Oppgave 1 - Klasser og objekter

Følgende klasseskjelett er gitt:

```
class Kaningaard {

    private Kanin [] kaniner = new Kanin[100];

    public boolean full() { . . . }
    public boolean tom () { . . . }
    public Kanin finnEn(String navn) { . . . }
    public void settInn (Kanin kanin) { . . . }
    public void fjern(String navn) { . . . }
}
```

a. Lag en klasse Kanin. Kaninen har et navn som angis ved opprettelse. I tillegg har klassen en metode, hentNavn(), som returnerer navnet til kaninen.

b. Skriv ferdig klassen Kaningaard ved å fylle ut metodene i klasseskjelettet gitt ovenfor. Du kan gå ut fra at alle kaniner skal ha forskjellige navn. Ta derfor høyde for at det ikke er lov å sette inn en kanin to ganger (altså med samme navn som en i arrayen). Ta også høyde for noen andre problemer som kan oppstå og gi gode feilmeldinger. To eksempler på problemer er dersom man forsøker å sette inn en kanin i en full kaningård, eller når man forsøker å fjerne en kanin som ikke er i listen. Du kan anta at klassen Kanin inneholder metoden hentNavn().

c. Skriv et hovedprogram der du gjør minst tre tester av de forskjellige delene av klassen Kaningaard. Test for eksempel tilfeller som innsetting av kanin ved full kaningård eller fjerning av en ikke-eksisterende kanin.

d. Tegn datastrukturen slik den ser ut etter at du har kjørt hovedprogrammet. Hvorfor kan datastrukturtegninger være nyttige? Diskutér gjerne med en medstudent.

Oppgave 2 - Robotfamilien

Du lever i en verden der kunstig intelligens florerer. Enkelte roboter er mer høyteknologiske enn andre. Her skal vi ta for oss GPT, men merk at det kan eksistere roboter med navn som starter med noe annet enn GPT. Ved opprettelse av objekter, tar konstruktøren til hver robot inn et navn. Samtidig skal det ikke være mulig å opprette instanser/objekter av Robot. Det skal også finnes en metode som returnerer navnet til roboten.

GPT 3, GPT 3.5 og GPT 4 har alle noen delte egenskaper. For eksempel kommuniserer de med mennesker. Altså har de et navn og en samling *delte* egenskaper. I tillegg har GPT 3, GPT 3.5 og GPT 4 en samling *unike* egenskaper de ikke deler med de andre. Når vi snakker om samling, mener vi her enten array eller liste.

a. Tegn et klassehierarki over robotene i robotfamilien beskrevet ovenfor.
b. Implementér klassene med utgangspunkt i klassehierarkiet og informasjonen du har
c. Opprett metoder for å legge til og fjerne egenskaper fra de ulike robotene. Du kan anta at egenskaper som legges til ikke finnes fra før, og du kan anta at egenskaper som skal fjernes eksisterer.
d. Skriv og utfør enhetstester, som tester klassene og metodene du har implementert. Hvorfor er enhetstesting nyttig? Diskutér gjerne med en medstudent.

Oppgave 3 - Kjøretøy

Lag et program som administrerer en samling (HashMap) av kjøretøy. Programmet skal ha følgende klasser:

1. Kjøretøy-klasse:
    - Egenskaper: id (String), merke (String), modell (String), år (int)
    - Konstruktør: for å initialisere egenskapene
    - Hent-metoder og sett-metoder for egenskapene (kun hent-metode for id)
2. Bil-klasse (arver fra Kjøretøy):
    - Ekstra egenskap: seter (int)
    - Konstruktør: for å initialisere egenskapene (inkludert de arvede egenskapene fra Kjøretøy)
    - Hent-metode og sett-metode for egenskapen seter
3. Motorsykkel-klasse (arver fra Kjøretøy):
    - Ekstra egenskap: motortype (String)
    - Konstruktør: for å initialisere egenskapene (inkludert de arvede egenskapene fra Kjøretøy)
    - Hent-metoder og sett-metoder for egenskapene
4. Program-klasse:
    - Main-metode: Opprett en samling (HashMap) for å lagre kjøretøy. Legg inn to instanser (objekter) av Bil og to instanser (objekter) av Motorsykkel. Skriv ut detaljene for kjøretøyene i samlingen.

Oppgave 4 - Filinnlesning, objekter og arv

a. Opprett en fil kalt *ord.txt,* som inneholder ett ord på hver linje. Lag et program som skriver ut alle linjene slått sammen til én setning/linje. Legg til et mellomrom etter hvert ord som blir lagt til.

Eksempel på ord.txt

```
Dette
er
en
setning
paa
flere
linjer.
```

b. Utvid programmet i oppgave a, slik at brukeren kan skrive inn navnet på filen som skal leses ved oppstart av programmet.
c. Når brukeren skriver inn feil filnavn, vil det kastes et unntak. Brukeren skal få en passende beskjed når feil filnavn oppgis. Deretter skal programmet avsluttes.
d. For hvert ord som leses inn fra filen skal du nå opprette et Ord-objekt samt kalle på en instansmetode som skriver ut ordet.
e. Forestill deg at du er interessert i å skille mellom verbene og substantivene. Verbet i ordlista ovenfor er “er” og substantivene i ordlista overfor er “setning” og “linjer”. Hvorfor er det hensiktsmessig eller uhensiktsmessig å bruke arv når du er interessert i å kategorisere de ulike ordene? Diskutér gjerne med en medstudent.

Oppgave 5 - Array, ArrayList og løkker

- **a.** Lag et program med en array med plass til 10 heltall. Fyll arrayen med alle sifrene mellom 0 og 10 i stigende rekkefølge. Skriv ut alle verdiene på hver sin linje.
- **b.** Endre programmet slik at du får utskriften i motsatt rekkefølge. Denne gangen skal du bruke en ArrayList i stedet for en array.
- **c.** Endre programmet fra a slik at utskriften nå er tallene i arrayen eller ArrayList-en og summen av alle tallene som er skrevet ut til nå. Denne gangen skal du bruke en “for each”-løkke.
- **d.** Endre programmet slik at tallet blir gjort om til en streng og at tallene blir lagt til i en string som samler alle tallene per iterasjon. Her velger du om du vil ta utgangspunkt i arrayen eller ArrayList-en, og du velger mellom vanlig for-løkke og “for each”-løkke.

Utskrift:

```
> 0
> 01
> 012
> 0123
> 01234
> 012345
> 0123456
> 01234567
> 012345678
```

```
> 0123456789
```

e. Lag et program som oppretter en array med plass til 10 strenger. Fyll arrayen med *, hvor plass/indeks 0 har én * og hvor plass/indeks 9 har ti *-er. Det vil si at hver plass inneholder (plassnummer + 1) antall *-er. Skriv ut alle strengene på separate linjer.

Utskriften:

```
> *
> **
> ***
> ****
> *****
> ******
> *******
> ********
> *********
> **********
```