Programación 2 - TUDAI

Adicional 2

Farmacia

Adicional 2

Farmacia

Una farmacia desea organizar sus medicamentos. De cada uno de ellos guarda, el nombre, laboratorio, precio, un conjunto de síntomas que trata.

La farmacia desea poder;

- Buscar todos los Medicamentos del laboratorio “Bayer”
- Buscar todos los medicamentos del laboratorio “Raffo”
- Buscar todos los medicamentos que en el nombre tenga la palabra “ina”
- Buscar todos los medicamento de precio menor a 1500
- Buscar todos los medicamentos que traten “Congestión Nasal”

Los anteriores son solo algunos ejemplos de las búsquedas que se desea proveer en la farmacia. Es posible que surjan nuevos tipos de busqueda asi como combinaciones lógicas de los anteriores, por ejemplo buscar todos los medicamentos del laboratorio “Bayer” y que el nombre contenga la palabra “ina”, o buscar todos los medicamentos que traten “Congestión” y cueste menos de 400 pesos

Tip: Leer el enunciado completo la primer vez así podemos ver que nos espera o nos piden

Aclaración: En este caso lo vamos a resolver en partes para que se vea que cosas nos guían a la solución final. Poder ver la evolución de la resolución

Farmacia - modelo básico

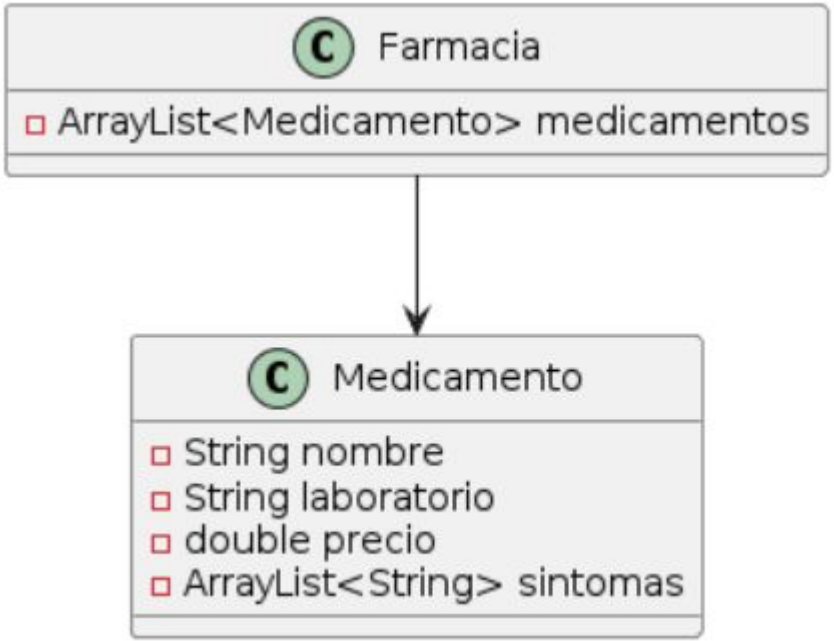
Reconocer las clases

Farmacia

Una farmacia desea organizar sus medicamentos. De cada uno de ellos guarda, el nombre, laboratorio, precio, un conjunto de síntomas que trata.

