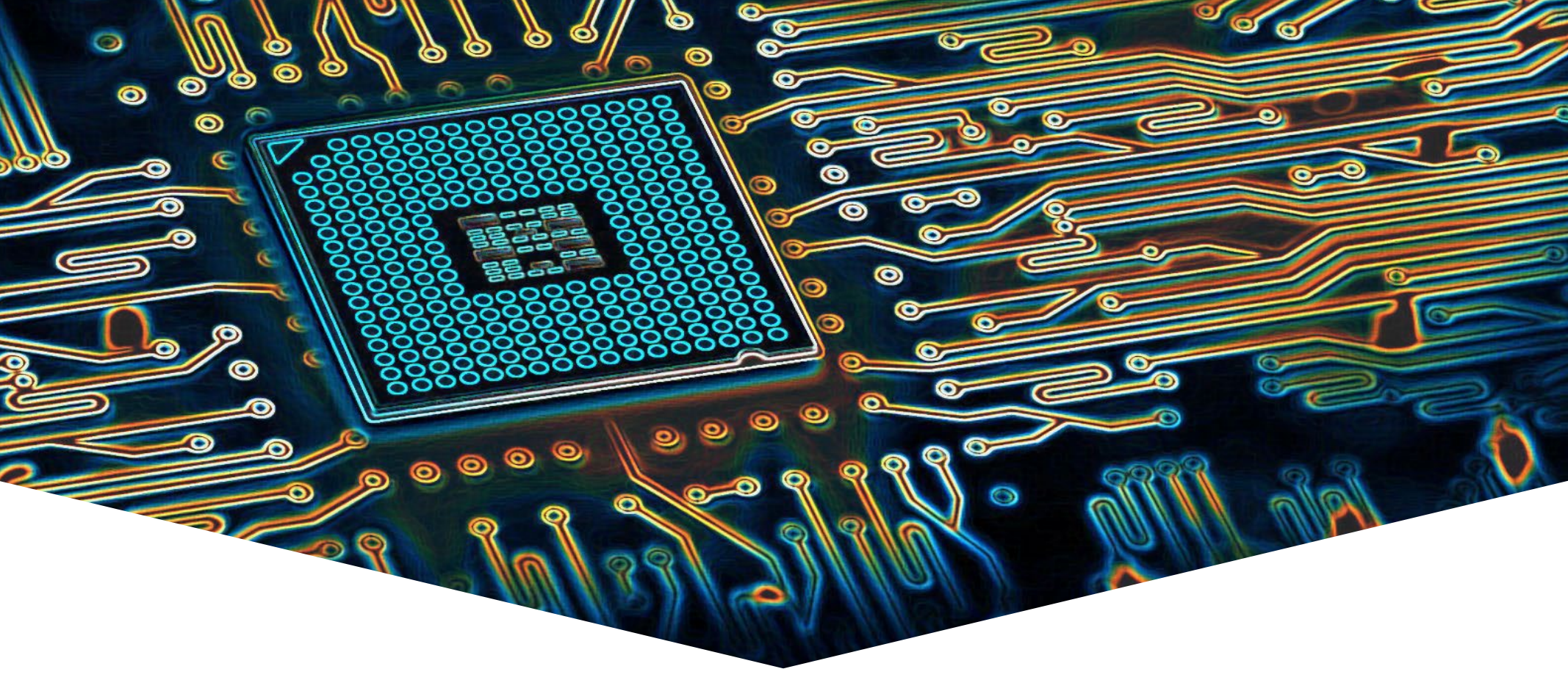


EMS31 Kwartaal 4 Week 4: Overerving en polymorfisme

Hergebruik van classes

Aggregatie (Aggregation)

- ... heeft een (of meer) ...

Overerving (Inheritance)

- ... is een (speciaal soort) ...

