Identidad, igualdad y equivalencia

Introducción a la programación orientada a objetos

Clases como tipos - 1

- Cuando el analista o el diseñador de un sistema orientado a objetos especifica una clase, establece sus atributos y servicios.
- Los servicios incluyen:
  - Constructores
  - Métodos (comandos y consultas)
- La implementación de una clase en Java permite mantener esta misma estructura.

Clases como tipos - 2

- Cuando una clase incluye atributos y métodos define un **tipo de dato** a partir del cual es posible declarar variables y crear objetos. Estas clases pueden modelar:
  - Entidades del mundo real como ciudades, empleados, alumnos,…;
  - Representaciones matemáticas como un grafo, un vector, un punto en el espacio tridimensional, …;
  - Manejo de cadenas; etc.
- Por otro lado hay clases que simplemente brindan **servicios** como por ejemplo:
  - Integer, math, System, etc.

EJEMPLO

Ciudad
<<atributos de instancia>> CP: entero poblacion : entero superficie: real
<<Constructores>> Ciudad(cod :entero) Ciudad(cod,p:entero,s:entero) **<<Comandos>>** establecerPoblacion(p:entero) establecerSuperficie(s:real) aumentarPoblacion(p:entero) **<<Consultas>>** obtenerCP():entero obtenerPoblacion():entero obtenerSuperficie():real

Se asumen valores no negativos

Se incrementa la población en p, el valor de p puede ser negativo pero se asume controlado que la población sigue siendo no negativa

```
class Ciudad {
//atributos de instancia

private int CP;
private int poblacion;
private float superficie;

//Constructores
Public Ciudad (int cod){
  CP = cod;
}
Public Ciudad (int cod, int pob, float sup){
  CP = cod;
  poblacion = pob;
  superficie = sup;
}

…
}
```

```
…
//Comandos
public void establecerPoblacion (int p){
  poblacion = p; }

public void aumentarPoblacion (int p){
  poblacion = poblacion + p; } //población += p

public void establecerSuperficie (float s){
  superficie= s; }

//Consultas
public int obtenerCP () {
  return CP; }

public int obtenerPoblacion () {
  return poblacion; }

public int obtenerSuperficie () {
  return superficie; }
…
```

EJEMPLO

ProcesadorNumerico
<<Consultas>> sumaDig(n:entero): entero estaDig (n:entero, d:entero): entero promedioDigitos(n:entero):real

```
Static class ProcesarNumero {

public static int sumaDig(int n ){
…
}
public static boolean estaDig(int n,int d ){
…
}
public static int sumaP(int n){
…
}}
```

Objetos y Variables - 1

La instrucción de **declaración**:

Ciudad ciu;

Reserva espacio en memoria para una **variable** con nombre **ciu.**

Luego de la declaración, la instrucción de **creación**:

ciu = new Ciudad(8000);

- **Reserva** un espacio en memoria para mantener el estado interno del objeto;
- **Liga** el objeto de software a la **variable** **ciu**;
- **Invoca** al constructor ;

