



Programazioa

Objektuetara Bideratutako Programazioa (OBP)

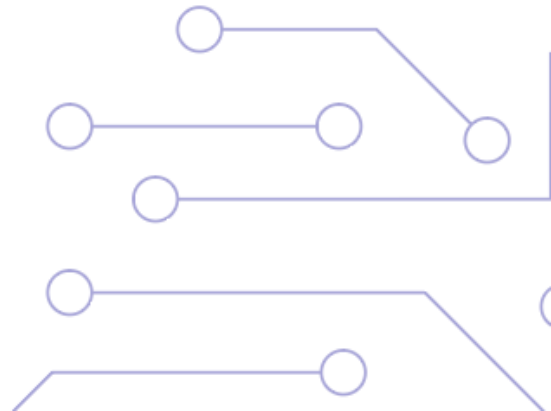


Objektuetara Bideratutako Programazioa

- Objektuen eta datuen inguruan antolatzen den programazio-paradigma bat da, ekintza eta logikaren inguruan baino
- Problema objektu izeneko entitate txikietan deskonposatzea ahalbidetzen du eta ondoren objektu horien inguruan datuak eta funtzioak eraikitzen ditu
 - Objektu hauek mundu errealeko gauzetan oinarritzen dira, adibidez, inbentarioetan edo langileen erregistroetan
 - Objektuek datuak dituzte eta logika bat jarraitzen dute
- Aplikazio bat elkarrekin komunikatzen diren objektu multzo bat da

Klase baten definizioa

- Klase bat objektuak sortzeko txantilo bat bezala uler daiteke
- Bi elementuz osatzen da:
 - **Atributuak:** Datuak irudikatzeko erabiltzen dira.
 - **Metodoak:** Objektuek egingo dutena definitzeko erabiltzen dira. Metodoek normalean klaseko atributuekin egingo dute lan.
- Klaseak lehen letra maiuskulaz izango dute, atributu aldagaiak eta metodoak lehen letra txikiz izango dute eta atributu konstanteak letra nagusiz idatziko dira
 - Konstanteek gainera *final* hitz erreserbatua izango dute



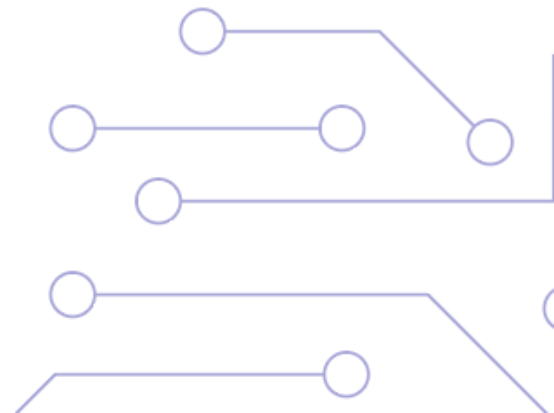
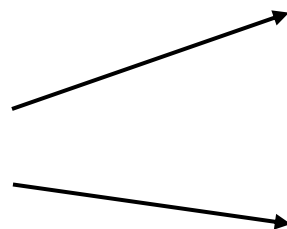
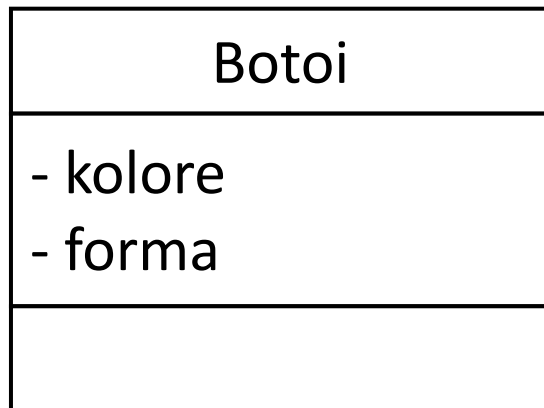
Klase baten definizioa

```
class Txakurra{  
    int tamaina;  
    String arraza;  
    String izena;  
    final int HANKA_KOP = 4;  
  
    void zaunkaEgin() {  
        System.out.println("Guau, guau");  
    }  
}
```