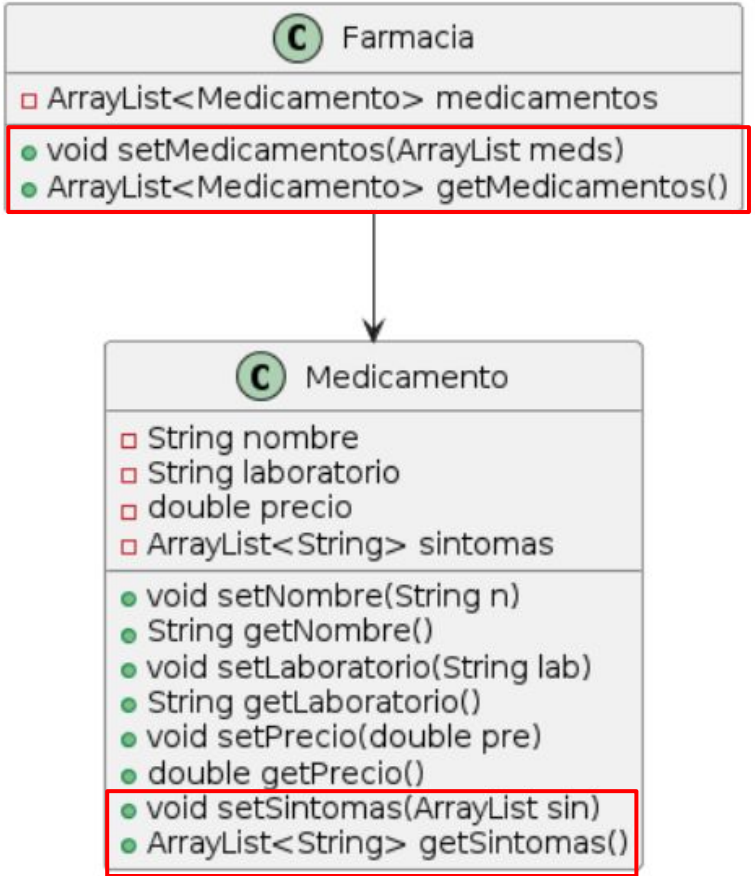
Reconocer los atributos

Farmacia - modelo básico



Clases y Atributos

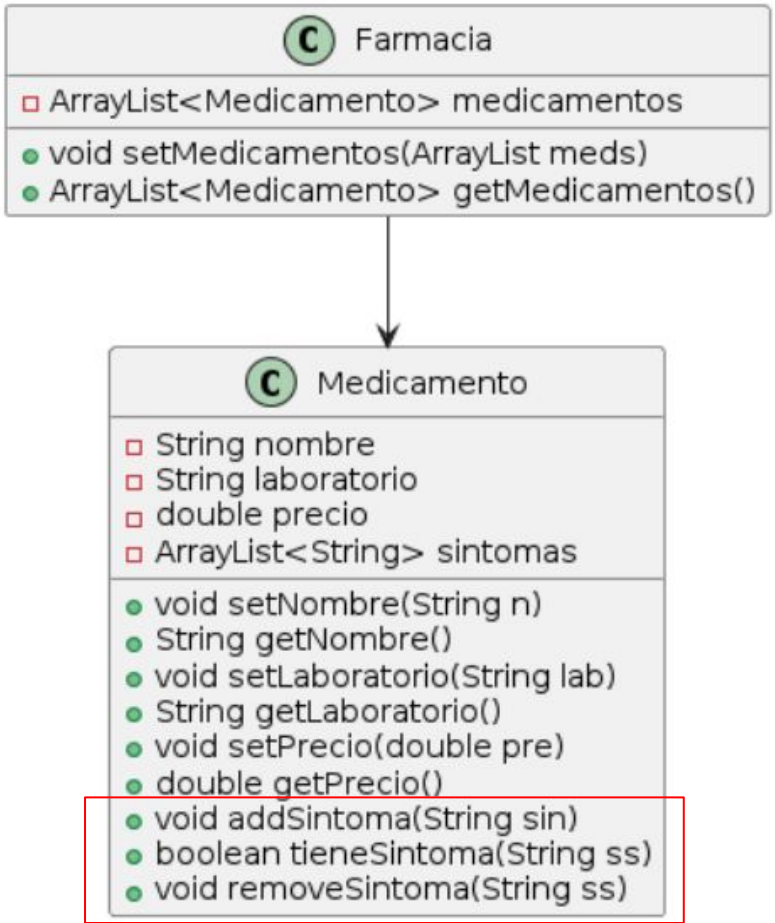
Farmacia - modelo básico



Agregamos getters y setters

Posible rotura de encapsulamiento

Posible mala delegación de responsabilidades

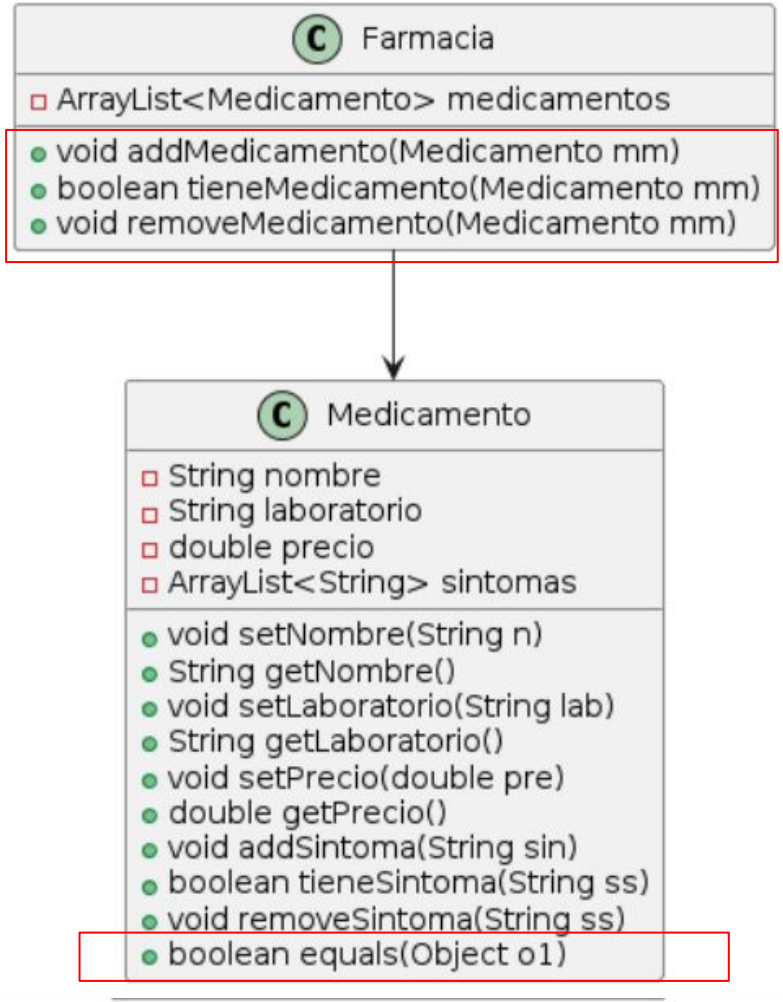


CUIDADO
repetidos, contiene o remove requieren que el objeto parámetro implemente el equals

El parámetro es String, ya lo implementa

Cambio el get y set del ArrayList por el agregado de a 1 elemento, el contiene y él remove.

- Se mantiene control
- Se puede controlar repetidos



Repito lo mismo para medicamentos en Farmacia

Ahora trabajo con Medicamento, debo implementar el equals

Farmacia - Servicios

La farmacia desea poder;

- Buscar todos los Medicamentos del laboratorio “Bayer”

Funcionalidad en la clase Farmacia y busca medicamentos

Funciona? Si

Es Orientada a Objetos? No

```
public ArrayList<Medicamento> buscarBayer(){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getLaboratorio().equals("Bayer")) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

```java
public ArrayList<Medicamento> buscarBayer(){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getLaboratorio().equals("Bayer")) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Esta bien que cree un método por cada laboratorio que quiera buscar?