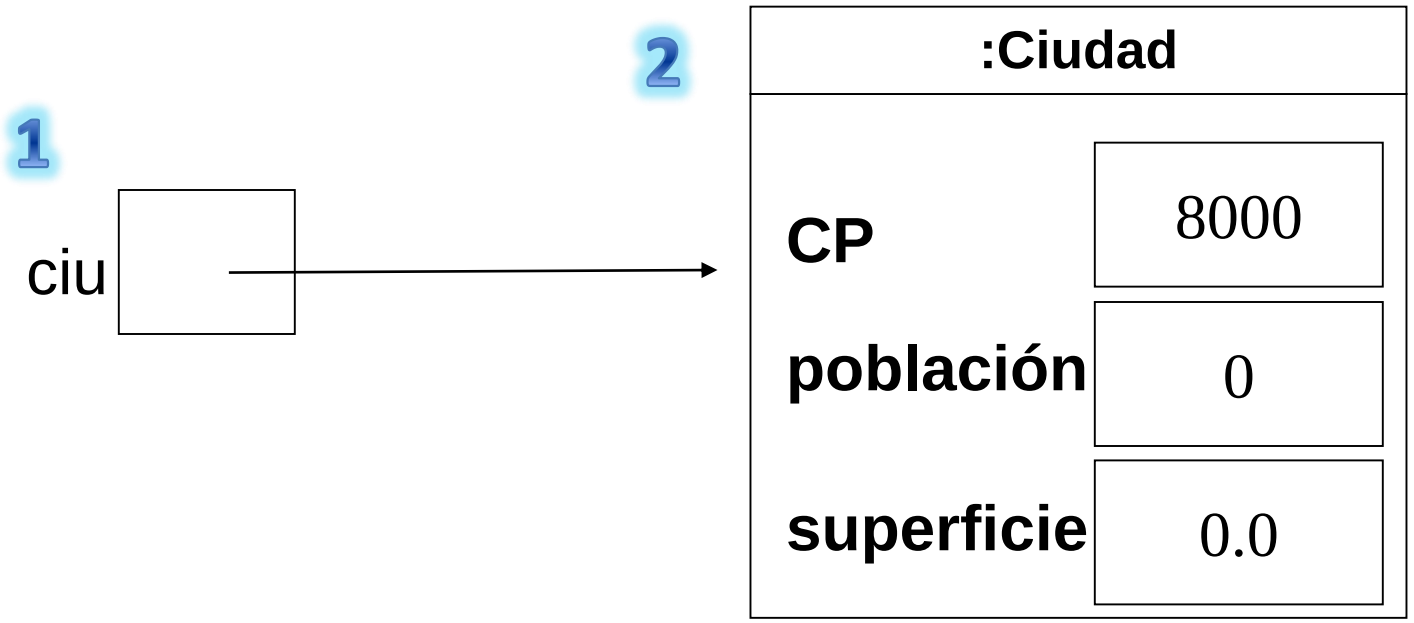
Objetos y Variables - 2

Diagrama de objetos para la **declaración**:

1 **Ciudad ciu;**

Y para la **creacion**:

2 **ciu= new Ciudad(8000);**



Objetos y Variables - 3

La siguiente instrucción **declara** la **variable** y **crea** el objeto:

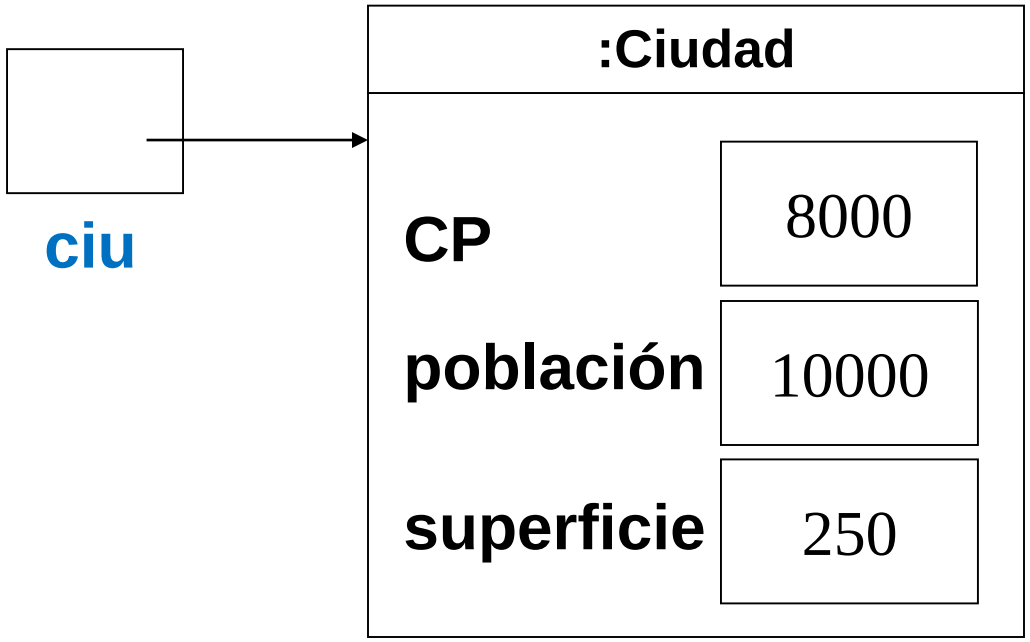
Ciudad ciu = new Ciudad(8000);

- **Reserva** espacio en memoria para una **variable** con nombre **ciu;**
- **Reserva** un espacio en memoria para mantener el estado interno del objeto;
- **Liga** el objeto a la **variable** **ciu**;
- **Invoca** al constructor.