Txakurra
- tamaina
- arraza
- izena
- HANKA_KOP
- zaunkaEgin

Klasea vs. Objektua (instantzia)

- Klase bat objektu bat definitzeko errezeta bat da, abstraktua da, ez du daturik gordetzen
- Klase batean oinarritutako objektu ezberdin asko sor daitezke eta bakoitzak bere balioak izan ditzake
 - Adibidez, Botoi klase batean oinarrituta hainbat botoi sor daitezke, bakoitza bere kolore eta formarekin

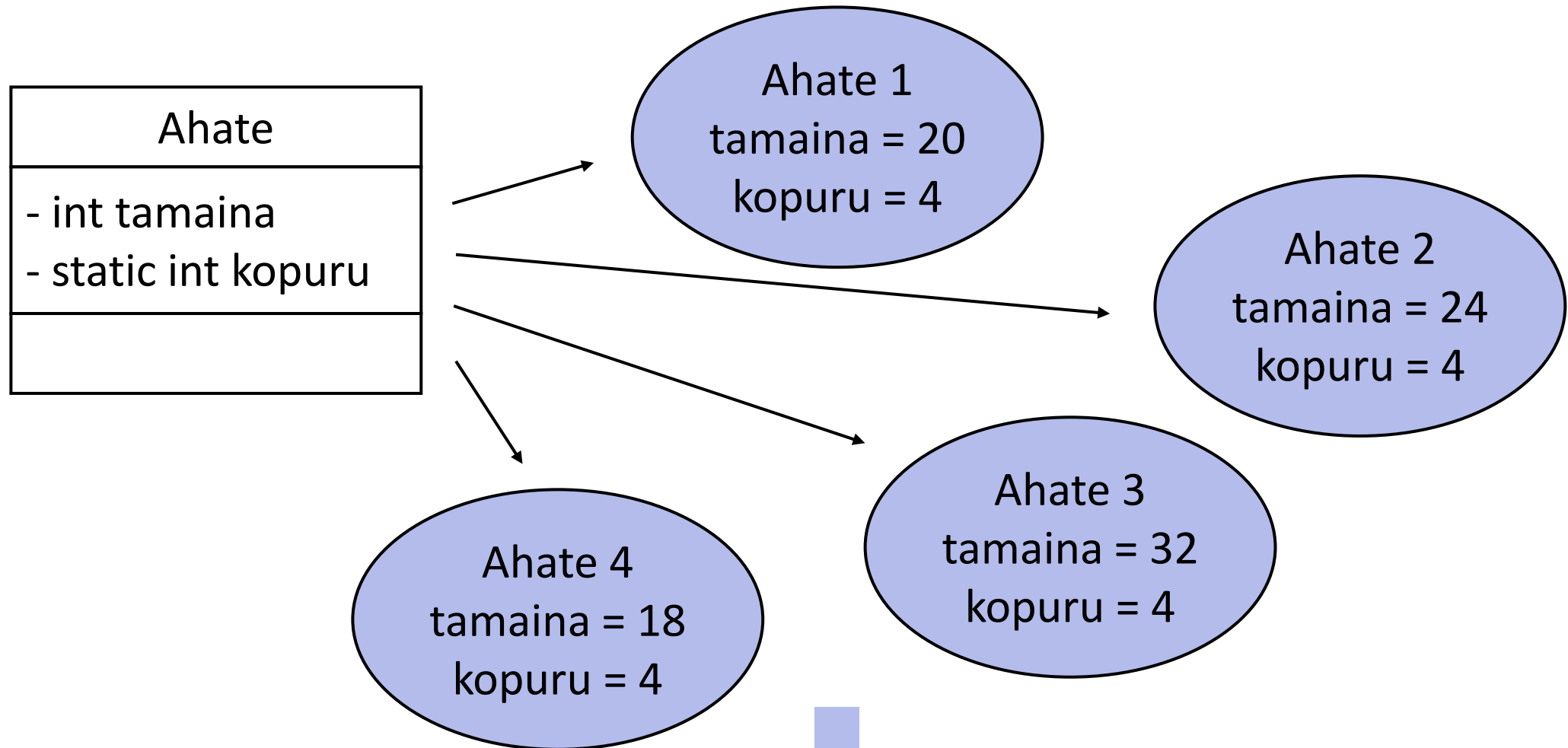


Atributuak eta klase aldagaiak

- Atributuak klase bati buruz gorde nahi den informazioa definitzeko erabiltzen dira
 - Datu-mota primitiboak (int, float...) edo objektuak izan daitezke
- Definizioa: *datu-mota atributulzena*;
- Aurretik *final* hitza gehitzen bazaio konstante bat izango da
- Aurretik *static* hitza gehitzen bazaio atributuaren balioa klase honen bidez sortutako objektu guztien artean konpartituko da

```
float luzera;  
String izena;  
final String KOLORE;  
static int instantziaKop;
```

Atributuak eta klase aldagaiak



Atributuak eta klase aldagaiak

- **Instantzia aldagaiak** klasearen testuinguruan definitzen dira eta klaseko metodo guztietan erabili daitezke
 - Balio lehenetsi bat dute
- **Aldagai lokalak** metodo baten testuinguruan definitzen dira eta metodotik kanpo ezin dira erabili
 - Ez dute balio lehenetsirik, beraz, erabili aurretik balio bat esleitu behar zaie

