Oppgaver - uke 7

Oppgave 1

Implementer klassen “Person”. En person skal ha et fornavn, et etternavn og en alder. La instansvariablene være private. Variabelverdiene skal settes i konstruktøren.

a) Implementer en toString-metode. La etternavnet komme før fornavnet i returstrengen.

b) Implementer en metode som henter alder.

c) Implementer metoden “public int compareTo(Person annen)”. Når vi sammenligner personer ønsker vi først og fremst å sortere på alder. Dersom alderen er lik, så ønsker vi å sortere på navn. *Hint: her kan du gjøre kall på både hentAlder() og toString().*

d) Test først programmet ditt med koden under. Diskusjon: Hva skjer med utskriften dersom du ikke kaller på Arrays.sort(personer) først? Hvorfor fungerer Arrays.sort(personer)?

```
import java.util.Arrays;

class Testprogram {
    public static void main(String[] args) {
        Person[] personer = {
            new Person("Andreas", "Hurthe", 25),
            new Person("Kristine", "Henriksen", 30),
            new Person("Henrik", "Golsen", 25),
            new Person("Sigvart", "Dagsland", 29)
        };

        Arrays.sort(personer);

        for (Person p : personer) {
            System.out.println(p);
        }
    }
}
```

Oppgave 2

Lag en generisk lenkeliste. Du kan ta utgangspunkt i kodeskjelettet gitt under. Benytt deg gjerne av Node-klassen fra forrige uke, men den skal nå flyttes inn i klassen Lenkeliste som en indre/privat klasse. Diskusjon: Hva slags endringer må gjøres på Node-klassen når du gjør den til en indre klasse? Lag et testprogram, der du oppretter en instans av lenkelisten du har implementert og tester alle de implementerte metodene.

```
public class FIFOListe<T> implements Iterable<T> {
        private Node hode;
        private Node hale;

        private class Node {}

        public int storrelse() {}

        public boolean erTom() {}

        public void leggTil(T t) {}

        public void fjernAlt() {}

        public T pop()

        public void skrivUtListe() {}

        @Override
        public Iterator<T> iterator() {}

        private class FIFOIterator implements Iterator<T> {

            Node denne = hode;

            @Override
            public boolean hasNext() {}

            @Override
            public T next() {}
         }
}
```

Oppgave 3

Du skal nå lage to varianter av en lenkeliste som holder på strenger; én enkeltlenket og én dobbeltlenket. Listene skal holde på en startnode og en sluttnode. Det skal være mulig å legge til nye verdier først i listene og sist i listene (leggTilForst; leggTilSist). Videre skal det være mulig å fjerne en vilkårlig streng. Verdien som skal fjernes skal oppgis som argument i fjern-metoden. Metoden skal returnere den fjernede strengen hvis den finnes og null ellers. Anta at listene dine ikke inneholder duplikater (ikke flere like strenger).

a) Implementer den *enkeltlenkede lista* med de nevnte metodene.
b) Implementer den *dobbeltlenkede* lista med de nevnte metodene.
c) Hva skiller leggTil-metodene i de to forskjellige listene fra hverandre?
d) Hva skiller fjern-metoden i de to forskjellige listene fra hverandre?
e) Når kan det være hensiktsmessig å bruke dobbeltlenket fremfor enkeltlenket liste?