AAU, Programmering i Java
Intern skriftlig prøve
18. maj 2007

Opgavebesvarelsen skal afleveres som enten en printerudskrift eller som et passende dokument sendt via email til fjj@noea.dk.

Besvarelsen skal være forsynet med navn, personnummer, gruppenummer samt uddannelsesretningen.

Besvarelsen skal være afleveret senest den 18.maj 2007.

Pensum:
Pensum beskrives i forhold til lærebogen: Introduction to Programming Using Java, Fifth Edition, David J. Eck.

Pensum er følgende afsnit i bogen:

- Afsnit 1.1-1.5
- Afsnit 2.1-2.6
- Afsnit 3.1-3.7
- Afsnit 4.1-4.5
- Afsnit 5.1-5.7
- Afsnit 7.1-7.5
- Afsnit 9.1

Opgave 1

Implementér en klasse (f.eks. Opgave1) som indeholder følgende metoder fra opgave 1.1 til opgave 1.4. Metoderne skal afprøves via en main metode på samme klasse.

Opgave 1.1

Implementér en statisk metode ud fra følgende:
Som parameter anvendes to decimaltal *a,b* af typen *double*. Metoden returnerer kvadratroden af *a*a+b*b*.

Opgave 1.2

Implementér en statisk metode med følgende egenskaber:
Som parameter anvendes to heltal af typen *int*.
Hvis begge heltal er negative returnerer metoden 0 og ellers returnerer metoden summen af de to heltal.

Opgave 1.3

Implementér en statisk metode med følgende egenskaber:
Som parametre anvendes tre heltal, metoden skal returnere det største af de tre tal.

Opgave 1.4

Implementér en statisk metode med følgende egenskaber:
Som parameter anvendes datatypen *char*. Hvis parameteren indeholder et af følgende små bogstaver (vokaler): a, e, i, o, u, y, æ, ø og å, returnerer metoden *true* og ellers returnerer metoden *false*.

Opgave 1.5

Følgende kode er blevet implementeret

```
public interface Person
{
    public String getNavn();
}

public class Ansat implements Person
{
    private String navn;
    private double løn;
    public double getLøn()
    {
        return løn;
    }
}
```

Når ovenstående kode kompileres modtager programmøren følgende fejl:

```
Ansat.java:1: Ansat should be declared abstract; it does not define getNavn() in Ansat
public class Ansat implements Person{
```

Men det er ikke meningen at klassen Ansat skal være abstrakt. Hvad er problemet?

Opgave 2

Implementér en klasse (f.eks. Opgave2) som indeholder følgende statiske metoder:

public static double genOver20(int[] array)
// Metoden returnerer gennemsnittet af alle de tal i arrayet *array*, der er større end 20

public static void opdaterAlle(int[] array, int x, int y)
//Metoden erstatter alle forekomster af *x* i *array* med *y*
//Hvis f.eks. *array* indeholder [12 , 7 , 4 , 6 , 2 , 6 , 1 , 2 , 3], og der kaldes *opdaterAlle(2,77)* skal
//arrayet efter kaldet indeholde [12, 7 , 4 , 6 , 77 , 6 , 1 , 77 , 3].

public static int[] returnNegative(int[] array)
//Metoden returnerer et array af heltal, som indeholder alle de tal i *array*, der er negative.
//Hvis f.eks. *array* indeholder elementerne [2 , -3 , 1 , -12 , 5] skal metoden returnerer et array med
//elementerne [-3, -12]

public static boolean containsX(int[] array, int x)
//Metoden returnerer true, hvis der er en forekomst af tallet *x* i *array* ellers false

public static int countX(int[] array, int x)
//Metoden returnerer antal af gange *x* forekommer i arrayet *array*

Metoderne skal afprøves via en main metode på samme klasse

Opgave 3

Denne opgave drejer sig om fletning af to sorterede lister. En fletning tager to sorterede lister som input og producerer ud fra disse en ny sorteret liste.

Afhængigt af formålet er der forskellige typer af fletninger, bl.a. *total fletning*, hvor den nye liste indeholder alle elementer fra de to inputlister.

Fx vil den totale fletningen af:

```
a = {1,2,3,4,6,8,12,15};
b = {2,4,5,6,7,9,11,12,17,18};
```

blive

```
{1,2,2,3,4,4,5,6,6,7,8,9,11,12,12,15,17,18}
```

I denne opgave er listerne blot arrays med heltal som elementer.

Neden for er der en test-klasse, som kalder en metode *totalFlet*.