Objektuen konparaketak

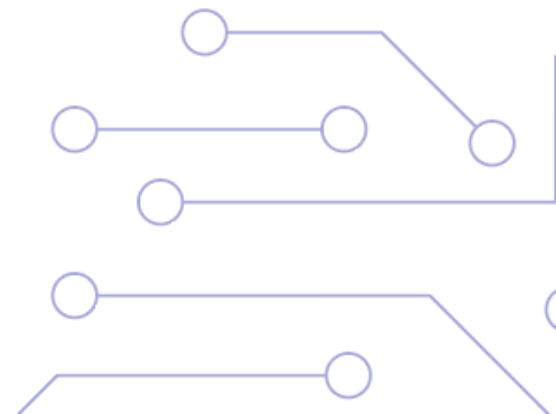
- Datu-mota primitiboak konparatzeko "==" konektorea erabiltzen da
- Objektuen artean "==" konektorea bi objektuk objektu bera direla (hau da, memoria posizio berdina) konprobatzeko erabiltzen da
 - Ezin da erabili bi objektu ezberdinek atributu balio berdinak dituzten konparatzeko, honetarako *equals()* metodoa erabiltzen da



Objektuak
konparatzeko ezin da
"==" konektorea
erabili

Metodoak


- Metodoek klase baten portaera definitzen dute
- Klase baten instantzia guztiek metodo berdinak izango dituzte
 - Beharbada, metodoek ezberdin jokatu dezakete atributu baten balioaren arabera





Metodoak

```
class Txakurra{  
    int tamaina;  
    String izena;  
    void zaunkaEgin() {  
        if (tamaina > 40) {  
            System.out.println("Woof! Woof!");  
        }  
        else {  
            System.out.println("Yip! Yip!");  
        }  
    }  
}
```



Metodoak

- Posible da metodo bati balioak ematea
 - Adibidez, txakur bati zenbat aldiz egin behar duen zaunka esan diezaiokegu

```
nireTxakurra.zaunkaEgin(3)
```

- Metodo batek **parametroak** ditu, deitzaileak **argumentuak** ematen ditu
 - Argumentuak metodo bati ematen zaizkion balioak dira
 - Parametroak mota eta izen bat duten metodo mailako aldagaiak dira, besterik ez