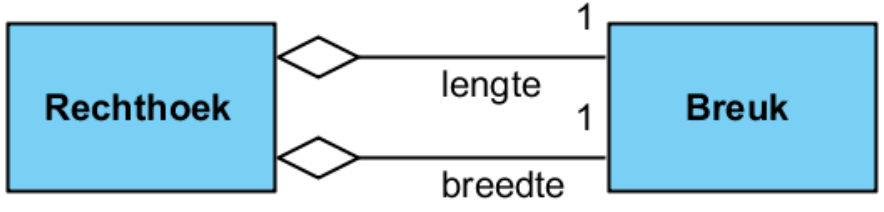
Aggregatie

Aggregatie (Aggregation)

- … heeft een (of meer) …

```
class Rechthoek {
public:
    // ...
private:
    Breuk lengte;
    Breuk breedte;
};
```

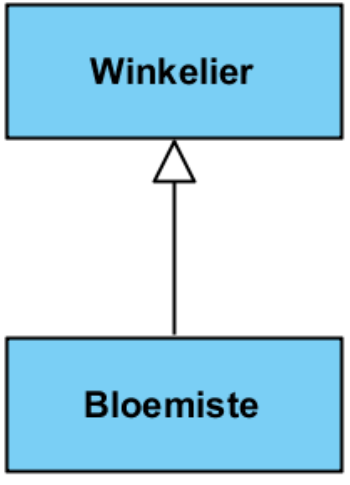
Een Rechthoek **heeft een** lengte en een breedte van het type Breuk.



Overerving

Overerving (Inheritance)

- … is een (speciaal soort) …




```
class Winkelier {
// ...
};

class Bloemiste: public Winkelier {
// ...
};
```

Een Bloemiste **is een (speciaal soort)** Winkelier.

UDT aanpak


```
enum Soort {sint_bernard, teckel};

class Hond {
private:
    Soort s;
    // ...
public:
    Krant haal_krant();
    void blaf();
    // ...
};

Krant Hond::haal_krant() {
    // ...
    blaf();
    return krant;
}
```

Niet goed uitbreidbaar!

Bron: https://webstockreview.net/images250_/dogs-clipart-newspaper-1.png

```
void Hond::blaf() {
    switch (s) {
        case sint_bernard:
            cout << "WOEF WOEF";
            break;
        case teckel:
            cout << "kef kef";
            break;
    }
}
```

Objectgeoriënteerde aanpak

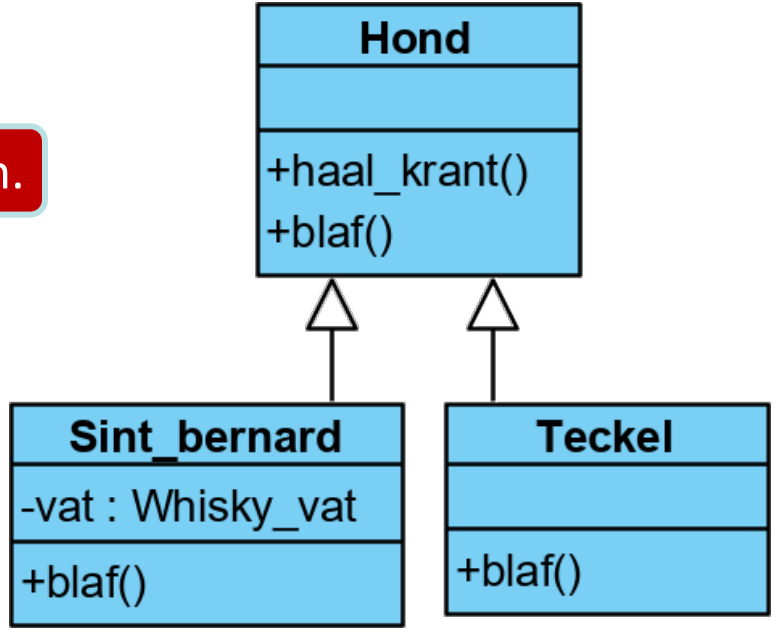

```
class Hond {
private:
    // ...
public:
    Krant haal_krant();
    virtual void blaf();
    // ...
};

Krant Hond::haal_krant() {
    // ...
    blaf();
    return krant;
}
void Hond::blaf() {
    cout << "blaf blaf";
}
```

Message kan **overridden** worden.

