• Renomear a clase Circulo por Circulito.

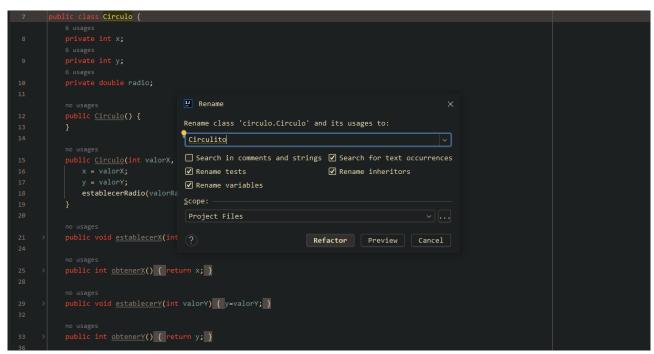


Figura 1: Renomear clase

• Renomear o método ObtenerArea por ObtenerAreaCirculo.

public void establ radio=(valorRa	4 usages	() [(2001)]]		
<pre> }</pre>		! Rename		
2 usages public double obte ? Refactor Preview Cancel	radio=(valorRa	Rename method 'obtenerAreaCirculo() of class	ss circulo.Circulito' and its usage	es to:
public double obte 2 usages Project Files Public double obte ? Refactor Preview Cancel	}	obtenerAreaCirculo		
2 usages Project Files public double obte ? Refactor Preview Cancel		☐ Search in comments and strings	☐ Search for text occurrences	
public double obte ? Refactor Preview Cancel 1 usage	public double obte	<u>S</u> cope:		
Refactor Preview Cancel 1 usage		Project Files		
	public double obte	?	Refactor Preview Car	ncel
		enCincumfenencia() [neturn Math DI * ehten	onDiametro(): 1	

Figura 2: Renomear método

• Renomear os campo x e y por coordenadaX e coordenadaY.

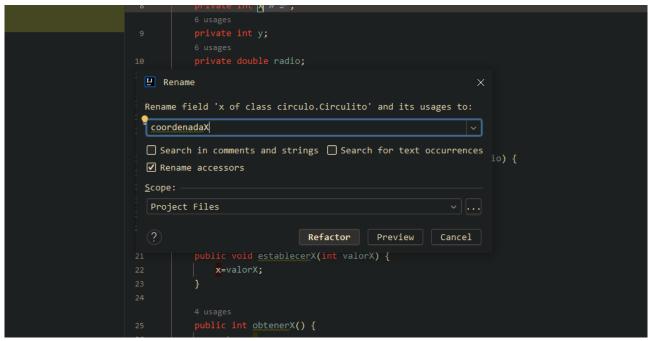


Figura 3: Renomear campos 1

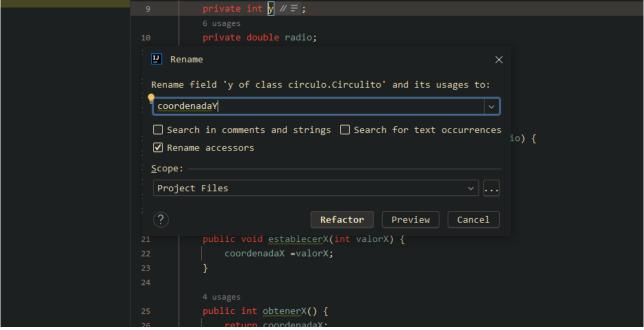


Figura 4: Renomear campos 2

• Introducir constante LIMITERADIO de tipo double co valor 0.0

```
public class Circulito {

public static final double DOUBLE = 0d;
6 usages
private int coordenadaX;

public static final double DOUBLE = 0d;
Replace all occurrences
Move to another class
no usages

double LIMITERADIO = DOUBLE;
4 usages
public Circulito() {

14
}

7 usages
```

Figura 5: Engadir nova constante (Ctrl + Alt + c)

• Cambiar parámetros do método trasladarCentro para que teña dous parámetros trasladarX e trasladarY de tipo int. Facer os cambios necesarios para que o código do método permita engadir á coordenada x o valor de trasladarX e o engadir á coordenada y o valor de trasladarY.

```
Find Conflicts ×

Find Conflicts ×

Find Conflicts ×

Find Conflicts in 1 result

Value read 1 result

Value read
```

Figura 6: Cambiar parámetros

• Encapsular os tres atributos da clase: coordenadaX coordenadaY, radio.