- Buscar todos los medicamentos del laboratorio "Raffo"

Constante en Código

```java
public ArrayList<Medicamento> buscarRaffo(){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getLaboratorio().equals("Raffo")) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Que limita la solución?

Farmacia - Búsqueda por Laboratorio

```
public ArrayList<Medicamento> buscarLaboratorio(String nombreLaboratorio){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getLaboratorio().equals(nombreLaboratorio)) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Es el usuario del método el que decide qué laboratorio quiere buscar

Farmacia - Búsqueda por nombre

- Buscar todos los medicamentos que en el nombre tenga la palabra “ina”

Ya aprendimos, **y no usamos constantes en código**, la palabra que se va a buscar en el nombre es un parámetro de la búsqueda

```
public ArrayList<Medicamento> buscarNombre(String palabra){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getNombre().contains(palabra)) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Si el nombre contiene la palabra que nos pasaron

Farmacia - Búsqueda por precio

- Buscar todos los medicamento de precio menor a 1500

```
public ArrayList<Medicamento> buscarPrecio(double valor){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).getPrecio()< valor) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Si el precio es menor que el valor que nos pasaron

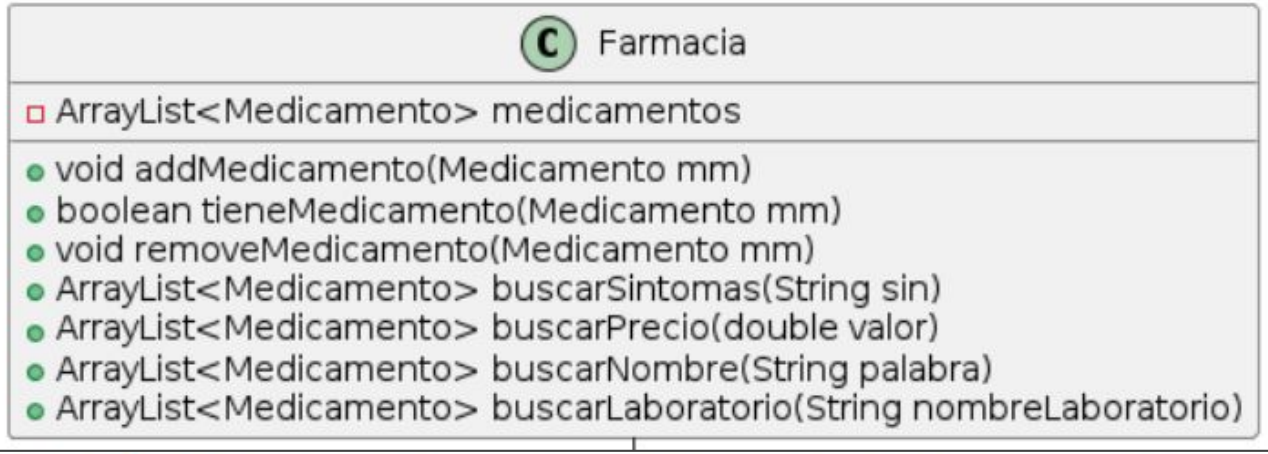
Farmacia - Tratan síntomas

- Buscar todos los medicamentos que traten “Congestión Nasal”

```
public ArrayList<Medicamento> buscarSintomas(String sin){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).tieneSintoma(sin)) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Si tiene el sintoma buscado

Farmacia -




```
public ArrayList<Medicamento> buscarSintomas(String sin){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).tieneSintoma(sin)) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Farmacia - Búsquedas

Agregó métodos por cada cosa que se me ocurre

...os ejemplos de las búsquedas que se desea proveer

medicamentos de
buscar todos los
pesos

ArrayList<Medicamento> medicamentos

- void addMedicamento(Medicamento mm)
- boolean tieneMedicamento(Medicamento mm)
- void removeMedicamento(Medicamento mm)
- ArrayList<Medicamento> buscarSintomas(String sin)
- ArrayList<Medicamento> buscarPrecio(double valor)
- ArrayList<Medicamento> buscarNombre(String palabra)
- ArrayList<Medicamento> buscarLaboratorio(String nombreLaboratorio)
- ArrayList<Medicamento> buscarSintomasYPrecio(String sin, double pre)
- ArrayList<Medicamento> buscarSintomasOPrecio(String sin, double pre)
- ArrayList<Medicamento> buscarSintomasYNombre(String sin,String nom)
- ArrayList<Medicamento> buscarSintomasYLaboratorio(String sin,String lab)