Objetos y Variables - 4

Diagrama de objetos para la declaración:

```
Ciudad ciu = new Ciudad(8000,10000,250);
```



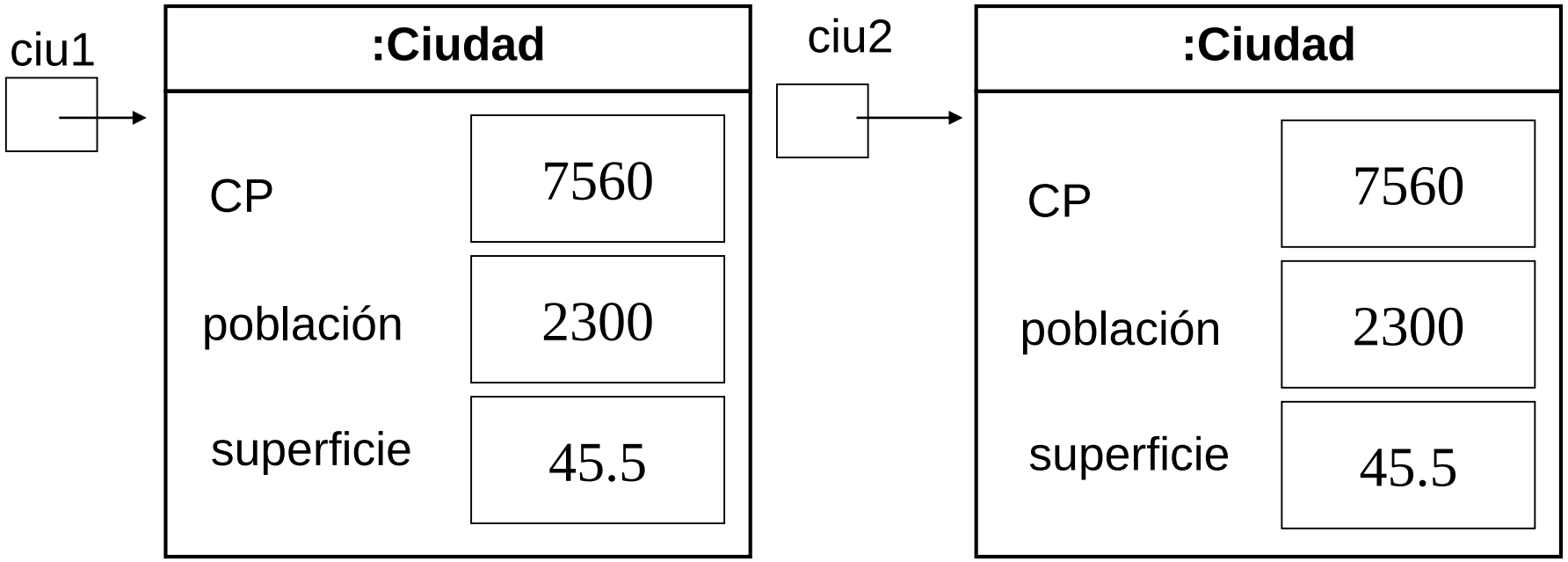
Problemática

¿Qué sucede si tenemos mas de una instancia de la clase ciudad y queremos saber si son iguales?

Identidad, Igualdad y Equivalencia - 1

```
Ciudad ciu1,ciu2;
ciu1=new Ciudad(7560,2300,45.5);
ciu2=new Ciudad(7560,2300,45.5);
if(ciu1==ciu2)
```