Een teckel **is een** hond.

Een sint-bernard **is een** hond.



OO aanpak


```
class Sint_bernard: public Hond {
private:
    Whisky_vat vat;
public:
    void blaf() override;
};

void Sint_bernard::blaf() {
    cout << "WOEF WOEF";
}
```

Bron: https://www.clipartkey.com/view/iRoowbh_vector-illustration-of-st-bernard-ski-patrol-and/

Bron: https://pngimg.com/uploads/dachshund/dachshund_PNG11.png

```
class Teckel: public Hond {
public:
    void blaf() override;
};

void Teckel::blaf() {
    cout << "kef kef";
}
```

Polymorfisme

doe_je_werk is een **polymorfe** functie.

h is een **polymorfe** parameter.

```
void doe_je_werk(Hond& h) {
    Krant k {h.haal_krant()};
    // ...
}

int main() {
    Sint_bernard boris;
    Teckel harry;

    if (!weekend)
        doe_je_werk(harry);
    else if (zaterdag)
        doe_je_werk(boris);
}
```

harry is van de class Teckel maar een Teckel **is een** Hond.

boris is van de class Sint_bernard maar een Sint_bernard **is een** Hond.

http://www.clipartgratis.it/animali/dettagli_divertenti.php?id=4766

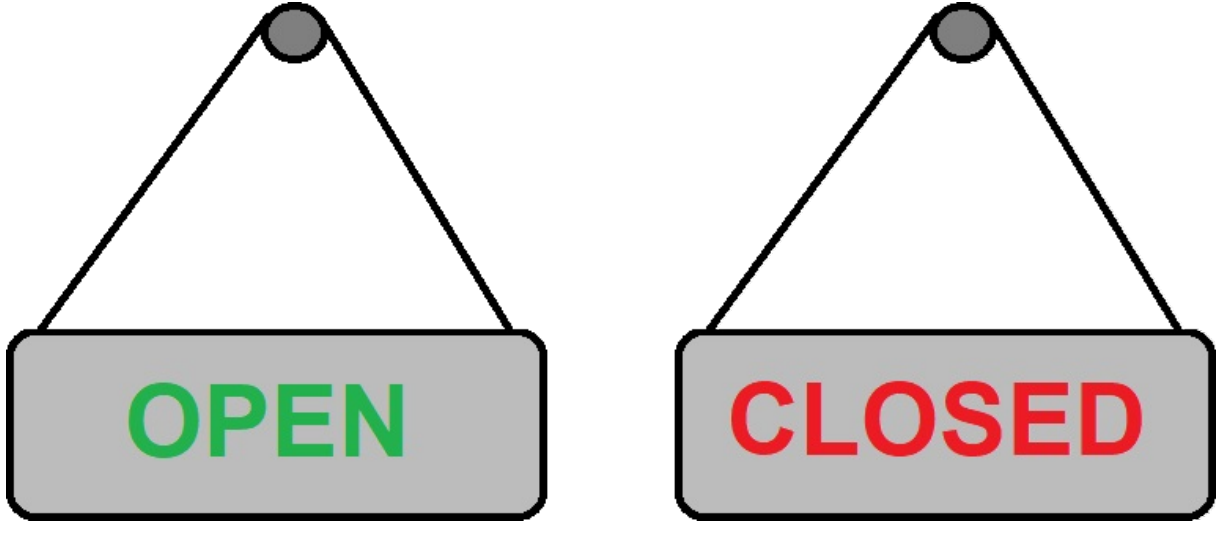
Uitbreidbaarheid

- Door de objectgeoriënteerde aanpak kan heel eenvoudig een nieuwe soort hond worden toegevoegd.
- Voeg zelf de classes `DuitseHerder` en `MechelseHerder` toe.
- Welke code moet nu gewijzigd worden?
- Welke code moet nu opnieuw gecompileerd worden?