¿Que tenían en común los métodos?

```
public ArrayList<Medicamento> buscarSintomas(String sin){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (medicamentos.get(i).tieneSintoma(sin)) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
```

No puedo tener parámetros variados

La estructura es la misma en todos los métodos. Cambia la pregunta que le hago al Medicamento

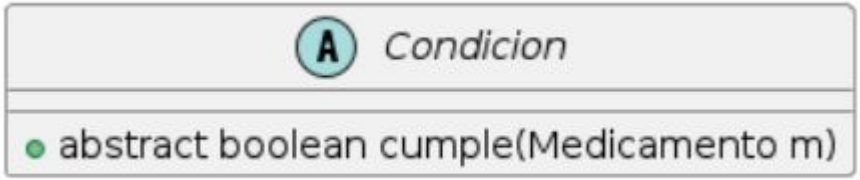
Qué condición debe cumplir para que lo agregue o no al resultado

Encapsular la condición

- Encapsular el comportamiento de preguntarle a un Medicamento si debe o no estar en el resultado
- Creó el concepto de la condición → Una clase que se encargue de la pregunta
- El usuario pasa la consulta que desea realizar como parámetro
- Se pueden crear hijos de la condición con los tipos de preguntas que puedo realizar

Condicion

Delegó la consulta de si un medicamento la cumple o no

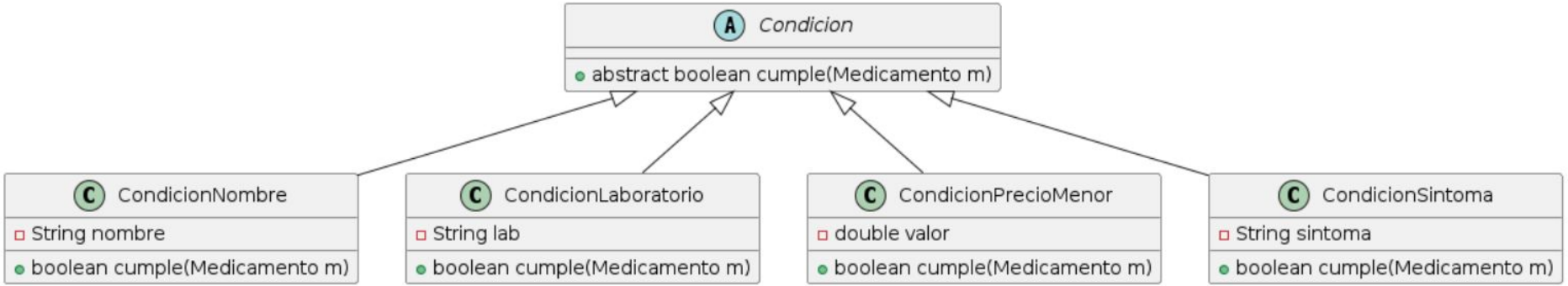



```
public ArrayList<Medicamento> buscarMedicamentos(Condicion cond){
    ArrayList<Medicamento> salida = new ArrayList<Medicamento>();
    for(int i =0; i<medicamentos.size();i++) {
        if (cond.cumple( medicamentos.get(i) )) {
            salida.add(medicamentos.get(i));
        }
    }
    return salida;
}
```

Farmacia - Condiciones

---

una condición por cada TIPO de pregunta que deseo hacer



Condicion Nombre

Encapsula la pregunta por nombre, tiene el nombre buscado de atributo

```
public class CondicionNombre extends Condicion {

    String nombre;

    //get y set Constructor

    public boolean cumple(Medicamento m) {
        return m.getNombre().equals(nombre);
    }

}
```

Condicion PrecioMenor

```
public class CondicionPrecioMenor extends Condicion {

    double valor;

    //get set y constructor

    public boolean cumple(Medicamento m) {
        return m.getPrecio() < valor;
    }

}
```

Condicion Sintoma

```java
public class CondicionSintoma extends Condicion {

    String sintoma;
    //get y set constructor
    public boolean cumple(Medicamento m) {
        // TODO Auto-generated method stub
        return m.tieneSintoma(sintoma);
    }

}
```