FALSE – se estan comparando las referencias -



Identidad, Igualdad y Equivalencia - 2

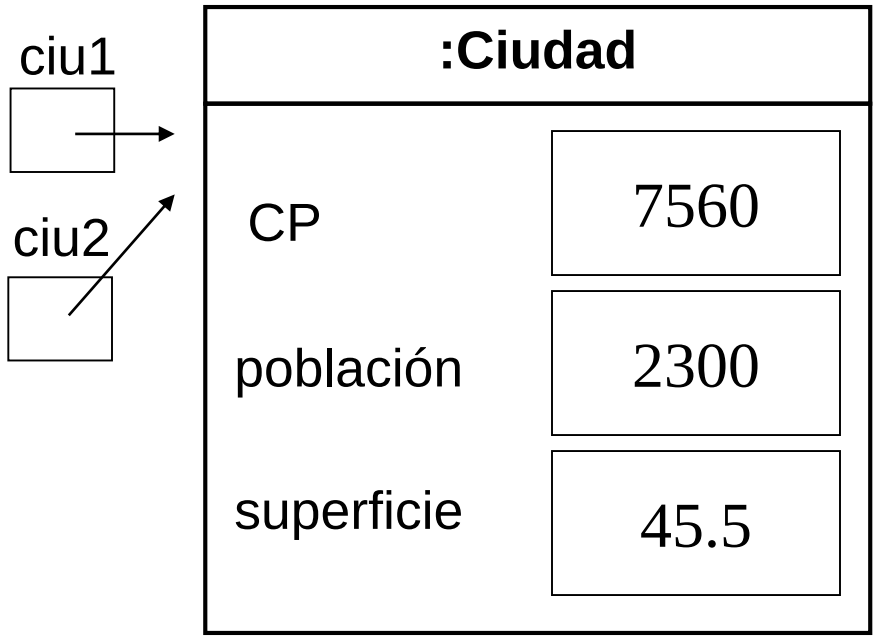
- Es importante distinguir entre el **nombre** de un objeto, el **objeto mismo** y el **objeto del problema** que representa.

- **Un mismo objeto puede tener dos nombres** diferentes y también es posible que **dos objetos diferentes, con diferentes nombres, representen a un mismo objeto del problema** (ejemplo anterior).

- Bajo estas circunstancias, ¿**cómo decidimos si dos objetos son iguales**?

Identidad, Igualdad y Equivalencia - 3

```
Ciudad ciu1, ciu2;
ciu1 = new Ciudad (7560,2300,45.5);
ciu2=ciu1;
```



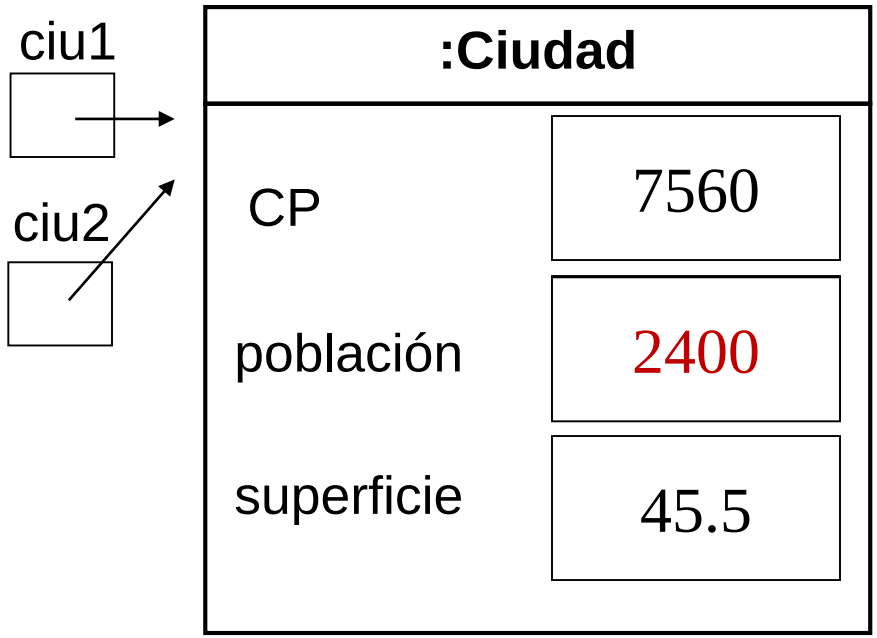
¿ciu2==ciu1?

