INF1000 - Obligatorisk innlevering 5

Temaer denne uka: *Klasser og objekter*.

Oppgave 5.1)

Tema: *Klasser, objekter*
Filnavn: `Mobil.java, Person.java, Oppgave51.java`

a) Programmer en klasse Mobil og gi den variabler for merke, eier, og telefonnummer. Lag deretter metoder for å lagre data i variablene. Opprett et objekt av klassen i en annen klasse, `Oppgave51`, som inneholder main-metoden. Fyll objektet med data.

b) Programmer en klasse Person som inneholder variabler for navn, alder og bosted. Lag også metoder for å sette informasjonen i objektet.

c) Opprett et objekt av klassen Person i main-metoden i klassen `Oppgave51`, og sett passende verdier for navn, alder og bosted fra main-metoden.

d) Endre nå klassen Person slik at personen får en mobiltelefon dersom personen er over 14 år. Hint: Husk å lage et objekt av klassen Mobil før du gir den til personen.

Synes du denne oppgaven var vanskelig? Se øvingsoppgaver 5.1.1, 5.1.2 og 5.1.3.

Oppgave 5.2)

Tema: *Teorioppgave*
Filnavn: `Teori.txt`

Gi korte svar på spørsmålene under:

a) Hva er innkapsling? Hvorfor er det nyttig?

b) Hva er det grensesnittet til en klasse? Hvordan skiller det seg fra implementasjonen av en klasse?

c) Hva er en instansmetode og hvordan skiller den seg fra en statisk metode (slike metoder vi har jobbet med tidligere)?

Oppgave 5.3)

Tema: *String, tekstbehandling*
Filnavn: `Tekster.java`

a) Lag et program med en variabel som inneholder teksten "Agnes i senga". Bruk en løkke til å printe ut teksten i variabelen baklengs.

b) Lag en variabel som inneholder teksten "INF1000" og en variabel som inneholder teksten "inf1000". Lag en test for om disse variablene er like.

c) Lag deretter en variabel som inneholder teksten "INF1100 INF1000 INF1010". Lag programmet ditt sånn at du henter ut kun "INF1000" og lagrer dette i en ny variabel som du skriver ut til terminal.

d) Lag en variabel som inneholder teksten "Espen Askeladd". Les om `String.toCharArray` under String sin dokumentasjon og bruk denne meotden til å lagre variabelens innhold i en array av bokstaver. Gå så gjennom denne arrayen og hver gang du ser bokstaven 'a' skal du bytte ut bokstaven med 'A'.

Synes du denne oppgaven var vanskelig? Se øvingsoppgaver 5.3.1 og 5.3.2.
Synes du denne oppgaven var lett? Se utfordringsoppgaver 5.3.3.

Oppgave 5.4)

Tema: *Klasser og objekter*
Filnavn: `Bondegard.java, Ku.jave, Gris.java, Hest.java`

Oppgaven går ut på å modellere en enkel bondegård. Vi har en grisebinge med plass til 10 griser, en stall med plass til 5 hester og et fjøs med plass til 30 melkekuer.

a) Lag en klasse Bondegård som inneholder:

```
Gris[] grisebinge = new Gris[10];
Hest[] stall = new Hest[5];
Ku[] fjos = new Ku[30];
```

Lag også de tre klassene: Gris, Hest og Ku. Hvert dyr har et unikt navn. Hver av disse klassene skal ligge i en egen fil.

b) Lag tre metoder i klassen Bondegård:

- settInnGris(Gris g);
- settInnHest(Hest h);
- settInnKu(Ku k);

Disse metodene skal sette inn dyr i arrayene. Merk at du bare kan sette et dyr inn i arrayen hvis arrayen ikke er full.

c) Lag deg en main-metode der du fyller grisebingen, stallen og fjøset med riktige dyr. Du skal fylle inn 5 griser, 2 hester og 8 kyr. Hvert dyr skal få et unikt navn.

d) Skriv en metode `selgDyr(String type, int antall)` som selger *antall* dyr av type *type*. Disse dyrene må då slettes fra grisebingen, stallen eller fjøset.

e) Selg 3 griser, 1 hest og 4 kyr.

f) Lag en metode `skrivUt` som skriver ut alle dyrene i bondegården.

g) Utvid programmet ditt med en ordreløkke. Her skal du tilby brukeren følgende kommandoer:

```
0: Avslutt
1: Sett inn en gris
2: Sett inn en hest
3: Sett inn en ku
4: Selg et dyr
5: Skriv ut bondegården
```

Hint: En ordreløkke lager du enklest ved å lese inn kommandoer fra brukeren i en løkke. Du fortsetter å lese inn nye kommandoer, helt til brukeren taster '0' (avslutt):

```
int valg = -1;
while (valg != 0) {
    System.out.println("Gi et valg: ");
    valg = Integer.parseInt(scan.nextLine());
    if (valg == 1) {
        lesInnGris();
    } else if (valg == 2) {
        lesInnHest();
    } // osv.
}
```

For hver kommando har du så en tilhørende metode som leser inn nødvendig informasjon fra brukeren og utfører kommandoen. Et enkelt eksempel på dette:

```
void lesInnGris() {
    System.out.println("Skriv navnet på grisen: ");
    String navn = scan.nextLine();
    // opprett en gris med navn 'navn' og sett den inn i bondegården
}
```

Oppgave 5.5)

Tema: *Egen oppgave*
Filnavn: `MinOppgave5.java`

Lag din egen oppgave der du bruker klasser og objekter. Både oppgaven og besvarelsen skal leveres inn.

Fremgangsmåte for innleveringer i INF1000

1. Lag en fil som heter README.txt. Følgende spørsmål skal være besvart i filen:
   - Hvordan synes du innleveringen var? Hva var enkelt og hva var vanskelig?
   - Hvor lang tid (ca) brukte du på innleveringen?
   - Var det noen oppgaver du ikke fikk til? Hvis ja:
     - Hvilke(n) oppgave er det som ikke fungerer i innleveringen?
     - Hvorfor tror du at oppgaven ikke fungerer?
     - Hva ville du gjort for å få oppgaven til å fungere hvis du hadde mer tid?
2. Logg inn på Devilry.
3. Lever de 8 .java-filene, samt teori.txt, README.txt og filene du bruker i den egne oppgaven din i *samme innlevering*.
4. Husk å trykke lever og sjekk deretter at innleveringen din er komplett.

Den obligatoriske innleveringen er minimum av hva du bør ha programmert i løpet av en uke. Du finner flere oppgaver for denne uken her og flere utfordringsoppgaver her.