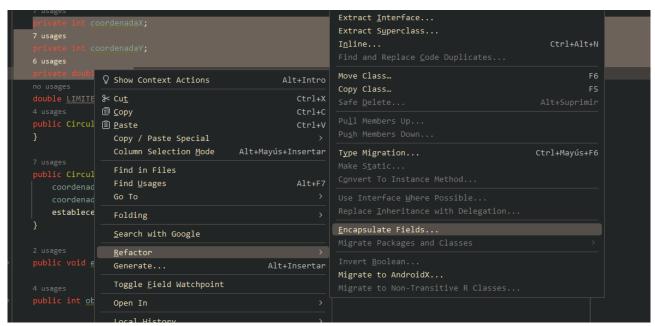


Figura 7: Encapsular atributos

• Eliminar de forma segura os métodos obtenerX, obtenerY, obtenerRadio, establecerX, establecerY e establecerRadio que agora son innecesarios facendo os cambios necesarios no código da clase, da clase Main e das probas para que sexan substituídos polos correspondentes métodos tipo get e set creados.

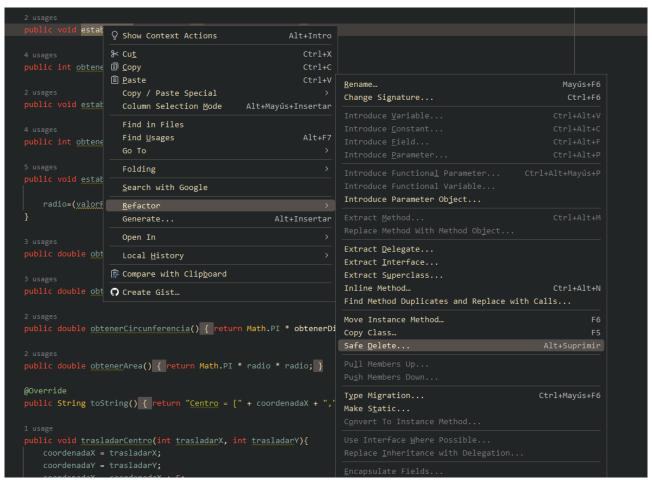


Figura 8: Eliminación de forma segura

• Ao rematar comproba que as probas seguen funcionando (achega captura de pantalla co resultado)

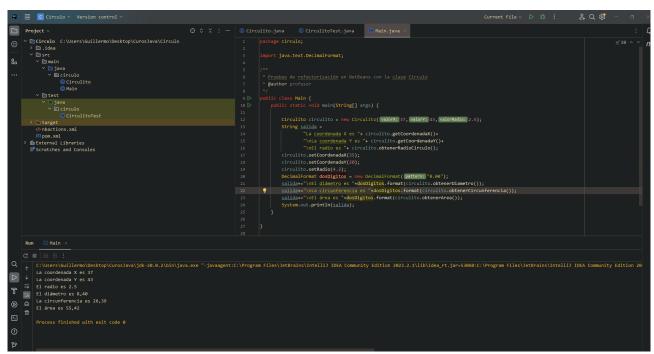


Figura 9: Pruebas (clase Main)

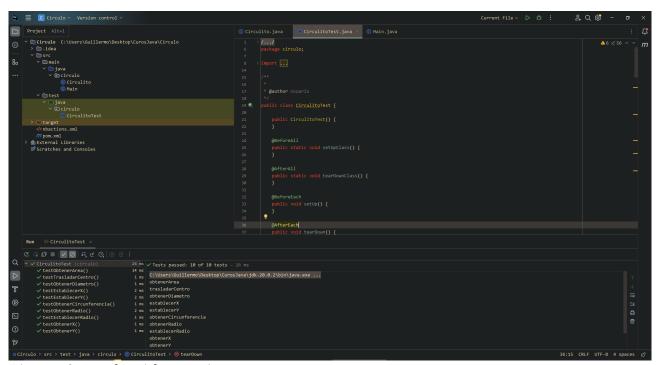


Figura 10: Pruebas (clase Test)

Realizar as seguintes reestruturacións sobre o proxecto Alumno, modificando se é o caso, os tests implementados.