TRUE

– porque ambas referencias estan asociadas al mismo objeto –

Identidad, Igualdad y Equivalencia - 4

```
Ciudad ciu1, ciu2;
ciu1 = new Ciudad (7560,2300,45.5);
ciu2=ciu1;
ciu2.aumentarPoblacion(100);
```



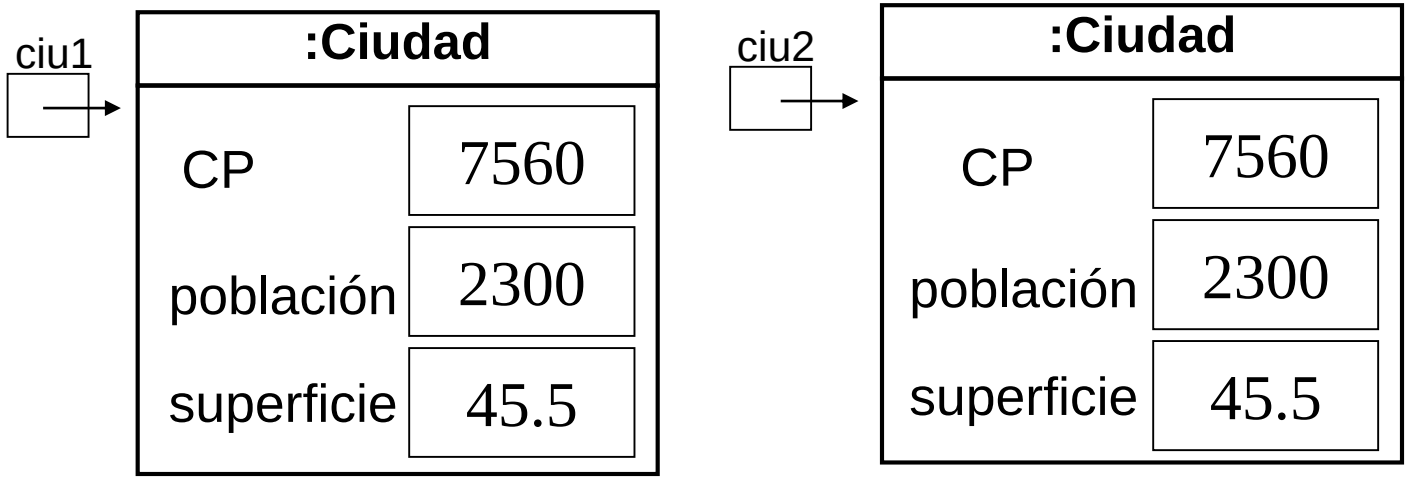
Como el mismo objeto es referenciado por dos variables, aunque se cambie su estado interno **ciu1==ciu2 siempre sera verdadero**

Identidad, Igualdad y Equivalencia - 5

```
Ciudad ciu1, ciu2;
ciu1 = new Ciudad (7560,2300,45.5);
ciu2 = new Ciudad (7560,2300,45.5);
```

