Ejercicio 4 – El pactómetro

Dado el elevado número de elecciones que se están convocando en los últimos años, desde el departamento de Lenguajes y Sistemas Informáticos, se desea desarrollar un algoritmo que facilite el cálculo de los posibles pactos electorales (en caso de que fueran necesarios).

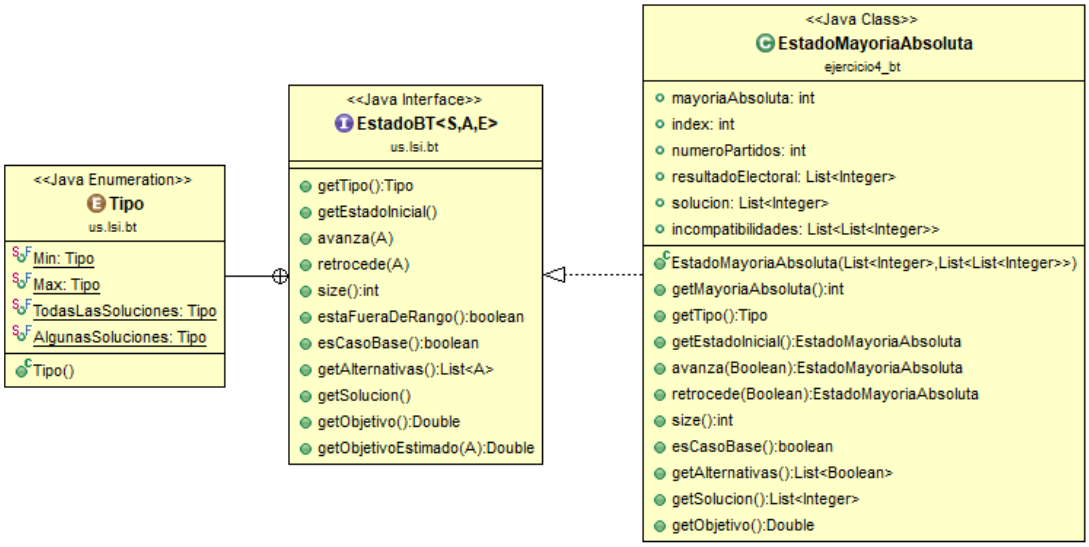
Se tiene una lista (**resultadoElectoral**) en la que cada elemento, representa el número de escaños obtenidos por cada partido en las últimas elecciones. Además, se tiene una lista de incompatibilidades (**incompatibilidades**), en la que cada elemento i-ésimo se corresponde a un partido y cada valor que toma, es una lista enteros que representa los partidos con los que no pueden pactar. La lista vacía, indica que el partido i-ésimo correspondiente no tiene incompatibilidades.

Se desea obtener la mejor coalición posible **(solucion)**, minimizando el número de partidos (**numeroPartidos**), cumpliendo las restricciones de incompatibilidad (**incompatibilidades**) y, además, consiguiendo la mayoría absoluta (**mayoriaAbsoluta**, mitad de los escaños totales más uno).

```
Ejemplo:
Resultado electoral = [120, 89, 52, 35, 13, 10, 8, 6, 5, 3, 2, 2, 2, 1, 1, 1]
Incompatibilidades = [[1, 2],[0, 3],[0, 3],[1, 2], [], [], [], [], [], [], [], [],
[], [], [], []]
Solución 1 = [0, 3, 4, 6]
Solución 2 = [0, 3, 4, 5]
```

SE PIDE:
Implementar los siguientes métodos de una solución basada en Backtracking:

a) Tipo getTipo()
b) EstadoMayoriaAbsoluta getEstadoInicial()
c) EstadoMayoriaAbsoluta avanza(Boolean a)
d) EstadoMayoriaAbsoluta retrocede(Boolean a)
e) boolean esCasoBase()
f) List<Boolean> getAlternativas()
g) Double getObjetivo()



Solución

Apartado a)

```
public Tipo getTipo() {
	return Tipo.Min;
}
```

Apartado b)

```
public EstadoMayoriaAbsoluta getEstadoInicial() {
	return new EstadoMayoriaAbsoluta(this.resultadoElectoral,
			this.incompatibilidades);
}
```

Apartado c)

```
public EstadoMayoriaAbsoluta avanza(Boolean a) {
	if (a) {
		Integer val = this.resultadoElectoral.get(this.index);
		this.mayoriaAbsoluta -= val;
		this.solucion.add(index);
		this.numeroPartidos++;
	}
	this.index += 1;
	return this;
}
```

Apartado d)

```
public EstadoMayoriaAbsoluta retrocede(Boolean a) {
	this.index -= 1;
	if (a) {
		Integer val = this.resultadoElectoral.get(this.index);
		this.mayoriaAbsoluta += val;
		this.solucion.remove(this.solucion.size() - 1);
		this.numeroPartidos--;
	}
	return this;
}
```

Apartado e)

```
public boolean esCasoBase() {
	return (this.index == this.resultadoElectoral.size())
			|| (this.mayoriaAbsoluta <= 0);
}
```

Apartado f)

```
public List<Boolean> getAlternativas() {
	List<Integer> inc = this.incompatibilidades.get(index);
	if (inc != null) {
		for (Integer i : inc) {
			if (this.solucion.contains(i)) {
				return Arrays.asList(false);
			}
		}
	}
	return Arrays.asList(true, false);
}
```

Apartado f)

```
public Double getObjetivo() {
	return (double) this.numeroPartidos;
}
```