: 1.0
Conductor: Penelope Centella
Edad: 23
Amonestaciones: 1
Número de ruedas 4
Distancia: 872
```

Opción 6: método mostrarPosicionMasJoven() Solo se podrá ejecutar si ha acabado la carrera. Se mostrará un mensaje si la carrera aún no ha acabado. Este método mostrará la posición final en la que ha quedado el participante de menor edad. Si hay varios participantes que tienen esa misma edad se mostrará el que mejor ha terminado en la carrera.