```
import java.util.ArrayList;

public class Ejercicio1Examen {

	public static void recorreIzquierda(int [] arreglo, int total, int pos) {
		int i;

		for(i = pos; i < total; i ++)
			arreglo[i] = arreglo[i + 1];
	}

	public static  ArrayList <Integer> SiSeEncuentraNvecesEnElArregloSeQuita(int [] arreglo,
int total, int n) {

		ArrayList <Integer> listaArr = new ArrayList(); // te faltan las '<>' en el new.
		int contDatos = 0;
		Integer entero = new Integer(5); // nunca usas este entero

		//Bien! Aunque creo que es mejor usar "i < total",  así puedes generalizar a
	aplicar tu método a una sección que pida el usuario y no necesariamente todo el arreglo
		for(int i = 0; i < arreglo.length; i ++)
			if(arreglo[i] == n)
				contDatos += 1;

		// la idea de recorrer funciona, pero puede ser muy tardada. Al final, solo quieres
meter los datos al nuevo arreglo que vas a devolver. Tú sabes que el arreglo inicial solo va a
quedar con puros elementos que sean n, y son exactamente n de ellos. Así que una alternativa
es solo al final hacer total = n, y llenar esos n elementos del número n.
		if(contDatos == n ) {
			for(int i = 0; i < arreglo.length; i ++)
				if(arreglo[i] == n) {
					recorreIzquierda(arreglo, total, i); //recorre a la izquierda
en pos i
					total = i - 1;
					// ¿Por qué 5? Creo que no hay nada que agregársele a la
listaArr cuando el valor es n
					listaArr.add(5);
				}
				// te falta agregar el número a la listaArr en el else
		}

		else
			listaArr = null;
```

```
        return listaArr;
}



public static void main(String[] args) {
        // TODO Auto-generated method stub
        int [] arreglo1 = new int[5];
        int [] arreglo2 = new int[5];
        int [] arreglo3 = new int[3];
        int [] arreglo4 = new int[3];
        int [] arreglo5 = new int[4];

        arreglo1[1] = 2;
        arreglo1[2] = 8;
        arreglo1[3] = 6;
        arreglo1[4] = 2;
        arreglo1[5] = 7;

        arreglo2[1] = 2;
        arreglo2[2] = 8;
        arreglo2[3] = 6;
        arreglo2[4] = 9;
        arreglo2[5] = 1;

        arreglo3[1] = 2;
        arreglo3[2] = 2;
        arreglo3[3] = 2;

        arreglo4[1] = 4;
        arreglo4[2] = 1;
        arreglo4[3] = 4;

        arreglo5[1] = 4;
        arreglo5[2] = 4;
        arreglo5[3] = 4;
        arreglo5[4] = 4;

        System.out.println(SiSeEncuentraNvecesEnElArregloSeQuita(arreglo1, 5, 2));
        System.out.println(SiSeEncuentraNvecesEnElArregloSeQuita(arreglo2, 5, 2));
        System.out.println(SiSeEncuentraNvecesEnElArregloSeQuita(arreglo3, 3, 2));
        System.out.println(SiSeEncuentraNvecesEnElArregloSeQuita(arreglo4, 3, 2));
        System.out.println(SiSeEncuentraNvecesEnElArregloSeQuita(arreglo5, 4, 4));
```

```
        }

}
```