• Move o atributo 'telefono' da clase Alumno a súa superclase Persona.

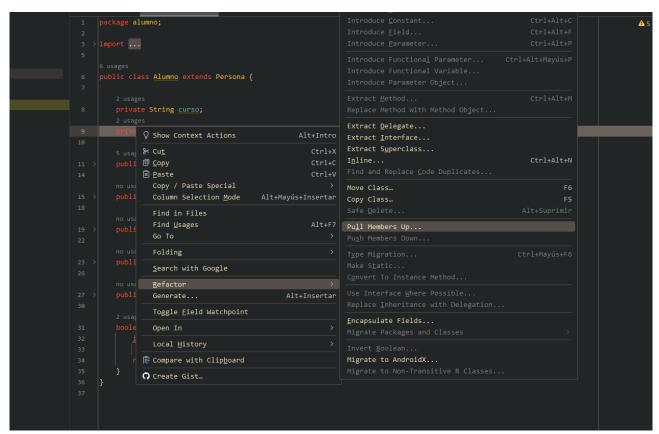


Figura 11: Mover atributos

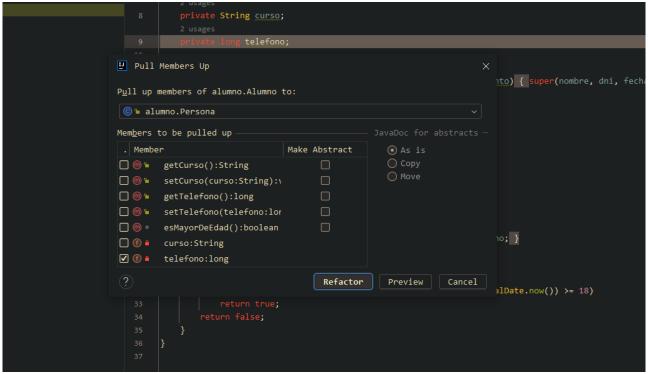


Figura 12: Mover atributos 2

```
package alumno;

import java.time.LocalDate;

yusages 1 inheritor

public class Persona {

2 usages
protected long telefono;
4 usages
private String nombre;
```

Figura 13: Mover atributos 3

• Move o método 'esMayorDeEdad' da clase Alumno a súa superclase Persona.

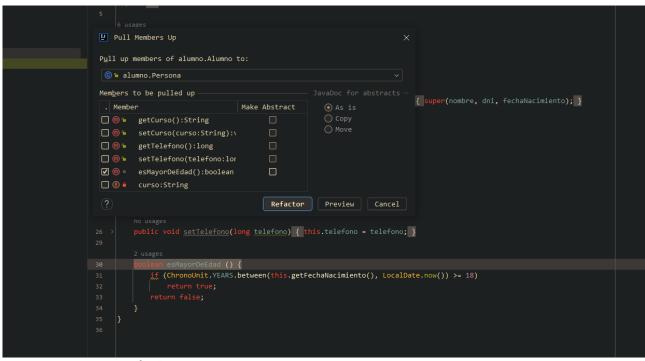


Figura 14: Mover métodos

Figura 15: Mover métodos 2

• Na clase 'Persona' substituir o número 18 por unha constante estática.

