Oblig4Hybelhus (INF1000 - Høst 2013) – én av to mulige oppgaver for oblig4 (du løser denne *eller* Oblig4Pi)

Gulbrand Grås Husleiesystem

Mål: *Formålet med oppgaven er å gi erfaring med å løse et større programmeringsproblem ved hjelp av klasser og objekter, dessuten trening i filbehandling, arrayer, metoder, og brukerinteraksjon via terminal.*

Levering

Frist for innlevering er fredag 25. oktober 2013 kl. 23.59 i Devilry. Krav til innlevering generelt av obliger på Ifi (MÅ LESES) finner du her. Besvarelsen leveres som én .java fil (alle klassene i en fil) til Devilry. Besvarelsen leveres som én .java fil (alle klassene i en fil) til Devilry. Husk alle skal levere hver sin oppgave, men hvis dere vil, kan to samarbeide om deler av besvarelsen. Begge skal da levere hver sin kode i Devilry som på noen, men ikke alle punkter kan være ganske like. I første linje i .javafilen skal det da være en kommentarlinje som sier hvilke to som evt. har samarbeidet og om hva – f.eks:

```
// Samarbeidet med:martinja gruppe 2 om valg 5 og 6 i menyen
```

Tips og starthjelp – krav til løsning.

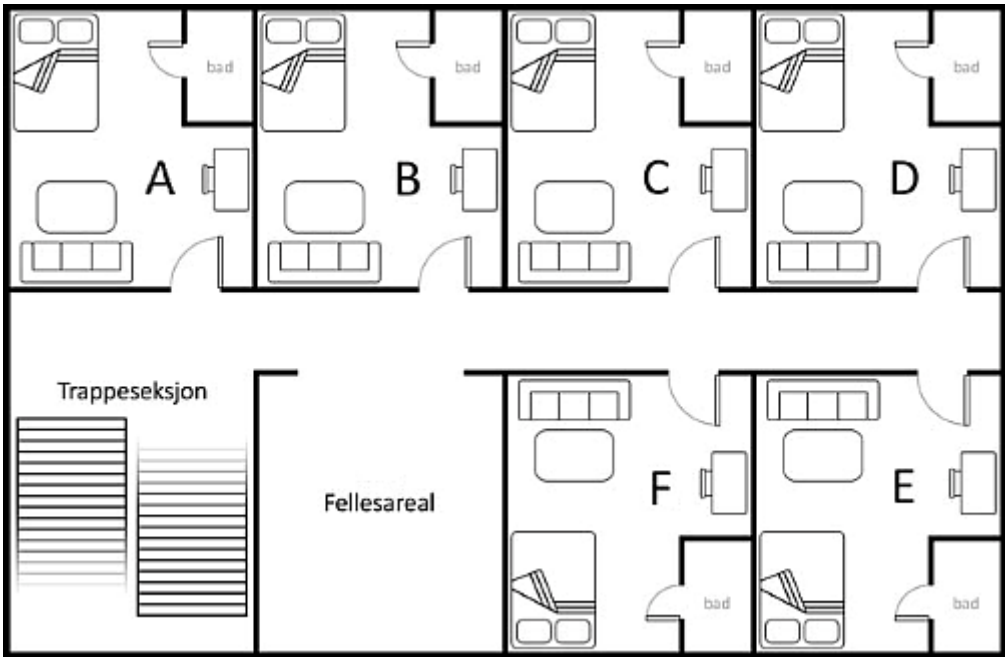
Løsningen skal omfatte minst 3 klasser. Det er gode løsninger med 3 klasser(Oblig4, Hybelhus og Hybel), med 4 klasser(Oblig4, Hybelhus, Etasje og Hybel) og med 5 klasser(Oblig4, Hybelhus , Etasje, Hybel og Student). Det finnes helt sikkert også andre gode løsninger; du må selv velge – og det er ikke sikkert at den med færrest antall klasser er enklest å lage. Flere tips finner du her etter 19 okt.

Det du skal leveres i Devilry er javafilen med løsningen din/deres og filen :hybeldata.txt etter at du har kjørt minst 10 innbetalinger av husleie og at Gudbrand har kjørt to månedskjøringer (se nedenfor) samt økt husleien en gang mellom de to månedskjøringene.

Oppgave

Gulbrand Grå har et hybelhus kalt Utsyn, med 18 studenthybler som han leier ut til studenter i Oslo med den hensikt å tjene godt. Du skal lage et system for å administrere utleie av hyblene i hybelhuset. Utsyn har 3 etasjer, nummerert fra 1 til 3. I hver etasje er det 6 hybler, kalt rom A til F, og et fellesrom. Hver hybel har et entydig «hybelnavn» som består av etasjenummer og rom-bokstav, f.eks. heter hybelen i rom C i andre etasje "2C".

