Seminar uke 3

IN1010 V23 - Gruppe 9 prosa

srhellst på Discourse
Epost: srhellst@ifi.uio.no

UiO : Universitetet i Oslo

Arv og subklasser

- Java: kan arve instansvariabler og metoder fra en klasse til en annen (gjenbruke egenskaper)
- Arver egenskapene til klassen man arver fra + at Subklasse-objektene har flere / mer spesielle egenskaper
- Bruker `extends` for å arve fra en klasse
- **subclass** (child) - klassen som arver fra en annen klasse
- **superclass** (parent) - klassen man arver fra
- `protected` : tilgjengelig innenfor sin *package* (mappe i vårt tilfelle)
- `final` : “siste utgave”, gjør at andre klasser ikke kan arve fra en klasse
- `abstract` : gjør at man ikke kan opprette et objekt av klassen (typisk i en superklasse som er litt abstrakt definisert)

Arv og subklasser: annet

Referanser

- En referanse av typen P harbare lov å peke på objekter som har P-egenskaper
- Typen viser hvilke egenskaper man har tilgjengelig

Arv og subklasser: annet

Konverte / “caste” verdi

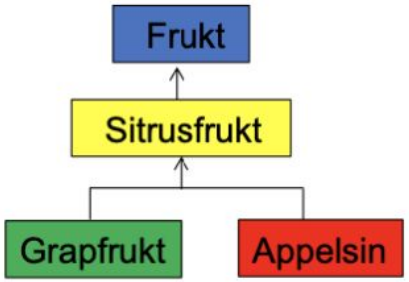
s = f; // Ikke lov

S = (Sitrusfrukt) f; // løse med konvertering

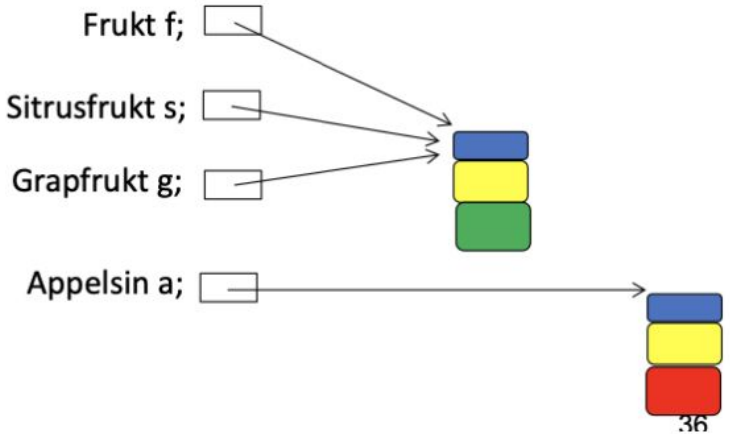
instanceof: sjekke hvilken type

```
if (f instanceof Sitrusfrukt) {
        s = (Sitrusfrukt)f;
}
```

Obs: kan ikke konverte et objekt som ikke har egenskapene til subklassen



Klassehierarki



Oppgave 1: Egenrefleksjon

Bruk et minutt på å skrive ned alt du husker om **arv og subklasser.** Det er en regel, og det er at man ikke får lov til å slutte å skrive - **3 min**

Oppgave 2: Diskusjon i grupper

```
class A {}
class B extends A {}
class C extends A {}
class D extends C {}

class TrueOrFalse {
        public static void main(String[] args) {
                A a = new A();
                B b = new B();
                C c = new C();
                D d = new D();
                A e = new D();
                A f = new B();

                System.out.println("a) " + (b instanceof A));
                System.out.println("b) " + (c instanceof D));
                System.out.println("c) " + (d instanceof A));
                System.out.println("d) " + (e instanceof C));
                System.out.println("e) " + (f instanceof C));
        }
}
```

true / false ?

Oppgave 3: Diskusjon i grupper

```
//Tilfelle 1
B ab = (B) a;

//Tilfelle 2
if( e instanceof B ){
   B eb = (B) e;
}

//Tilfelle 3
D ed = (D) e;

//Tilfelle 4
if( e instanceof D ){
   D ed = (D) e;
}

//Tilfelle 5
B fb = (B) f;

//Tilfelle 6
C ec = (C) e;
```

I hvert av tilfellene under diskuter om koden kommer eller ikke kommer til å kræsje, forklar hvorfor.

Vil dere si 3 eller 4 er den beste løsningen, og hvorfor

Oppgave 4: Programmering

Dere skal nå lage et program som visualiserer et **parkeringshus**. Et parkeringshus har parkeringsplasser hvor både Motorsykler og Biler kan parkere. **Motorsykler** og **Biler** har en del til felles, bl.a. har de begge en sjåfør og et registreringsnummer. Mens en bil også skal kunne holde på alt fra 2 til 5 personer (spesifisert i det bilen lages), derav en av de må være sjåføren.

Oppgave 4: Programmering

1. Tegn opp hvordan dette vil se ut (klassehierarki tegning)
2. Lag alle klassene med instansvariablene sine, enn så lenge skal de ikke ha noen metoder. **Hint:** Det kan være lurt at Motorsykkel og Bil arver av samme klasse, hvorfor tror du det?
3. Parkeringshus skal ha to metoder en for å sette inn en bil eller en motorsykkel og en for å fjerne en bil eller motorsykke på en gitt plass (Vi ønsker da at metoden skal returnere bilen/motorsyklen som står på plassen, hvis ingen motsykler er der skal null returneres). Når man skal sette inn en bil eller motorsykkel skal det også gjøres på en ønsket plass. Lag unntakshåndtering til parkeringshuset ditt, for å håndtere tilfeller der man prøver å sette inn et kjøretøy når det er fullt, eller på en ugyldig plass. **Hint:** ArrayIndexOutOfBoundsException