Oblig3 - obligatorisk oppgave nr. 3 (av 4) i INF1000

Ærlige Johans husleiesystem

Leveringsfrist

Oppgaven må leveres senest fredag 21. oktober kl 16.00. Viktig: les slutten av oppgaven for detaljerte leveringskrav. **N.B**. Det presiseres at du minimum skal ha de 4 klassene beskrevet i hintene for å få godkjent løsningen din. En løsning uten disse klassene godkjennes ikke.

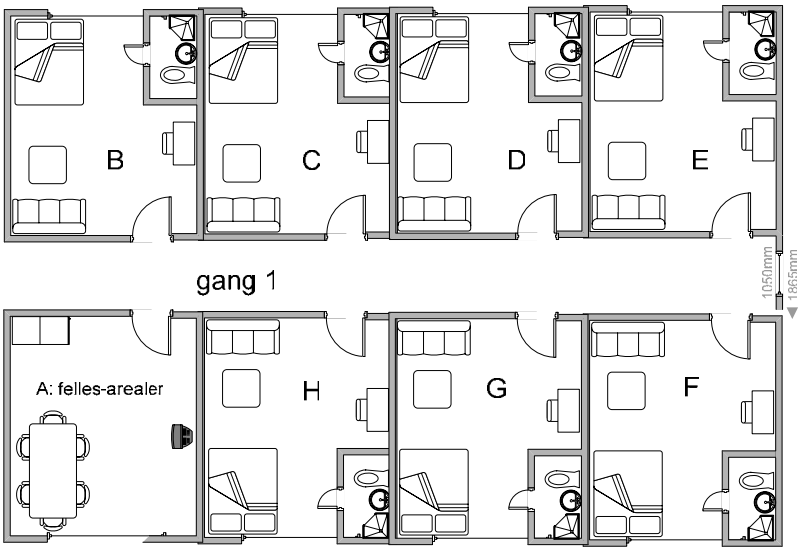
Formål

Trening i å løse et litt større programmeringsproblem ved å kombinere de ulike elementene vi har sett på til nå (arrayer, metoder, klasser/objekter, brukerinteraksjon og filbehandling).

Oppgave

I likhet med de fleste andre universitetsbyer har Ruritanias hovedstad Uqbar dessverre fortsatt mangel på studenthybler. Ærlige Johan og hans firma HaiHus A/S lever av denne mangelen. Du skal hjelpe Ærlige Johan med å lage et system for å administrere utleie og effektiv utkasting av studenter som ikke betaler.

I denne oppgaven skal vi se på hans enetasjes studenthus Utsyn. Utsyn består av 4 separate ganger, hver med 7 hybler + fellesarealer. Hver hybel har et unikt hybelnavn som består av gangnummeret (1 - 4) etterfulgt av en bokstav (B,C,D,E,F,G eller H). Fellesarealet i hver gang har bokstaven A. Dermed er f.eks. 2A fellesarealet i gang 2, mens 3G er hybel G på gang 3. Nedenfor vises de 7 hyblene på gang 1 og fellesarealet (de andre 4 gangene er identiske):



Programmet skal være menystyrt (bruk switch!), dvs. det skal skrive ut på skjerm en meny over mulig kommandoer og be brukeren om å gi en kommando. Programmet skal gi en feilmelding hvis brukeren taster inn feil. Menyen skal gå i en løkke helt til brukeren taster inn tallet 0 for å avslutte.

Vi skal ha følgende menyvalg i systemet:

0. Avslutt
1. Skriv liste over ledige hybler
2. Registrer ny leietager
3. Registrer frivillig flytting av leietager
4. Månedskjøring av husleie med strøm
5. Registrer betaling fra leietager
6. Sjekk om noen leietagere skal kastes ut
7. Skriv totalrapport

Programmet skal ved oppstart lese inn status for alle hyblene fra filen ”HaiHus.data”. Status pr. i dag ligger på fila Haihus.data på kurssidene, så begynn med å kopiere over denne fila til ditt område. Når brukeren velger Avslutt, skal programmet skrive alle disse opplysningene tilbake til samme fil slik de da har blitt endret. Dette gjør at Ærlige Johan ikke vil tape data hvis strømmen går eller maskinen slås av. Fila ”HaiHus.data” er på et bestemt format. Den har først 28 linjer, en for hver hybel, der hver linje er på følgende form:

int gang; char bokstav; String studentnavn; int saldo;

Et eksempel på en linje kan være:

2; C; Albert Aalesund; 2339;

For tomme hybler settes studentnavn lik ”TOM HYBEL” og saldo er mindre eller lik 0 (hvordan den kan bli negativ er beskrevet under). Etter dette skal det stå en linje med format

int totaltAntallMåneder; int antallHybelmånederMedTommeHybler; int totalFortjeneste

Det nest siste tallet teller totalt antall hybelmåneder med ledige hybler – har programmet gått i 2 måneder med 32 ledige hybler i den første og 28 i den neste, viser dette tallet 60. Det siste tallet skal akkumulere totalfortjeneste på all utleievirksomhet. Inn i dette beløpet skal regnes 2000 kr fortjeneste pr. utleid hybel pr. måned, mellomlegget mellom hva studentene betaler for strømmen og hva han selv betaler videre til strømleverandør, og evt. torpedogebyr og sluttgebyr (se under). Siden Ærlige Johan regner med å alltid få inn det noen skylder ham, oppdaterer vi totalfortjenesten hver gang vi oppdaterer skyldig beløp og ikke når vi registrerer betalt beløp.