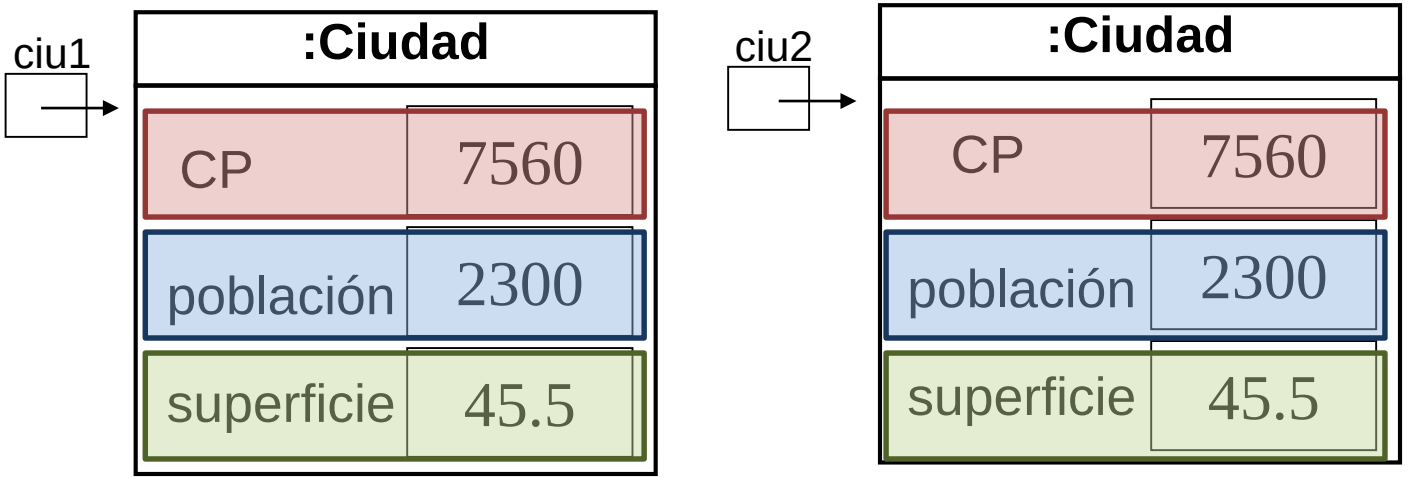
¿ciu2==ciu1? ➡ **FALSE**

¿Cómo comparamos dos objetos referenciados por distintas variables?



Equals - 1

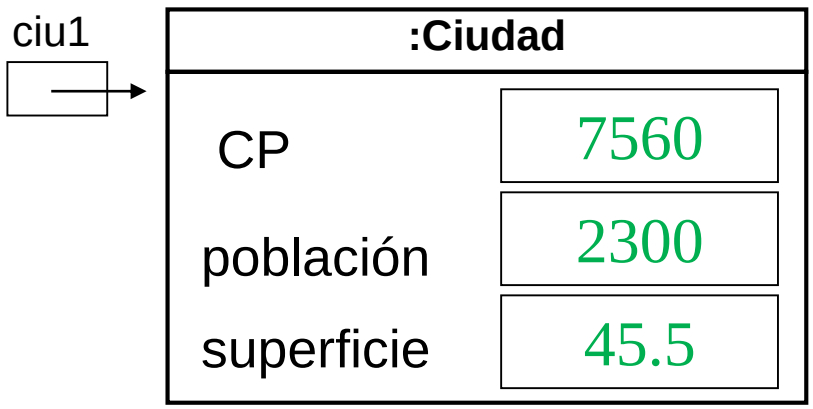
- Para comparar dos objetos instancia de la misma clase referenciados por distintas variables, tenemos que impementar el método **equals;**
- Este método debe comparar ambos objetos **atributo a atributo** para determinar su igualdad.