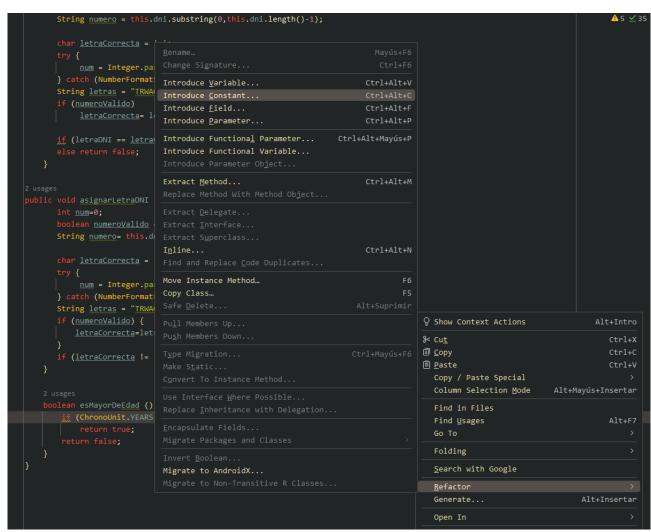


Figura 16: Crear constante

Figura 17: Crear constante 2

• Nos métodos comprobarDNI e asignarLetraDNI repítese o código. Crea un método privado con ese código que sexa invocado desde os outro dous métodos.

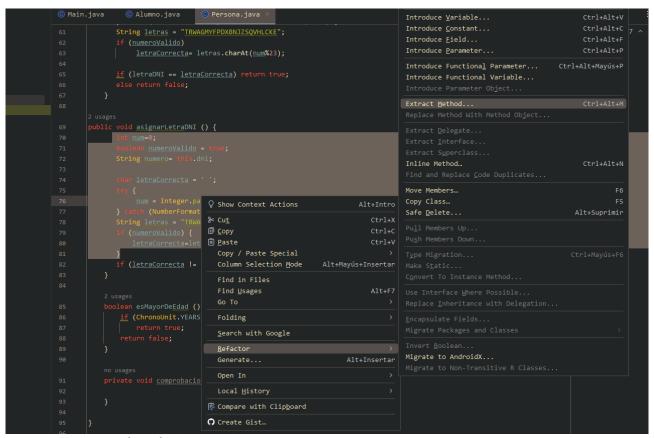


Figura 18: Creación método privado

```
gublic void asignarLetraD

char letraCorrecta = getLetraCorrecta();

if (letraCorrecta != ' ') this.dni+=letraCorrecta;
}

1 usage
private char getLetraCorrecta() {
   int num=0;
   boolean numeroValido = true;
   String numero= this.dni;

   char letraCorrecta = ' ';
   try {
        num = Integer.parseInt(numero);
   } catch (NumberFormatException numberFormatException) { numeroValido = false;}

String letras = "IRWAGMYFPDXBNJZSQVHLCKE";
   if (numeroValido) {
        letraCorrecta = letras.charAt(num%23);
   }
   return letraCorrecta;
}
```

Figura 19: Creación método privado 2

• Na clase Main, converte a clase anónima que está como segundo parámetro do método Collections.sort nunha clase membro.

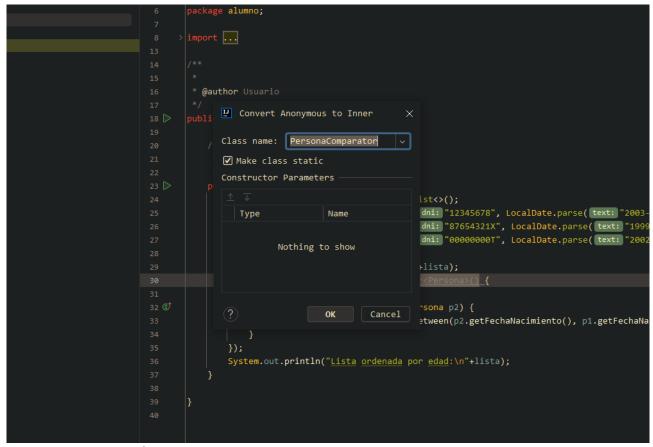


Figura 20: Clase anónima a clase membro

Realizar as seguintes reestruturacións sobre o proxecto MTB.

