

## Oppgaver

Finnes også på nett:

[https://github.uio.no/IN1010/seminaroppgaver\\_v19/blob/master/uke04.md](https://github.uio.no/IN1010/seminaroppgaver_v19/blob/master/uke04.md)

<p><b>1.a - Dyr, Hund og Katt:</b> Skriv en klasse Dyr. Klassen skal være tom, utenom en metode lagLyd(). Lag deretter to subclasser av dyr; Hund og Katt.</p> <p>Om man kaller på lagLyd i et hundeobjekt skal det printes "Voff!" til terminalen. Om man kaller på lagLyd i et katteobjekt skal det printes "Mjau!"</p>	<pre>abstract class Dyr{     public abstract void lagLyd(); }  class Hund extends Dyr{      public void lagLyd(){         System.out.println("Voff!");     } }  class Katt extends Dyr{      public void lagLyd(){         System.out.println("Mjau!");     } }</pre>
---	---

<p><b>1.b - Diskuter i grupper:</b> Hvilken fordel får vi ved å ha en superklasse med en tom metode lagLyd?</p> <p>I felleskap/gruppen: Skriv et testprogram som demonstrerer denne fordelene.</p>	<pre>public class Test{     public static void main(String[] args) {          Dyr dyr1 = new Hund();         Dyr dyr2 = new Katt();          dyr1.lagLyd();         dyr2.lagLyd();      } }</pre>
--	---

<p><b>1.c - Tegning:</b> På egen hånd: Tegn klassehierarki og datastruktur til oppgaven.</p> <p>Diskuter med gruppen: Sammenlign tegninger. Hva har dere gjort ulikt?</p>	
---	--

## 2.a - Person og student:

Diskuter hvilke metoder her som er eksempler på overriding, og hvilke metoder som er eksempler på overloading. Fyll inn @Override der det hører hjemme.

I klassen Person:

registerPerson(String) og  
registrerPerson(String, String) >  
**overloading**.

I klassen Student:

registrerPerson(String, String) og  
registrerPerson(String, String,String) >  
**overrider** klassen Person sine  
registrerPerson-metoder.

registrerPerson(String, String) og  
registrerPerson(String, String,String) >  
**overloading**.

skrivPerson() i Student **overrider** klassen  
Person sin skrivPerson().

Kode:

```
class Person {
    String navn = "ikke registrert.";
    String adresse = "ikke registrert.";

    public void registrerPerson(String n) {
        navn = n;
    }

    public void registrerPerson(String n, String a) {
        navn = n;
        adresse = a;
    }

    public void skrivPerson(){
        System.out.println("Navn: " + navn + ",
adresse: " + adresse);
    }
}

class Student extends Person {
    String idnr = "ikke registrert.";

    public void registrerPerson(String n, String i) {
        navn = n;
        idnr = i;
    }

    public void registrerPerson(String n, String i, String a) {
        navn = n;
        idnr = i;
        adresse = a;
    }

    public void skrivPerson(){
        System.out.println("Navn: " + navn + ",
studentnr: " + idnr + ", adresse: " + adresse);
    }
}
```

## 2.b - Opprettelse av person og studenter:

Opprett 3 personer. 1 av dem skal være en student. Velg selv hvilke metoder du vil benytte, men prøv ut litt forskjellig. Skriv deretter ned hva som skrives ut hvis du kaller på "skrivPerson()" på Person-objektene dine.

*Diskusjon: hva skrives ut her:*

```
Person a = new Student();
a.registrerPerson("Lise", "Holmveien 2");
a.skrivPerson();
```

*Diskusjon 2:*

*Hva skrives ut her:*

```
Person a = new Student();  
a.registrerPerson("Siri");  
a.skrivPerson();
```

*Diskusjon 3:*

*Hva skjer her:*

```
Person a = new Person();  
a.registrerPerson("Siri", "1235", "Trimveien  
9");  
a.skrivPerson();
```