

1(a) **class Tidspunkt**

```
class Tidspunkt implements Comparable<Tidspunkt> {
    :
    Tidspunkt(int aar, int mnd, int dag, int time, int min, int sek) {
        :
    }
    :
}
```

Vi trenger en klasse for å lagre et tidspunkt, dvs dato og klokkeslett. Ta utgangspunkt i skjelettet vist over og fyll inn det som mangler.

Det er ikke nødvendig å skrive get-metoder for instansvariablene.

NB! Legg spesielt merke til at klassen skal implementere Comparable.

Maks poeng: 5

2(a) **Mor og far**

Hunder og hundekull

Du har en venn som arbeider med oppdrett av hunder, og du skal i dette oppgavesettet hjelpe henne med å skrive deler av et program for å holde orden på hunder og deres unger. En hund er definert ved klassen Hund som er beskrevet i vedlagte kode. Du skal ikke legge inn flere instansvariabler i klassen Hund, bare skrive innmaten i de angitt metodene. Når en tisper (hun-hund) får hvalper, kalles alle hvalpene født samtidig for et kull, og det er definert ved klassen Kull. Dette er en abstrakt klasse og du skal ikke endre den, men i oppgave 3 skal du skrive to subklasser til den. Klassen Kull har referanser til kullets mor og far, men for mange kull vil mor og/eller far være ukjent, og da inneholder referansen null. Klassen Tidspunkt ble skrevet som svar på oppgave 1. (Selv om du ikke skrev den, kan du likevel bruke den.)

Oppgave

Skriv metodene mor og far i klassen Hund. De skal returnere referanse til henholdsvis hundens mor og far. Om mor eller far ikke er kjent, returneres null. Husk at du i denne oppgaven ikke har lov å legge inn flere instansvariabler i klassen Hund.

Maks poeng: 5

2(b) **Sammenligning**

Skriv metoden i klassen Hund som gjør at Hund implementerer grensesnittet Comparable slik at hunder kan sammenlignes. En hund som er født før en annen, kommer også først i ordningen av hunder.

Maks poeng: 5

2(c) **Søsken**

Skriv de to metodene `erHelsosken` og `erHalvsosken`. To hunder er helsøsken om de har samme mor og samme far. To hunder er halvsøsken om de enten har samme mor eller samme far, men ikke begge deler.

Maks poeng: 7

2(d) **Eldste kjente opphav**

Skriv den rekursive metoden `finnEldsteKjenteOpphav`. En hunds opphav er dens mor og dens far samt deres mødre og fedre osv. Om både moren og faren til en hund er ukjent, er det eldste kjente opphavet hunden selv.

Maks poeng: 20

3(a) **KullListe.settInn**

Subklassene KullListe og KullArray

I oppgave 3 skal du programmere to subklasser av klassen `Kull` for å holde oversikt over hundene i et kull: `KullListe` og `KullArray`. Begge disse klassene skal ha en metode `settInn` for å legge nye hunder til et kull, men de skal ikke ha noen metode for å fjerne hunder. Metoden `iterator` skal lage en iterator for hundene i kullet, men du skal bare lage den for `KullListe`. (Klassen `KullArray` skal også inneholde en iterator-metode, men du skal *ikke* skrive den.)

I klassen `KullListe` skal alle hundene i kullet være lenket sammen i en liste, og du skal bruke variabelen `neste` i klassen `Hund` til dette. Når du itererer over hundene i kullet, skal de yngste komme først, så du må la listen være sortert slik at yngre hunder kommer foran eldre. Du kan ikke anta noe om rekkefølgen hundene blir lagt inn i kullet, så programmet ditt må sørge for at listen av hunder er sortert. I oppgavene 3a og 3b skal du til sammen skrive hele klassen `KullListe`, og i oppgavene 3c og 3d skal du skrive hele klassen `KullArray` (unntatt iteratoren i `KullArray`).

Oppgave

Programmer klassen `KullListe` med metoden `settInn`.

Maks poeng: 15

3(b) **KullListe.iterator**

Programmer resten av klassen `KullListe` med metoden `iterator`.

Maks poeng: 13

3(c) **KullArray**

I klassen KullArray skal hundene lagres i en array basert på fødselstidspunktet. Klassen skal inneholde en array med 60 elementer, og en hund som er født på sekundet 0 (dvs nøyaktig på et helt minutt, for eksempel kl 12:45:**00** eller kl 00:04:**00**) skal refereres av arrayelementet med indeks 0. Tilsvarende skal en hund som er født ett sekund over et minutt (for eksempel kl 05:17:**01** eller kl 22:06:**01**), refereres av arrayelementet med indeks 1 og så videre helt til de som er født 59 sekunder over et helt minutt.

Hvis to hunder i samme kull er født på samme sekund (for eksempel fordi den ene er født nøyaktig ett minutt etter den andre), skal neste-pekeren i klassen Hund brukes til å lenke disse sammen. Rekkefølgen i denne listen spiller ingen rolle, og metoden settlnn skal sette inn den nye hunden på enkleste måte, dvs først i listen.

Oppgave

Tegn på vedlagte skissepapir et objekt av klassen KullArray med 7 hunder med fødselstidspunkter som vist under. Det viktige er å vise kulletts datastruktur med array og referanser, så det er nok å vise alle detaljene for én hund.

Navn	Født
Adam	12.11.2018 kl 23:49:02
Benjamin	13.11.2018 kl 00:00:05
Caleb	12.11.2018 kl 23:58:58
Daniel	12.11.2018 kl 23:48:02
Ephraim	13.11.2018 kl 00:01:55
Frank	12.11.2018 kl 23:52:05
Gideon	12.11.2018 kl 23:55:05

Maks poeng: 7

3(d) **KullArray.settlnn**

Skriv klassen KullArray med metoden settlnn. (Husk at du ikke skal skrive metoden iterator for denne klassen.)

Maks poeng: 15

3(e) **KullArray.skrivUtAlle**

Skriv en metode skrivUtAlle i klassen KullArray. Denne metoden skal (ved å kalle på System.out.println) skrive ut navnene på alle hundene i kullet. Rekkefølgen spiller ingen rolle.

Maks poeng: 8

```

class Hund implements Comparable<Hund> {
    String navn;
    Kull mittKull;
    Tidspunkt minFodselstid;
    Hund neste = null;

    Hund(Kull k, String navn, Tidspunkt fodt) {
        this.navn = navn;
        mittKull = k;
        minFodselstid = fodt;
    }

    @Override
    public int compareTo(Hund h) {
        // Oppgave 2b
    }

    public Hund mor() {
        // Oppgave 2a
    }

    public Hund far () {
        // Oppgave 2a
    }

    public boolean erHelsosken(Hund h) {
        // Oppgave 2c
    }

    public boolean erHalvsosken(Hund h) {
        // Oppgave 2c
    }

    public Hund finnEldsteKjenteOpphav() {
        // Oppgave 2d
    }
}

```

```

abstract class Kull implements Iterable<Hund> {
    Hund mor, far;

    Kull (Hund mor, Hund far) {
        this.mor = mor;
        this.far = far;
    }

    public abstract void settInn(Hund h);
    public abstract Iterator<Hund> iterator();
}

```