• Move o atributo marcha, o método getMarcha e o método setMarcha dende a superclase Bicicleta á subclase MTB ca opción de menú Reestructurar/Descender probablemente quede algún problema que solucionar e non se faga todo automaticamente soluciona e explica dito problema.

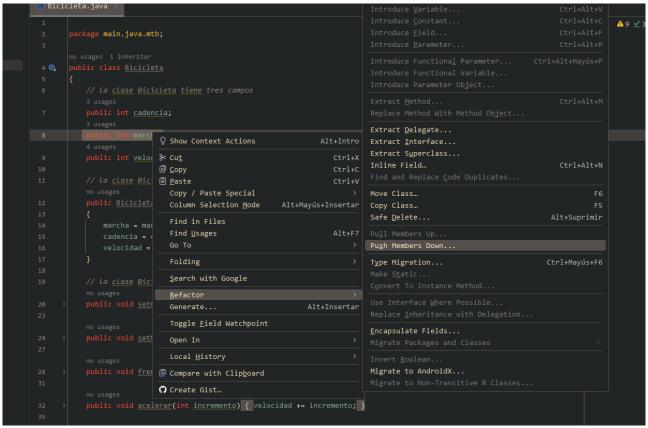


Figura 21: Mover clases e atributos á subclase

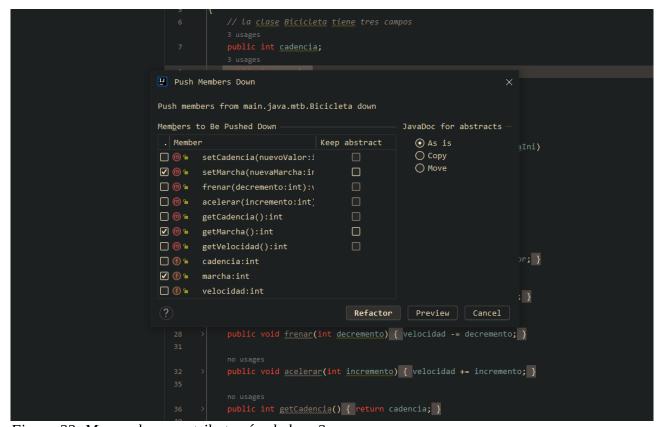


Figura 22: Mover clases e atributos á subclase 2

El problema que se produce es que la clase "Bicicleta" utiliza el atributo marcha por lo que, si lo cambiamos de clase, el constructor se queda sin este atributo.

• Extrae un interface para os métodos getAltoAsiento e setAltoAsiento, usando a opción de menú Reestructurar/Extraer Interface.