Pero pero y las combinaciones lóg

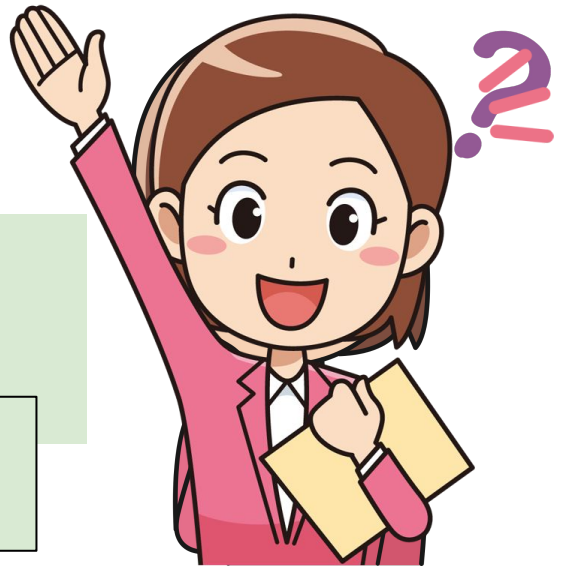
Como hago combinaciones lógicas?

Pero esto ya lo hicimos, al menos separado!!

Los anteriores son solo algunos ejemplos de las búsquedas que se desea proveer en la farmacia. Es posible que surjan nuevos tipos de busqueda asi como combinaciones lógicas de los anteriores, por ejemplo buscar todos los medicamentos del laboratorio “Bayer” y que el nombre contenga la palabra “ina”, o buscar todos los medicamentos que traten “Congestión” y cueste menos de 400 pesos

Son dos Condiciones que se unen por un operador lógico and

Lo mismo en el segundo ejemplo



Farmacia Condicion AND

- AND es una condición que se forma como la operación lógica and de dos condiciones
- Hereda de Condición
- Tiene dos condiciones

CondicionAnd

```
public class CondicionAnd extends Condicion {

    private Condicion c1;
    private Condicion c2;
    //gets y sets Constructor
    public boolean cumple(Medicamento m) {

        return c1.cumple(m) && c2.cumple(m);
    }

}
```

Es una Condicion

tiene dos condiciones

c1 y c2 pueden ser cualquier tipo de condición. Incluido otro and, un or o un not

la CondicionAnd se cumple si: se cumple la condición c1 Y se cumple la condición c2

CondicionOr

Similar al And, pero cambia la operación lógica

```
public class CondicionOr extends Condicion {


    private Condicion c1;
    private Condicion c2;
    //gets y sets Constructor
    public boolean cumple(Medicamento m) {

        return c1.cumple(m) || c2.cumple(m);
    }

}
```

CondicionNot

```
public class CondicionNot extends Condicion {

    private Condicion c1;
    public boolean cumple(Medicamento m) {

        return ! c1.cumple(m);
    }

}
```

La negación se cumple si NO se cumple la condición referida

Farmacia – Instanciación de las condiciones

La instanciación de las condiciones se realiza fuera de las clases que representan el modelo. Las hace el usuario cuando quiere hacer una busqueda

```
// Para buscar todos los medicamentos que contengan “ina”
Condicion cond1 = new CondicionNombre("ina");

// Para buscar todos los medicamentos del lab “Bayer”
Condicion cond2 = new CondicionLaboratorio("Bayer");

// Para buscar todos los medicamentos que contengan “ina” y sean del lab “Bayer”
Condicion cond3 = new CondicionAnd(cond1, cond2);

// Para buscar todos los medicamentos que contengan “ina” O sean del lab “Bayer”
Condicion cond4 = new CondicionOr(cond1, cond2);

// Para buscar todos los medicamentos que NO (contengan “ina” y sean del lab “Bayer”)
Condicion cond5 = new CondicionNot(cond3);
```

Solución final