Examen Parcial 2do corte POO

En una carrera ciclística, un **Equipo** está conformado por un conjunto de ciclistas (array) y se identifica por el nombre del equipo (tipo String), la suma de los tiempos de carrera de sus ciclistas en minutos y país del equipo. **Spoiler:** Sus atributos deben ser privados, pues esta clase no proporcionará atributo ni métodos a otra.

Un **Ciclista** es una clase que se describe con varios atributos: identificador (int), nombre del ciclista y tiempo acumulado de carrera (en minutos, con valor inicial cero). Un ciclista tiene un método imprimirTipo que devuelve un String.

Los ciclistas se clasifican de acuerdo con su especialidad (sus atributos deben ser privados y sus métodos protegidos). Estas especialidades heredan de la clase Ciclista y tienen los siguientes aspectos:

- **Velocista**: tiene nuevos atributos como potencia promedio (en vatios) y velocidad promedio en sprint (Km/h) (ambos de tipo double).
- **Escalador**: tiene nuevos atributos como aceleración promedio en subida (m/s2) y grado de rampa soportada (grados) (ambos de tipo double).
- **Contrarelojista**: tiene un nuevo atributo, velocidad máxima (km/h).

Definir clases y métodos para el ciclista y sus clases hijas para realizar las siguientes acciones:

- Constructores para cada clase (deben llamar a los constructores de la clase padre en las clases donde se requiera).
- Métodos get y set para cada atributo de cada clase.
- Método toString para imprimir los datos de un ciclista. Debe sobreescribir el método de la clase padre e imprimir los valores de los atributos propios.
- Método imprimirTipo que devuelve un String con el texto “Es un xxx”. Donde xxx es la clase a la que pertenece. Debe sobreescribir el método.

La clase Equipo debe tener los siguientes métodos:

- Métodos get y set para cada atributo de la clase.
- Imprimir los datos del equipo en pantalla. **Spoiler:** este método debe imprimir los datos del equipo y los nombres de los ciclistas que conforman el equipo.
- Añadir un ciclista a un equipo. **Spoiler:** este método debe pedir los datos del ciclista, preguntar de que tipo es y añadirlo al array de ciclistas.
- Calcular el total de tiempos de los ciclistas del equipo (suma de los tiempos de carrera de sus ciclistas). **Spoiler:** Este método será invocado en el método de imprimir los datos, guardándolo en el atributo suma de los tiempos de carrera.
- Listar los nombres de todos los ciclistas que conforman el equipo por tipo.
- Dado un identificador de un ciclista por teclado, es necesario imprimir en pantalla los datos del ciclista, incluyendo los datos específicos de cada subclase. Si no existe, debe aparecer el mensaje correspondiente. **Spoiler:** Utilizar el método toString.

En una clase de App, en un método main se deben crear **dos** equipos y un menú con opciones que se repita hasta que el usuario digite salir:

- Añadir ciclista a un equipo.
- Imprimir datos de un equipo.
- Listar nombres de los ciclistas de un equipo por tipo.
- Buscar ciclista en un equipo.
- salir

Nota: agregar ciclistas de diferentes tipos. De la clase padre ciclista no se creará ninguno.

SUPER Spoiler:

Para Hacer un array dinámico utilizamos la clase Vector, importada desde la librería:

```
import java.util.*;
```

Luego el atributo se define:

```
private Vector array;
```

En el constructor:

```
this.array= new Vector();
```

Para agregar elementos al vector:

```
array.add(objeto)
```

Para acceder a los elementos:

```
for(int i=0; i<array.size(); i++){
    ((ClasePadre) array.elementAt(i));}
```

Nota: como en este array no se le especifica que tipo de objetos contendrá, debemos hacer un cast (parte verde) hacia el tipo de estos objetos, en este caso la clase padre que es la general para todos.