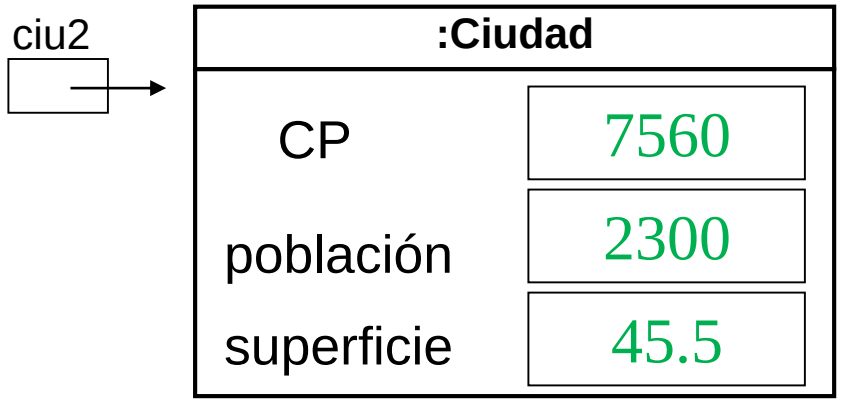


Equals - 2

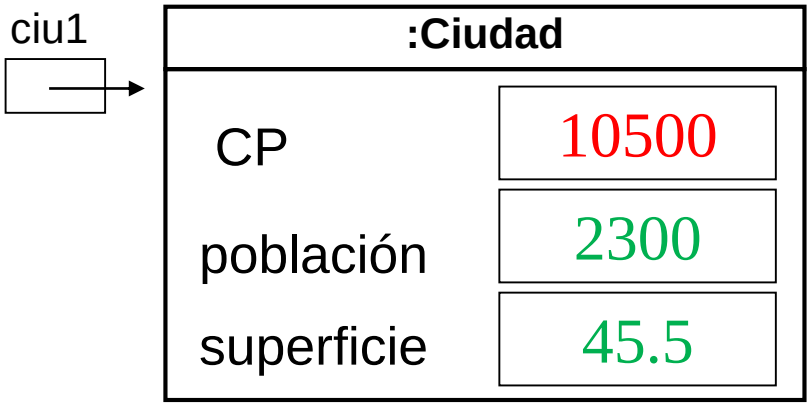
```
Public class ciudad{
private int CP;
private int poblacion;
private float superficie;
…
<<consultas>>
public boolean equals (Ciudad otra) {
   boolean resultado;
   resultado = CP == otra.obtenerCP() &&
   poblacion == otra.obtenerPoblacion()&& superficie
   == otra.obtenerSuperficie();
return resultado;}
…
}
```