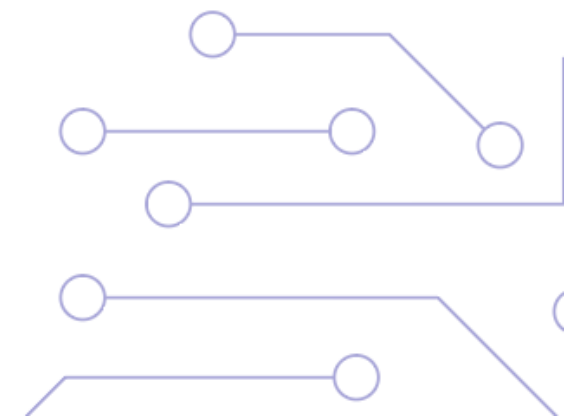
```
void zaunkaEgin(int zaunkaKop) {  
    while (zaunkaKop > 0) {  
        System.out.println("Guau!"); zaunkaKop--;  
    }  
}
```

Metodoak

- Metodoak prozedurak edo funtzioak izan daitezke
- Prozedurek ez dute baliorik itzultzen (*void* hitzarekin adierazten dira)
- Funtzioek balio bat itzultzen dute
 - Itzultzen duten balioaren mota izenaren aurretik adierazi behar da

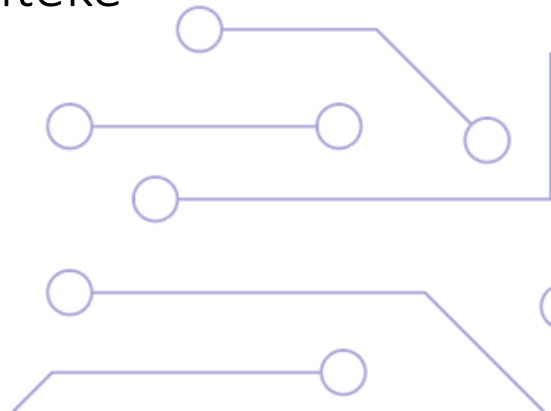
```
int zenb = sekretua.esanZenbakia();
```

```
int esanZenbakia() {  
    return 42;  
}
```



Balio-bidezko erreferentzia

- Javan metodoen parametroak balio-bidez pasatzen dira
 - Balioaren kopia pasatzen zaio metodoari
- Pasatu behar den parametroa datu-mota primitibo bat bada, honen kopia bat pasatzen da
- Parametroa objektu bat bada, honen memoriako erreferentzia pasatzen da, ez objektuaren kopia
 - Objektua bakarra izaten jarraitzen du eta metodotik modifikatu daiteke
 - Funtzioetan ez da beharrezkoa objektua itzultzea



Objektu metodoak vs. Klase metodoak

- Objektu metodoak erabiltzeko objektu instantzia bat sortuta egon behar da
- Klase metodoak *static* bezala definitzen dira eta klasearen izena adieraziz erabili daitezke
 - Normalean atributu estatikoen balioak jasotzeko erabiltzen dira

```
class NireKlasea{  
    static jasoBalioa() {  
        ...  
        NireKlasea.jasoBalioa();  
    }  
}
```



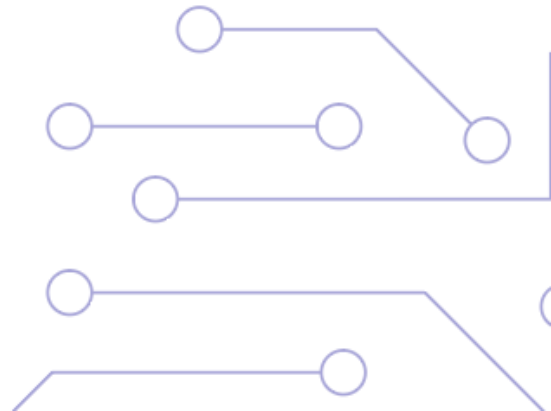

Objektu metodoak vs. Klase metodoak

- Objektu metodoek atzitu ahal dituzte:
 - Objektuaren aldagaiak eta metodoak
 - Klase aldagaiak eta klase metodoak
- Klase metodoek atzitu ahal dituzte:
 - Klase aldagaiak eta klase metodoak
- Klase metodoek ezin dituzte atzitu:
 - Objektu jakin baten aldagaiak eta metodoak
- Klase metodoak erabiltzen dira:
 - Objektuekin lotuta ez dauden atributuekin lan egiteko (klase atributuak)
 - Helburu orokorreko metodoen liburutegi bat sortzeko (String, Integer...)