Bron: http://www.clipartgratis.it/animali/dettagli_divertenti.php?id=4766

Open/closed-principe

Class Hond is open voor uitbreiding maar gesloten voor verandering.



Bron: https://scientificprogrammer.net/2019/10/22/why-open-closed-principle-is-important/

Zie eventueel:

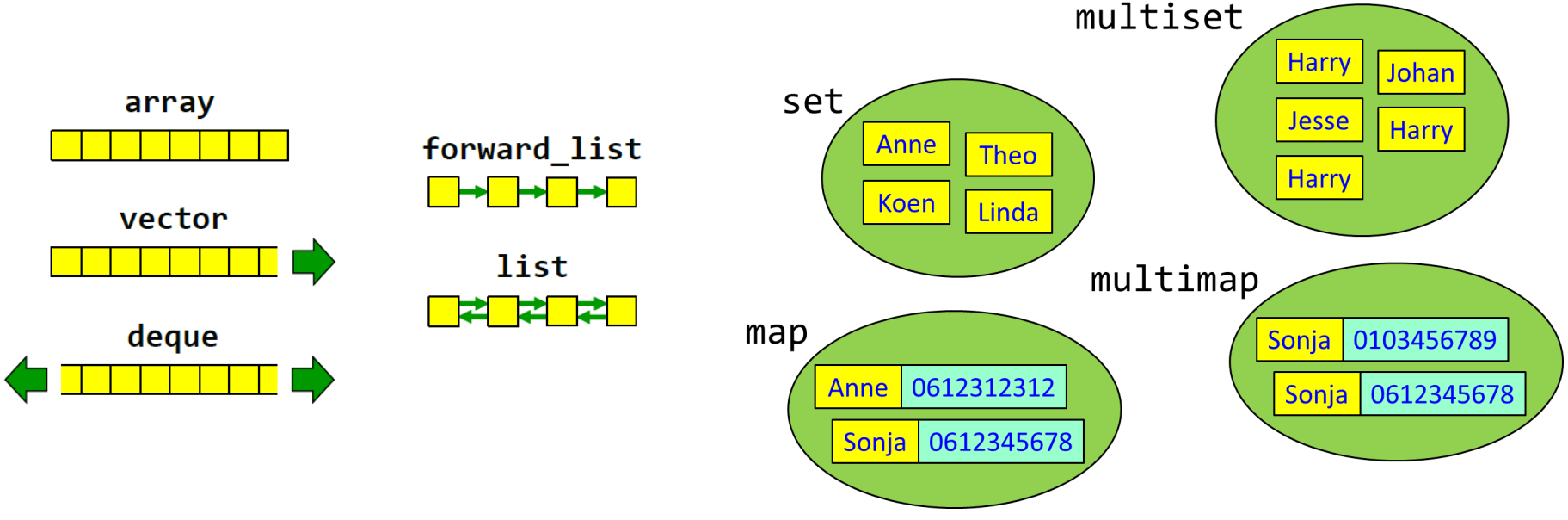
- https://en.wikipedia.org/wiki/Open%E2%80%93closed_principle

Details, details, …

- Tot hier: dictaat H4 t/m paragraaf 4.3
- Details:
  - Abstract base class (dictaat: §4.4)
  - Constructors bij inheritance (dictaat: §4.5)
  - `protected` members (dictaat: §4.6)
- Voorbeelden:
  - ADC kaarten (dictaat: §4.7)
  - Impedantie calculator (dictaat: §4.8)
- Details:
  - Dictaat hoofdstuk 18 (§18.1 t/m §18.11)

Volgende les…

Datastructuren



Aan de slag!

Aan de slag met Opdrachten_Week_4.4.pdf