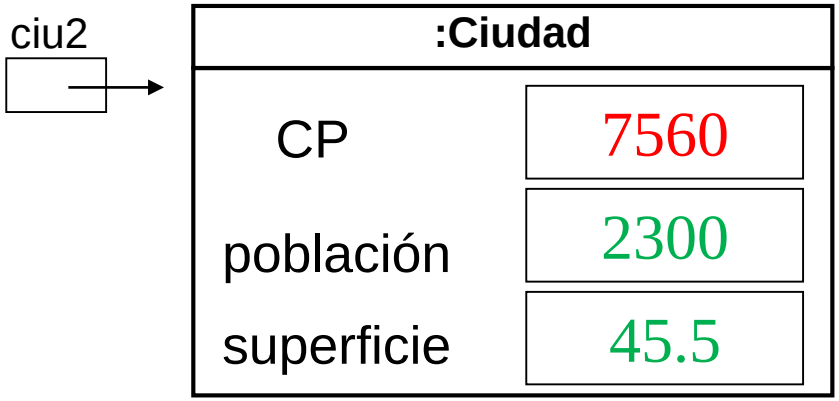
Equals - 3





¿ciu1.equals(ciu2)? ➡ **TRUE**

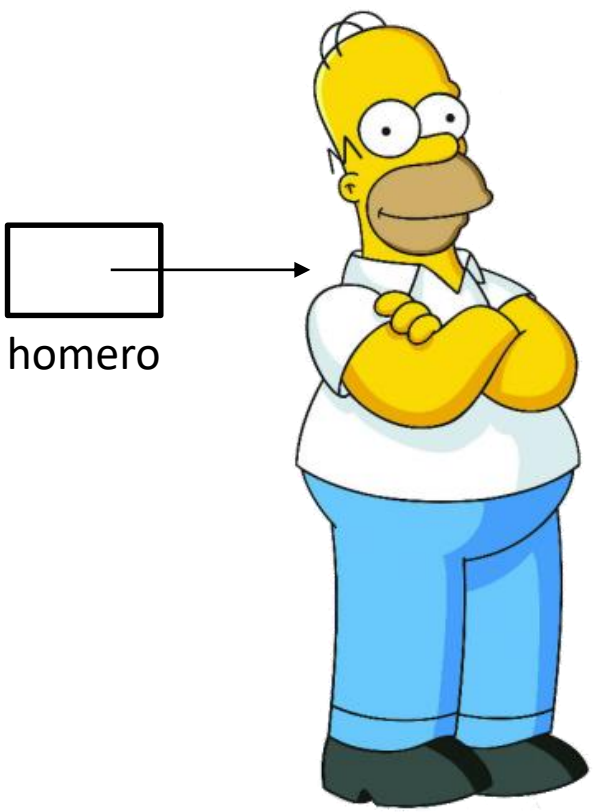




¿ciu1.equals(ciu2)? ➡ **FALSE**

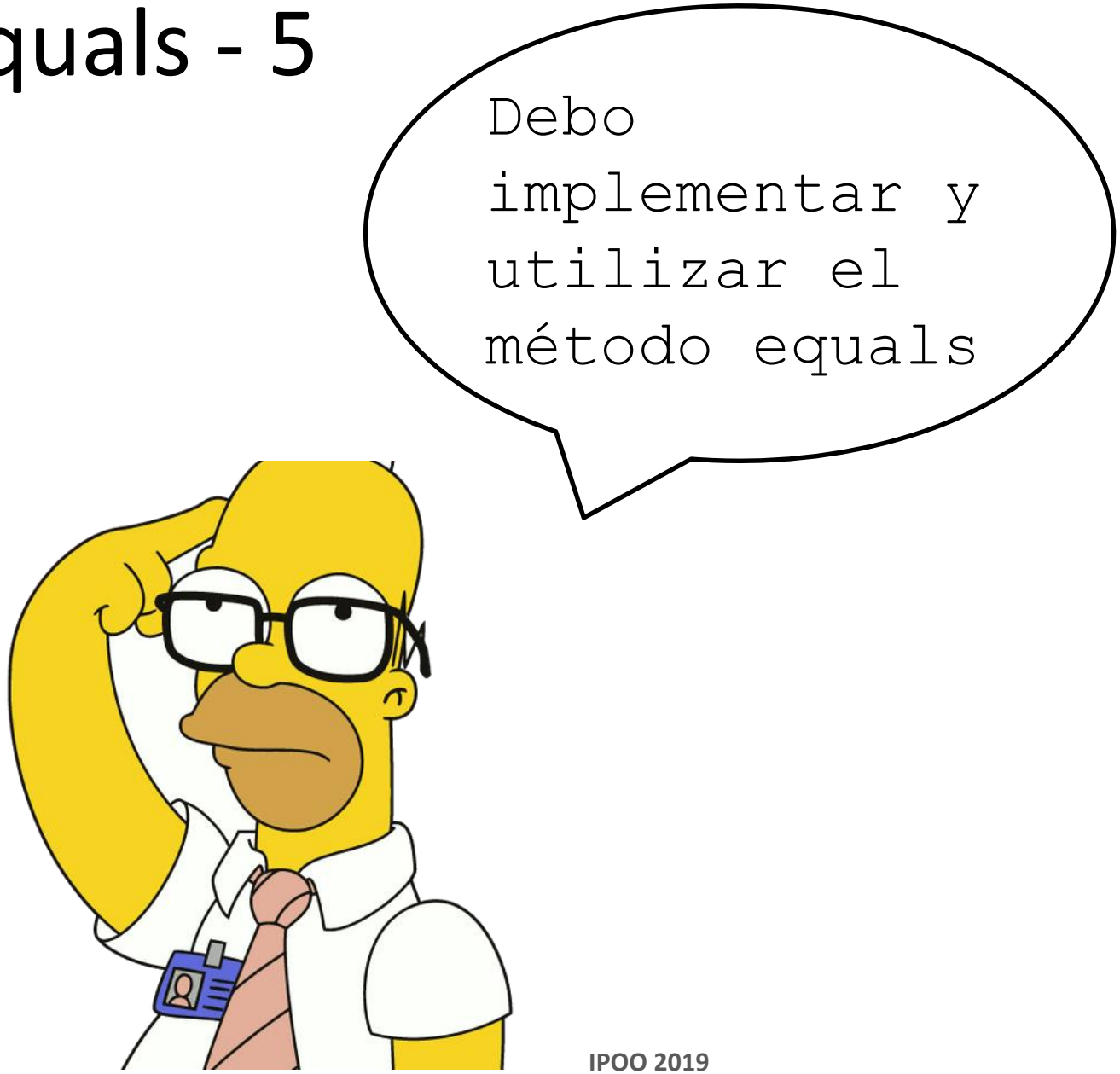
Equals - 4

```
Personaje homero, hombrePie;
```



¿Cómo determinamos si Homero y el Hombre Pie son objetos equivalentes?

Equals - 5



Resumen

- Al momento de comparar hay que tener en cuenta:
  - El operador **==** se utiliza para comparar igualdad de tipos elementales (char, boolean, int,float);
  - Si se utiliza el operador **==** entre dos variables de tipo clase el resultado sólo será **verdaro si ambas variables referencian al mismo objeto.**
  - Si se desea comparar la **igualdad entre dos objetos** se debe hacer mediante el método **equals** que **compara campo a campo el valor de sus atributos de instancia.**

Consejo

Leer los apuntes Teóricos!!!