Objektu baten sorrera (instantziazioa)

```
Txakur nireTxakurra = new Txakurra();
```

- 3 pausotan egiten da:
 - Aldagaia definitzen da objektuaren mota adieraziz eta izen bat emanez.
 - new hitza erabiltuta Javak objektua gordetzeko haina memoria erreserbatzen du
 - Erabiliko den memoriaren erreferentzia aldagaian gordetzen da





ADI: Aldagaien ez da
objektu baten kopia
gordetzen, baizik eta
honen memoriako
erreferentzia


Eraikitzailleak

- Metodo baten itxura hartzen dute, baina objektuak instantziatzeko erabiltzen dira
 - Objektu bat sortzean **new** hitzaren atzetik adierazten dira
 - Klasearen izen berdina dute (lehen letra maiuskulaz) eta ez dute baliorik itzultzen (ez dira ez prozedura ez funtzio)
- Eraikitzaillearen kodea objektuaren erreferentzia sortu aurretik exekutatzen da
- Eraikitzaillearen ohiko erabilera klaseko atributuei balioak esleitzea da
- Klase batek eraikitzaille bat baino gehiago izan dezake
 - Ez bada eraikitailerik definitzen Javak lehenetsitako bat sortzen du



Eraikitzaileak

```
class Ahatea{  
    int tamaina;  
    public Ahatea{  
        tamaina = 24;  
    }  
    public Ahatea(int aTamaina){  
        tamaina = aTamaina;  
    }  
}
```



Objektuen metodoen atzipena

- OBPko lengoaia gehienek "." karakterea erabiltzen dute objektu baten atributuak eta metodoak atzitzeko
 - Atributua: `instantzialzena.atributua`
 - Metodoa: `instantzialzena.metodolzena(parametroak)`

```
class Txakurra{                                Txakurra nireTxakur = new Txakurra();
    int tamaina;                                nireTxakur.tamaina = 30;
    Txakurra() {...}                            nireTxakur.zaunkaEgin();
    void zaunkaEgin() {...}                      nireTxakur.korrikaEgin(10);
    void korrikaEgin() {...}
}
```

Aldagaien irismena (testuingurua)

- Klaseko atributuak:
 - Java klaseko edozein metodotik atzitu daitezke
- Aldagai lokalak:
 - Definitzen diren metodoaren barruan bakarrik atzitu daitezke
 - Aldagai lokal batek klaseko atributu izen berdina badu lokala gailenduko da
 - Klaseko atributua erabiltzeko *this* hitz erreserbatua erabili beharko da
- Begizta aldagaiak:
 - Definitzen diren begiztaren barruan bakarrik erabili daitezke
 - Begizta aldagaiak ere izen berdineko eta irismen zabalagoko aldagaiei gailentzen zaizkie

Aldagaien irismena (testuingurua)

```
class Txakurra{  
    int adina;  
    void zaunkaEgin(int kop) {  
        for (int i = 0; i < kop; i++) {  
            System.out.println("Guau!");  
        }  
    }  
}
```


Aldagaien irismena (testuingurua)

```
public class Scope {  
    int x = 10;  
    private static int f1(int x) {  
        x = 1;  
        return x *= 2;  
    }  
    private static void f2() {  
        x -= 1;  
    }  
}
```

```
public static void main (String [] args) {  
    int x = 5;  
    x = f1(x);  
    System.out.println("x = " + x );  
    System.out.println("x = " + this.x);  
    f2();  
    x = f1(this.x);  
    System.out.println("x = " + x );  
    System.out.println("x = " + this.x);  
    this.x = f1(x);  
    System.out.println("x = " + x );  
    System.out.println("x = " + this.x);  
}}
```