




# PROCESO HITO

ROGER CAMACHO VALLEJOS



***¿CUAL ES LA DESCRIPCION QUE CREES QUE DEFINE  
MEJOR EL CONCEPTO 'CLASE' EN LA PROGRAMACION  
ORIENTADA A OBJETOS?***

ES UN MODELO O PLANTILLA A PARTIR DE LA CUAL CREAMOS OBJETOS



***¿QUÉ ELEMENTOS CREES QUE DEFINEN A UN OBJETO?***

SUS ATRIBUTOS Y SUS MÉTODOS



***¿QUE CODIGO DE LOS SIGUIENTES TIENE QUE VER CON  
MANEJO DE INTERFACES?***

PUBLIC CLASS MICLASE IMPLEMENTS PRODUCTO



# ***¿QUÉ SIGNIFICA INSTANCIAR UNA CLASE?***

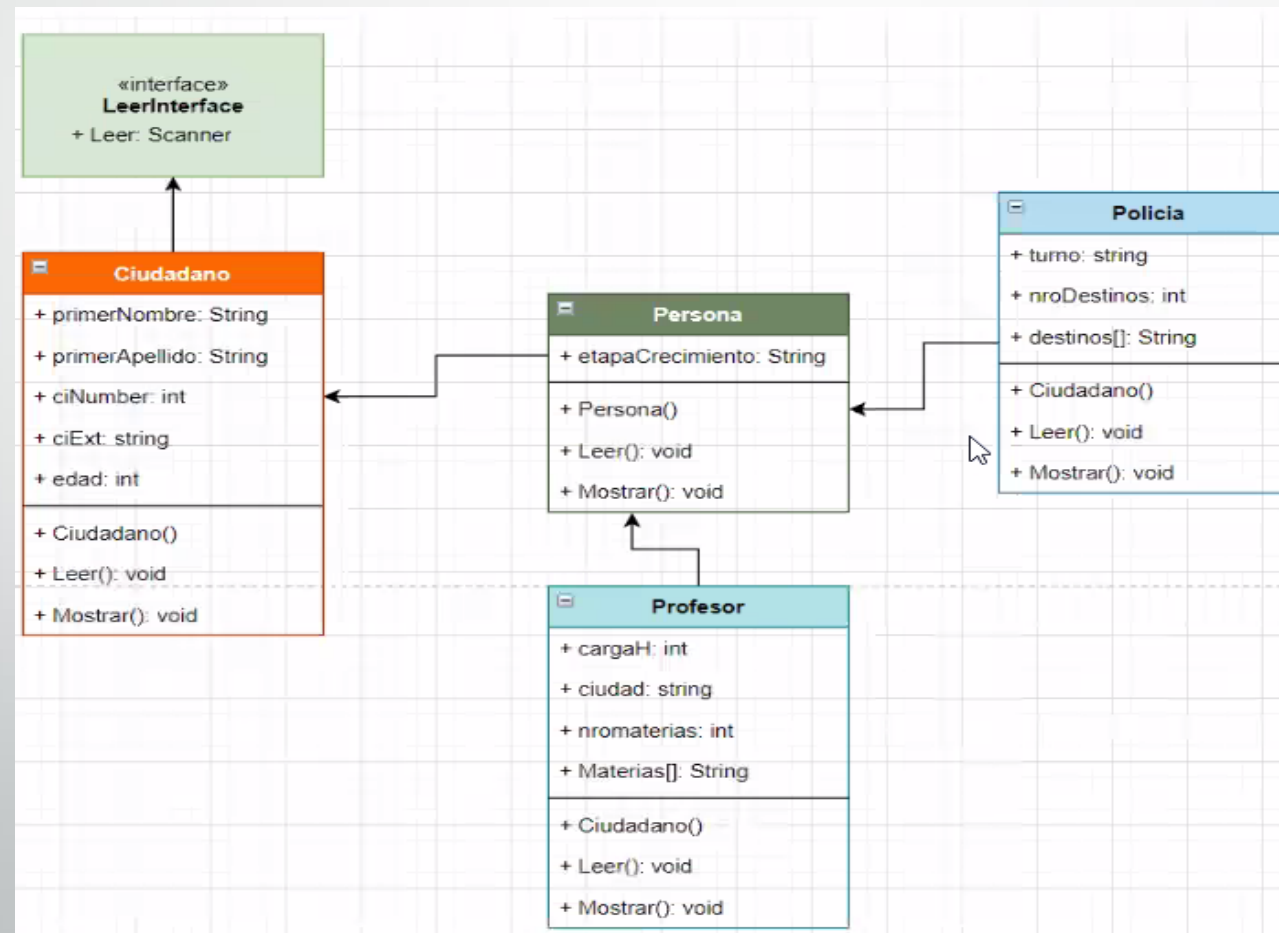
CREAR UN OBJETO A PARTIR DE CLASE

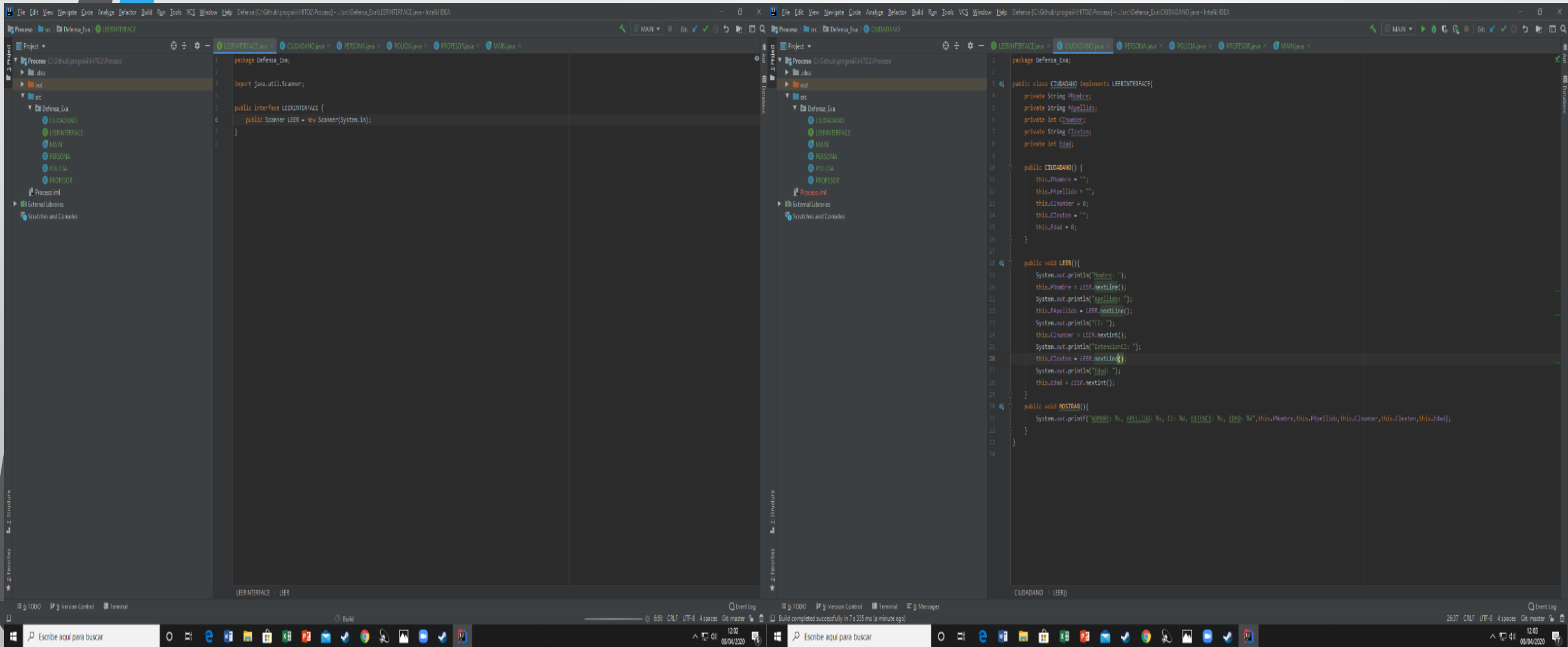


***DEFINA CON SUS PALABRAS QUE ES LA PROGRAMACION  
ORIENTADA A OBJETOS.***

ES UNA FORMA DE PROGRAMAR DE COSAS QUE VEMOS EN LA VIDA REAL

# GENERAR EL SIGUIENTE DISEÑO DE CLASES EN JAVA MANEJANDO HERENCIA Y EL MANEJO DE INTERFACES





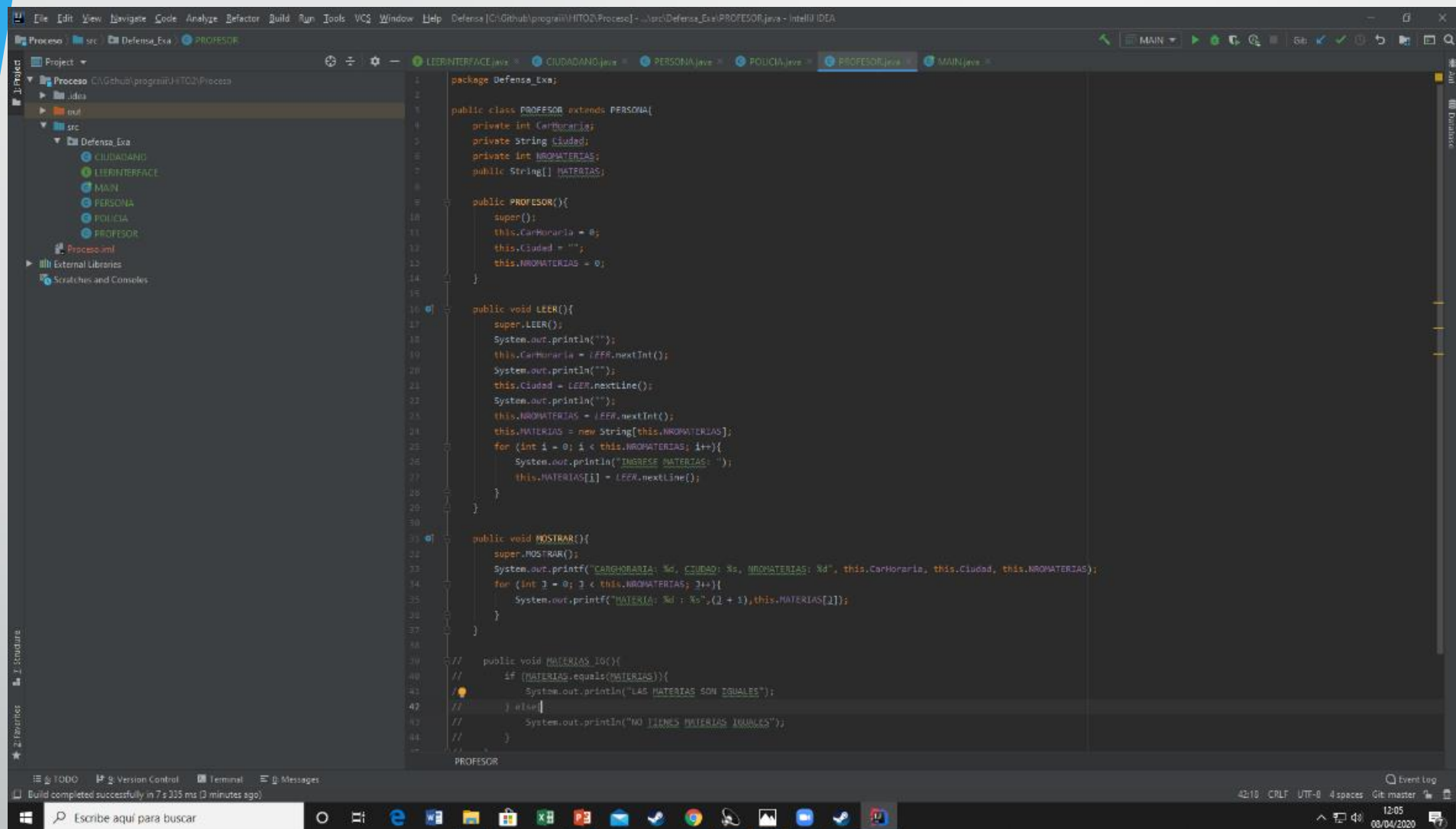


```
1 package Defensa_Eja;
2
3 public class PERSONA extends CIUDADANO{
4     public String EtapCrec;
5
6     public PERSONA(){
7         super();
8         this.EtapCrec = "";
9     }
10
11     public void LEER(){
12         super.LEER();
13         System.out.println("Etap de crecimiento: ");
14         this.EtapCrec = LEER.nextLine();
15     }
16
17     public void MOSTRAR(){
18         super.MOSTRAR();
19         System.out.printf("Etap: %s", this.EtapCrec);
20     }
21 }
22
```