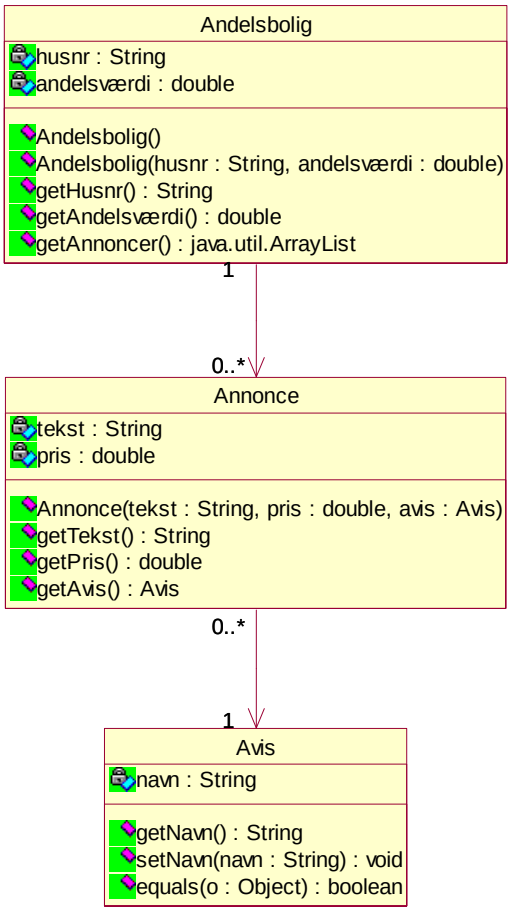
Implementér denne metode i henhold til ovenstående

```
class Flet
{
     public static void main(String[] args)
    {
        int[] a = {1,2,3,4,6,8,12,15};
        int[] b = {2,4,5,6,7,9,11,12,17,18};
        int[] c = totalFlet(a,b);
    }
}

public static int[] totalFlet(int[] a, int[] b){
//Antag at a og b er sorterede. Metoden returnerer den totale fletning af a og b
```

Opgave 4

Hos Andelsboligforeningen Bytoften kan man købe og sælge andelsboliger. Når en andelsbolig sættes til salg bestilles der annoncer hos forskellige aviser i lokalområdet.
Nedenstående viser et udsnit af systemet beskrevet ved et såkaldt klassediagrammet:



Opgave 4.1

Som det fremgår af klassediagrammet er en avis modeleret ved klasse Avis med følgende klasse medlemmer:

en private instansvariabel *navn* af typen String
set og get metoder til instansvariablen *navn*
samt metoden equals, der override(r) metoden equals fra klassen java.lang.Object.

Implementér klassen Avis

Opgave 4.2

Som det endvidere fremgår af klassediagrammet er en annonce modeleret ved klasse Annonce med følgende klasse medlemmer:

en private instansvariabel *tekst* af typen String

en private instansvariabel *pris* af typen double
en private instansvariabel *avis* af typen Avis (fremgår ikke direkte af figuren)
get metoder til alle instansvariablerne *tekst, pris, avis*
En constructor som initialiserer alle instansvariablerne.

Implementér klassen Annonce

Opgave 4.3
Klassediagrammet viser også en klasse Andelsbolig. Klassen repræsenterer naturligvis andelsboliger i forening og har i systemet ansvaret for at gemme oplysninger om hvilke annoncer der er knyttet til en andelsbolig bestilt i forbindelse med foreningens salgsaktiviteter.
Klassen har følgende klasse medlemmer:

en private instansvariabel *husnr* af typen String
en private instansvariabel *andelsværdi* af typen double
get metoder til instansvariablerne *husnr* samt *andelsværdi*
en default constructor
en constructor som initialiserer instansvariablerne *husnr* samt *andelsværdi*

Endvidere, har klassen en private instansvariabel *annoncer* af typen java.util.ArrayList, som skal anvendes til at opbevarer referencer til annonce objekter, der er knyttet til netop denne andelsbolig. (Listen fremgår ikke direkte af diagrammet)

Slutteligt, har klassen en instansmetode *public java.util.ArrayList getAnnonce()*, som er en get metode til instansvariablen *annoncer*.

Implementér klassen Andelsbolig

Opgave 4.4
Når en ny annonce skal oprettes i systemet overføres annonceteksten, annonceprisen samt avisreferencen til en metode på klassen Andelsbolig. Herefter er det andelsboligens ansvar at oprette annoncen og gemme denne i sin liste.
Implementér følgende metode på klassen Andelsbolig:

```
public void opretAnnonce(String tekst, double pris, Avis avis)
        //En annonce er oprette med tekst og pris.
        //Annoncen har referencen til avisen avis
        //Annoncen er gemt i andelsboligens annonceliste
```

Opgave 4.5
Foreningens regnskab ”kræver” en metode, som beregner den samlede annonceudgift knyttet til denne bolig. (Dvs. summen over de enkelte annoncepriser)
Implementér følgende metode på klassen Andelsbolig:

```
public double getAnnonceudgift()
//metoden returnerer den samlede annonceudgift knyttet til denne andelsbolig
```