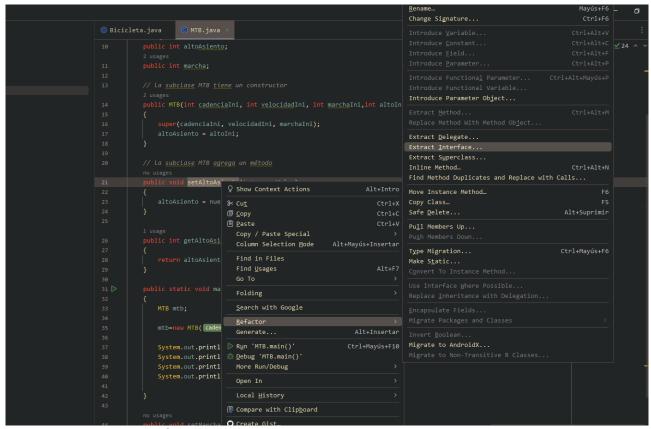


Figura 23: Extraer interfaz

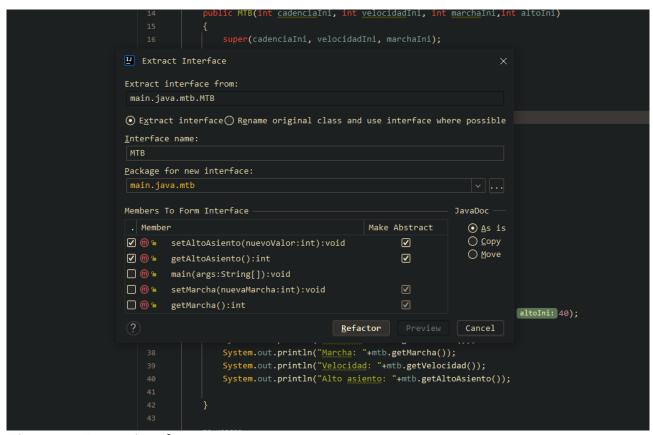


Figura 24: Extraer interfaz 2

• Extrae unha superclase co campo velocidad e os métodos getVelocidad, acelerar e frenar usando a opción de menú Reestructurar/Extraer Superclase.

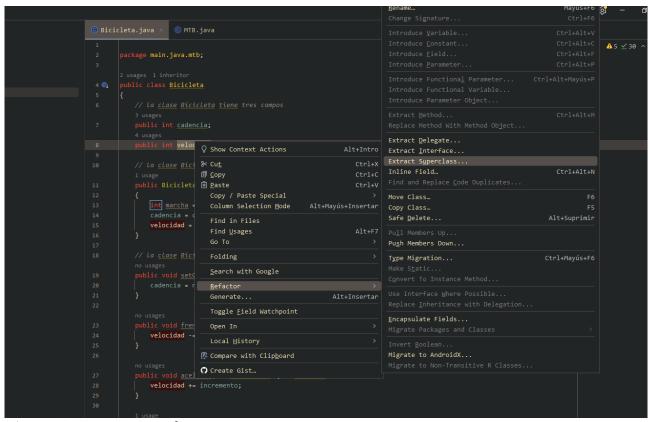


Figura 25: Extraer superclase

5	1			
	// la clase Bicicleto	a tiene tres campo		
<u>□</u> Extra	3 usages		×	
EXT.a	ct Superclass		*	
Extract s	uperclass from:			
main.ja	a.mtb.Bicicleta			
⊙ Extrac	t superclass○ Rename origina	al class and use s	superclass where possible	ni)
<u>S</u> uperclas	s name:			11)
Biciclet	a			
<u>P</u> ackage f	or new superclass:			
main.jav	ra.mtb			
Members T	o Form Superclass		- JavaDoc for abstracts —	
. Membe		Make Abstract	⊙ As is	
■ ⑩ •	setCadencia(nuevoValor:int):		О Сору	
✓ @ •	frenar(decremento:int):void		○ Move	
☑ ⑩ •	acelerar(incremento:int):voi			
□ ⑩ ኈ	<pre>getCadencia():int</pre>			
⊘ ⊚ •	getVelocidad():int			
□ ⑥ •	cadencia:int			
✓ (f) •	velocidad:int			
		Refactor	Preview Cancel	
	<pre>public void acelerar velocidad += inco } 1 usage public int getCadence {</pre>	remento;	(//acelerar	
	return cadencia:			

Figura 26: Extraer superclase 2

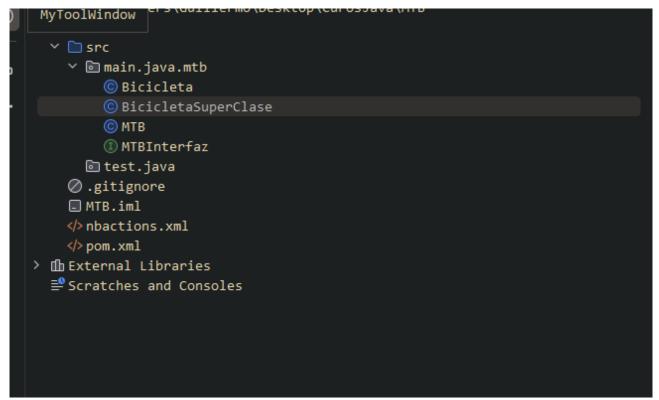


Figura 27: Resultado