Nedenfor ser du en illustrasjon av en etasje. Alle etasjene har samme planløsning.



Økonomi

Gulbrand Grå leier ut hyblene i de to nederste etasjene for 7500 kroner i måneden, mens husleien for en hybel i toppetasjen er 9000 kroner på grunn av utsikten.

Gulbrand har hyret inn firmaet Vedlikehold A/S som tar seg av alle løpende utgifter: Vedlikehold, reparasjoner, kommunale avgifter, nettabonnement i hyblene, og utstyr og strøm til fellesarealer. For dette betaler Gulbrand hver måned: 1200 kr per hybel A-F i hver etasje, uansett om en hybel har beboer eller ikke, pluss 1700 kr per etasje for fellesarealer. Beboerne betaler husleie for sine hyblene til Gulbrand.

Datafilen "hybeldata.txt"

Når programmet starter (før hovedmenyen skrives ut på skjermen), skal programmet lese datafilen ”hybeldata.txt”. Her er det lagret blant annet informasjon om hvor lenge systemet har vært i drift og navn på de nåværende leietagerne. Første linje i filen inneholder seks heltalls-verdier adskilt med semikolon:

```
int måned; int år; int totalfortjeneste; int totaltAntallMåneder; int månedsleieVanligHybel; int månedsleieToppEtasjeHybel.
```

Her er "`int måned`" og "`int år`" månedsnummeret og årstallet da månedskjøring sist ble utført, hvor måned er et tall i området 1 til 12, og årstallet er firesifret. Det tredje tallet er Gulbrands totale fortjeneste siden systemet ble satt i drift; og det fjerde tallet angir antall måneder systemet har vært i drift. Tall fem og seks er selvinnlysende, og er hhv. 7500 og 9000 første gang systemet kjøres.

Siden systemet skal settes i drift 1. november 2013 (når du sikkert er ferdig med systemet), beskriver den filen du får tak i hybeldata.txt et hybelhus med de som bodde i huset da ditt system settes i drift, beboere og deres saldo, men uten noen fortjeneste.

Deretter er det 18 linjer, en for hver hybel, med følgende format:

```
int etasje; char bokstav; int saldo; String studentnavn.
```

For tomme hybler skal studentnavnet lagres i datafilen som `TOM HYBEL`, med saldo `0`. Du kan anta at alle studenter har unike navn. Alle disse dataene skal holdes oppdatert internt i programmet mens det kjører, og skal skrives tilbake til datafilen når brukeren utfører ordre '8' i hovedmenyen ("Avslutt"). Slik kan Gulbrand starte og avslutte programmet uten å miste data.

Menyvalgene

Programmet skal være menystyrt. Det skal skrive ut på skjermen en meny over mulige ordre og be bruker om å taste en av disse. Hvis brukeren taster inn en ulovlig ordre, skal det gis feilmelding. Programmet skal gå i løkke og fortsette å lese og utføre ordre helt til brukeren taster ordre '8' for å avslutte. Bruker skal kunne gi 8 ordre:

1. Skriv oversikt
2. Registrer ny leietager
3. Registrer betaling fra leietager
4. Registrer frivillig utflytting
5. Månedskjøring av husleie
6. Kast ut leietagere
7. Øk husleien.
8. Avslutt

Detaljert beskrivelse av ordrene:

1. **Skriv oversikt**
   På denne ordren skal programmet skrive ut en oversikt over alle hyblene, som viser for hver hybel: hybelnavn, leietager-navn, og saldo. Dersom hybelen er ledig, skal teksten LEDIG skrives ut i stedet for leietager-navn, og saldoen vises som 0. Til slutt skal nåværende måned, år, antall måneder systemet har vært i drift, og totalfortjeneste skrives ut på skjermen.

Eksempel på hvordan oversikten kan se ut:

```
Hybel Leietager               Saldo
----- --------------------- -------
1A    Ole Johan                7500
1B    Guri Smith              15000
1C    ( LEDIG )                   0
           ...osv...
Måned/år, og driftstid:  9/2010,  24 mnd. i drift
Totalfortjeneste:        1400500 kr
```

2. **Registrer ny leietager**
   Ordren brukes når en student ønsker å flytte inn og leie en av hyblene. Først sjekker programmet om det finnes ledige hybler, hvis ikke skriver det en melding om det og returnerer til hovedmenyen.

   Hvis det finnes ledige hybler skal hybelnavnene til disse (f.eks 1C, 2B) skrives ut på skjermen, og så skal programmet spørre hvilken av disse studenten ønsker å leie. Bruker skal kunne taste inn ønsket **hybelnavn** som 1C, 2B, osv. Er valgt hybel ledig, skal programmet spørre om studentens navn, og registrere innflyttingen.