PERSONA - LEER()

```
1 package Defensa_Eja;
2
3 public class POLICIA extends PERSONA{
4     private String Turno;
5     private int Ardest;
6     private String[] Dest;
7
8     public POLICIA(){
9         super();
10         this.Turno = "";
11         this.Ardest = 0;
12         //this.Dest = "";
13     }
14
15     public void LEER(){
16         super.LEER();
17         System.out.println("Turno: ");
18         this.Turno = LEER.nextLine();
19         System.out.println("Ardest: ");
20         this.Ardest = LEER.nextInt();
21         this.Dest = new String[this.Ardest];
22         for (int i = 0; i < this.Ardest; i++){
23             System.out.println("Destino: ");
24             this.Dest[i] = LEER.nextLine();
25         }
26     }
27
28     public void MOSTRAR(){
29         super.MOSTRAR();
30         System.out.printf("Turno: %s, Ardest: %d, this.Turno, this.Ardest);
31         for (int i = 0; i < this.Ardest; i++){
32             System.out.printf("Destino: %s", this.Dest[i]);
33         }
34     }
35 }
36
```

POLICIA



# ***INSTANCLAR 2 OBJETOS DE LA CLASE POLICIA Y DOS INSTANCIAS DE LA CLASE PROFESOR.***

```
19      case 1:
20          POLICIA P1 = new POLICIA();
21          POLICIA P2 = new POLICIA();
22          P1.LEER();
23          P1.MOSTRAR();
24          P2.LEER();
25          P2.MOSTRAR();
26
27          PROFESOR PR1 = new PROFESOR();
28          PROFESOR PR2 = new PROFESOR();
29          PR1.LEER();
30          PR1.MOSTRAR();
31          PR2.LEER();
32          PR2.MOSTRAR();
33          System.out.println();
34          break;
```

# ***INSTANCLAR 2 PROFESORES Y VERIFICAR SI TIENEN ASIGNADO UNA MISMA MATERIA.***

```
35         case 2:
36             PROFESOR PRF1 = new PROFESOR();
37             PROFESOR PRF2 = new PROFESOR();
38             PRF1.LEER();
39             PRF1.MOSTRAR();
40             PRF2.LEER();
41             PRF2.MOSTRAR();
42
43             if (PRF1.MATERIAS.equals(PRF2.MATERIAS)){
44                 System.out.println("TIENEN LAS MISMAS MATERIAS");
45             }else {
46                 System.out.println(" NO!!!! TIENEN MATERIAS IGUALES");
47             }
48             System.out.println();
49             break;
```