✅

Ejercicio_12 REFACTORIZADO

- La bandeja puede ser de tipo Bandeja, no solo para guardar la cantidad de hojas sino también para hacer las operaciones pertinentes (validar que haya suficientes para imprimir cierto documento, por ejemplo).
- Lo mismo el nivel de tinta, que debería estar dentro de un Cartucho (porque es monocromática, sino habría un array de ellos)
  - El chequeo de la tinta y el descuento cuando se imprime lo debe hacer el cartucho.
- Por último, si vas a usar excepciones que sean de tipo `'RuntimeException'`. El funcionamiento es el mismo, pero te vas a ahorrar de poner el 'throws' por todos lados.

```
package ejercicio_12;

public class ImpresoraMonocromatica {
    private boolean estaEncendida;
    private Bandeja bandeja;
    private Cartucho cartucho;

    //CONSTRUCTOR
    public ImpresoraMonocromatica() {
        this.estaEncendida=false;
        this.bandeja = new Bandeja(0); // la bandeja empieza con 0 hojas
        this.cartucho = new Cartucho(100);
    }
    //GETTERS
    public boolean isEstaEncendida() {
        return estaEncendida;
    }
    //SETTERS
    public void setEstaEncendida(boolean estaEncendida) {
        this.estaEncendida = estaEncendida;
    }
    public void recargarBandeja(int cant){
        bandeja.recargar(cant); // TELL, DON'T ASK
    }
    // METODOS
    public void estadoImpresora(){
        System.out.println(toString());
    }

    //se consume 1 punto de tinta cada 50 caracteres
    public int nivelDeTintaPara(int cantCaract){
        return cartucho.chequeoPara(cantCaract);
    }

    public void imprimir(Documento doc1) throws Exception {
        if(verificacion(doc1)==1){
            restarTinta(doc1);
            restarPapel(doc1);
            // Imprime la fecha
            System.out.print("\n"+doc1.getFecha() + "\t");
            // Imprime el titulo
            System.out.println(doc1.getTitulo());
            // Imprime el cuerpo
            System.out.println(doc1.getCuerpo());
        }
    }

    private void restarTinta(Documento doc1){
        int cantCaract = doc1.getCuerpo().length();
        cartucho.restar(cantCaract);
    }

    private void restarPapel(Documento doc1){
        int cantCaract = doc1.getCuerpo().length();
        bandeja.restar(cantCaract);
    }

    private int verificacion(Documento doc1) throws Exception {
        int cantCaract = doc1.getCuerpo().length();
        if (estaEncendida == true) {
            if (cartucho.getNivelTinta() >= nivelDeTintaPara(cantCaract)) {
                if (bandeja.getCantHojas() > 0) {
                    return 1;
                } else {
                    throw new RuntimeException("La bandeja de la impresora está vacía.");
                }
            } else {
                throw new RuntimeException("El nivel de tinta no es suficiente para imprimir este documento.");
            }
        } else {
            throw new RuntimeException("La impresora está apagada.");
        }
    }

    @Override
    public String toString() {
        return "ImpresoraMonocromatica{" + "estaEncendida=" + estaEncendida + ", bandeja=" + bandeja.getCantHojas() + ", nivelTinta=" + cartucho.getNivelTinta() + '}';
    }
}
```

```
package ejercicio_12;

class Cartucho {
    private int nivelTinta;

    //CONSTRUCTOR
    public Cartucho(int nivelTinta) {
```

```
        this.nivelTinta = nivelTinta;
    }

    //GETTERS y SETTERS

    public int getNivelTinta() {
        return nivelTinta;
    }

    public void setNivelTinta(int nivelTinta) {
        this.nivelTinta = nivelTinta;
    }

    // OTROS METODOS

    public void restar( int cantCaract){
        int tintaARestar = getNivelTinta()- chequeoPara(cantCaract);
        setNivelTinta(tintaARestar);
    }

    public int chequeoPara(int cantCaract) {
        return cantCaract/50;
    }

}
```

```
package ejercicio_12;

public class Bandeja {
    private int cantHojas;

    //CONSTRUCTOR

    public Bandeja(int cantHojas) {
        this.cantHojas = cantHojas;
    }

    //GETTERS y SETTERS

    public int getCantHojas() {
        return cantHojas;
    }

    public void setCantHojas(int cantHojas) {
        this.cantHojas = cantHojas;
    }

    // OTROS METODOS
    public void restar(int cantCaract){
        int papelARestar = getCantHojas()-(cantCaract/20);
        setCantHojas(papelARestar);
    }

    public void recargar(int cant){
           if(getCantHojas()<=35 && cant > 0 && cant <= 35){
           setCantHojas((cant+getCantHojas()));
        }
    }
}
```

```
package ejercicio_12;

import java.time.LocalDate;

public class Principal {

    public static void main(String[] args) throws Exception {
        ImpresoraMonocromatica impresora= new ImpresoraMonocromatica();
        //veo el estado de la impresora
        impresora.estadoImpresora();
        //prendo impresora
        impresora.setEstaEncendida(true);
        //cargo papel
        impresora.recargarBandeja(35); // recargo hojas al tope
        //veo el estado de la impresora (post encendido + recarga)
        impresora.estadoImpresora();

        // verificar cant de tinta según cant de caracteres
        System.out.println("nivel de tinta que deberia usarse " + impresora.nivelDeTintaPara(100));

        // creo el documento a imprimir
        String titulo = "Messi Goat";
        String campo = "Messi es el mejor futbolista de todos los tiempos.";
        Documento doc1= new Documento(LocalDate.now(),titulo,campo);
        impresora.imprimir(doc1);

        //veo el estado de la impresora (post impresion)
        impresora.estadoImpresora();
    }
}
```

Cosas que aprendí:

- No hacer que la clase `ImpresoraMonocromatica` sea un **objeto todopoderoso**
    - En programación orientada a objetos, un **objeto todopoderoso** (en inglés ***God Object***) es un objeto que *conoce demasiado* o *hace demasiado*. El objeto todopoderoso es un ejemplo de un anti-patrón.

    - Esto lo logré aplicando el termino TELL, DON'T ASK donde delego a otras clases (modulo) los métodos. Por ejemplo en el caso de la `Bandeja` y el `Cartucho`.

- **uso de `'RuntimeException'` en Java**
    - Armaré un resumen de eso.