   Studenten som flytter inn betaler samtidig et depositum på 15 000 kroner. Fra dette trekkes det med én gang månedsleien for den første måneden (husk at månedsleien er avhengig av etasje). Det som er til overs blir studentens **saldo** (beholdningen i en slags forenklet bankkonto som leietagerne har hos Gulbrand). Når de betaler inn for husleie legges beløpet til i saldoen, og ved "månedskjøring" blir husleien trukket fra saldo. Saldoen vil være negativ hvis studenten skylder Gulbrand penger. Han trekker alltid én hel månedshusleie ved innflytting, uansett hvilken dag i måneden studenten flyttet inn.

   Programmet skal til slutt skrive ut en beskjed på skjermen om at innflyttingen ble gjennomført. Beskjeden skal inneholde hybelnavnet (etasje+bokstav), studentens navn, og gjenværende saldo.

3. **Registrer betaling fra leietager**
   Programmet skal spørre om et hybelnavn og beløpet som betales. Hvis hybelen som ble oppgitt ikke har beboer skal det skrives en feilmelding, ellers skal beløpet adderes til studentens saldo og en passende melding skrives på skjermen. Hver student må passe på å ha nok penger på sin saldo til å dekke husleien hver måned.

4. **Registrer frivillig utflytting**
   Programmet spør om *navnet på studenten* som ønsker å flytte ut, og leter deretter gjennom hyblene etter dette navnet. Hvis studenten ikke ble funnet, skal det gis en feilmelding, ellers skal det registreres i systemet at hybelen ikke lenger har beboer. Det forutsettes at leie er betalt for inneværende måned, og det utbetales ikke noe restbeløp selv om leietaker flytter før månedsslutt, men selvsagt refunderes beløpet som gjenstår på saldo etter at inneværende måned er betalt fullt ut. Dette beløpet skrives ut.

5. **Månedskjøring av husleie**
   Gulbrand utfører denne ordren første dag i hver måned. Programmet ber brukeren bekrefte at det skal startes månedskjøring for måneden etter forrige månedskjøring. Hvis siste månedskjøring ble utført for måned 9 i år 2013, kan spørsmålet være "Ønsker du å utføre månedskjøring for måned 10/2013 (j/n)?". Svarer brukeren `n`, returnerer programmet til hovedmenyen.

   Svarer brukeren `j`, skal månedsnummer og evt. årstall oppdateres, og de månedlige overføringene utføres.

Programmet går gjennom alle hyblene: For hver hybel som har beboer trekkes månedsleien for hybelen fra studentens saldo og legges til Gulbrands månedsfortjeneste. Gulbrand belaster altså husleien forskuddsvis, for måneden som nettopp har begynt. Hvis noen leietagere ikke hadde nok i saldoen går denne i minus, men Gulbrands fortjeneste økes bare med det som var på saldo (han får inn resten av fortjenesten som forklart i menyvalg 6).

Gulbrands utgifter er det han betaler til Vedlikehold A/S, og dette trekkes fra månedsfortjenesten hans. Husk at det er forskjellige utgifter for hybler og fellesarealer.

Til slutt skal følgende skrives til skjerm:

a. **Måned/år** som månedskjøringen gjelder for; **og driftstid** i antall måneder systemet har vært i drift, inkludert den nye måneden.
b. **Husleiesatsene** for en hybel (toppetasjen og de andre)
c. **Månedens fortjeneste** er Gulbrands inntekter minus utgifter i denne månedskjøringen. Hvis du ønsker å vise andre inntekter fra innflyttinger eller annet siste måned, så skriver du dette ut som en egen post, den vanlige "månedens fortjeneste" skal kun vise regnskapet for månedskjøringen forklart ovenfor.
d. **Totalfortjeneste** er Gulbrands nye totalfortjeneste, oppdatert med denne månedens fortjeneste.
e. **Gjennomsnittlig månedsfortjeneste** regnes selvsagt ut som: totalfortjeneste / totaltAntallMåneder.

6. **Kast ut leietagere**
Leietagere som skylder mer enn én husleie, blir kastet ut ved hjelp av torpedoen H. Hole når denne ordren utføres. Pengekravet til studenten som kastes ut er det han skylder i husleie pluss et utkastingsgebyr på 3000 kroner. Hole og Gulbrand deler gebyret likt. Gulbrands halvdel av gebyret pluss det studenten var i minus på saldoen legges til totalfortjenesten med én gang menyvalget kjøres (siden Hole alltid ordner disse sakene raskt), og hybelen registreres som ledig.

Programmet går gjennom alle hyblene og finner studentene med saldo lavere enn minus én månedsleie (husk de forskjellige leieprisene!). For hver av disse studentene skal du kalle følgende hjelpemetode (som du også skal programmere)

```
void tilkallHole(int etasje, int rom, int krav) { // ...
```

Denne metoden skriver hybelnavn, studentnavn og pengekrav til skjerm og til filen **`torpedo.txt`**. Metoden skal ikke overskrive det som ligger på filen fra før, men legge til en ny linje på slutten.'

7. **Øk husleien**
Gudbrand ser for seg at hvis hybelmarkedet blir enda mer vanskelig for studenter, kan han jo øke husleien (for å tjene litt mer). Her skal du altså skrive ut de to

husleiesatsene som nå gjelder og spørre Gudbrand hva de nye månedsleiene skal være. Ny husleie gjelder innværende måned og fremover – altså slik at neste gang et månedoppgjør kjøres, gjelder de nye satsene.

8. **Avslutt**
   Ved utførelse av denne ordren skal nødvendige data skrives til `hybeldata.txt`: måned, år, totalfortjeneste, antall måneder i drift, samt leietager og saldo for alle hyblene. Deretter skal programmet avslutte.

Hjelpemetoder
Her er fem hjelpemetoder med inn- og ut-parametre som vil være nyttige hvis løsningen din både har klassen Student og klassen Hybel. Programmér disse og eventuelle andre du får bruk for og benytt dem i programmet (se tips ved behov).

```
Hybel spørOmHybel(String ledetekst) {
    // Skriver ut "ledetekst" på skjermen, leser et hybelnavn fra
    // tastatur, og returnerer den tilhørende Hybel-pekeren.
}
String finnHybelnavn(int etg, char rom) {
    // Konverterer etg. og rom til et hybelnavn.
 }
Student finnBeboer(String hybelnavn) {
    // Finner og returnerer peker til leietageren som leier hybelen
    // angitt i inn-parameteren "hybelnavn".
 }
String stortTallTilString(int tall) {
   // Lager en tekst med en mer lettlest form av inn-parameteren
   // "tall" med ett mellomrom for hvert 3. siffer (bakfra),
   // f.eks. 1400500 skal gi "1 400 500".
}
```

Hvis du har spørsmål, kommentarer, eller finner feil i oppgaveteksten møt opp på INF1000-seminaret til FUI lørdag 19. okt (og spør gruppelæreren din)!

Lykke til!

Tips: kode-skisse med 4 klasser

```
// Skriv en kommentar om besvarelsen din her: ...
// ...
import easyIO.*;

class Oblig4 {
    public static void main(String [] args) {
        Utsyn s = new Utsyn();
        s.ordrelokke();
    }
}

class Student {
    String navn; // studentens navn
    int saldo;   // studentens saldo

    // Evt. metoder for å behandle en Student...
}

class Hybel {
    Student leietager;  // peker på et Student-objekt
    int husleie; // 6000 hvis hybelen er i 3. etasje, ellers 5000.

    // Evt. metoder for å behandle en Hybel...
}

class Utsyn {
    In tast = new In();
    Out skjerm = new Out();

    Hybel[][] hybler = new Hybel[3][6];
    // Variabler for økonomidata kan legges inn her...

    // Konstanter:
    final String FILNAVN = "hybeldata.txt";
    final String TOM_HYBEL = "TOM HYBEL";
```

```
    // Konstruktør for klassen Utsyn
    Utsyn() {
     // Her kan du lese datafilen "hybeldata.txt" og lagre hele innholdet
     // i datastrukturene dine.  Husk å opprette Hybel- og Student-objekter.
     // Eksempel på innlesing av en saldo, for en gitt "etg" og "rom":
     //    hybler[etg][rom] = new Hybel();
     //    hybler[etg][rom].leietager = new Student();
     //    hybler[etg][rom].leietager.saldo = innfil.inInt(" ;");
    }

  void ordrelokke() {
        int ordre = -1;
        while (ordre != 0) {
            // Skriv ut meny:
            skjerm.outln("Meny:");
            skjerm.outln("1. ..."

            // Les ordre fra bruker
            skjerm.out("Ordre: ");
            ordre = tast.inInt();

            switch(ordre) {
                case 1: skrivOversikt(); break;
                case 2: registrerNyLeietager(); break;
                case 3: registrerBetaling(); break;
                case 4: // ...
                // ... Fyll ut resten...
                default: // Gi feilmelding.
            }
        }
    }

    // Metoder for de forskjellige ordrene i ordrelokke()

    void skrivOversikt() { /* ... */ }
    void registrerNyLeietager() { /* ... */ }
    void registrerBetaling() { /* ... */ }

    // ... Fyll ut med minst 5 metoder til
}// end class Utsyn
```