Ærlige Johan har beregnet at avskrivninger og faste kostnader kommer på 1000 uqbarske kroner per hybel per måned. Han krever 3000 kroner per hybel per måned, ***pluss*** strøm for forrige måned. Videre får Ærlige Johan en månedlig spesifisert strømregning fra elektrisitetsverket Lysmegopp A/S. Regningen kommer på fil med følgende format:

husnr:gangnr:bokstav for arealet:antall kilowatt-timer forbruk

Her er **husnr** alltid 42, **gangnr** er et av heltallene 1-4, **bokstav for arealet** er en av bokstavene A-H, og **antall kilowatt-timer forbruk** er et ikke-negativt heltall. Et eksempel på en linje i filen kunne være

42:1:A:365

som ville svare til at fellesarealet i gang nr 1 sist måned hadde et strømforbruk på 365 kWh (kWh = kilowatt-timer). Lysmegopp A/S krever 80 øre per kilowatt-time, mens Ærlige Johan krever 2 kroner av studentene per kilowatt-time (noe (altså 1,20 kr per kilowatt-time) må han jo ha igjen for strevet, og dessuten må Ærlige Johan selv dekke kostnadene til strøm i fellesarealene). Strømregningen for hver hybel legges direkte til husleien for hybelen. Dette skal gjøres automatisk (dvs. via et menyvalg). Programmet skal ikke ta vare på tidligere måneders strømregninger.

Når en hybel blir ledig og ikke blir leid ut den påfølgende måned, legges strømregningen likevel til saldoen for hybelen. På denne måten kan saldoen bli negativ for en ledig hybel. Systemet er altså at man i første måned betaler utestående strøm fra forrige leieboer, mens man til gjengjeld slipper å betale siste måned i sin egen leieperiode.

For de ulike menyvalgene gjelder:

- **Skriv liste over ledige hybler**
  En enkel liste med på skjermen med hybelnavn (gangnummer+bokstav) på de ledige hyblene.

- **Registrer ny leietager**
  Her spør systemet om navn til studenten og hybelnavn samt at det registreres at studenten har betalt kr. 8000 i forskudd på husleie. Hvis det skrives inn navn på en opptatt hybel skal det gis feilmelding. Fortjenesten oppdateres videre med 3000 kr.

- **Registrer frivillig flytting av leietager**
  Her spørres det om studentens navn og hybel-navnet (eks. 3F, 1B,...) og det regnes ut hva studentene har tilgode på husleie fratrukket et ekspedisjonsgebyr på kr. 800, som legges til fortjenesten. Beløpet studenten til gode skrives til skjerm. Deretter registreres hybelen som ledig.

- **Månedskjøring av husleie med strøm**
  I begynnelsen av måneden kjøres dette menyvalget (straks Ærlige Johan har fått strømregningen). Her belastes først alle leietagernes objekter med 3000 kr hver. For hver utleid hybel legges 2000 kr til fortjenesten, mens det for hver ledige hybel trekkes 1000 fra fortjenesten. Så leses en fil (”Lysregning.data”) fra Lysmegopp med forrige månedens strømforbruk og denne legges til husleien for de enkelte hyblene som har leietagere, med 2 kr per kilowatt-time. Strøminntekter fratrukket strømregningen legges til fortjenesten.

- **Registrer betaling fra leietager**
  Denne metoden spør om hybelens navn(gangnummer og bokstav) samt hvilket beløp som betales. Metoden registrerer selvsagt i leietagerens objekt hvor mye som er betalt.

- **Sjekk om noen leietagere skal kastes ut**
  På slutten av måneden kjører Ærlige Johan denne rapporten. Den sjekker om noen av leietagerne har kommet i minus med husleien (de har da brukt opp forskuddet og ikke betalt nok i husleie). Disse kastes uten bønn. For hver av disse kalles en metode **`tilkallTorpedo (String hybelNavn, String student, int krav)`**, som skriver på en fil (”Torpedo.txt”) med oversikt over alle studentene som skal kastes ut denne måneden. Denne filen sender Ærlige Johan til en av sine gode venner som lever av slike oppdrag. Kravet til studenter som kastes ut er skyldig husleie + et torpedogebyr på kr. 1000. (Dette torpedogebyret går til Ærlige Johan selv, som kompensasjon for ekstra arbeid, dekning av tellerskritt og praktisk bryderi. Eventuell betaling av torpedoene er en sak mellom studentene og torpedoene, og behandles følgelig ikke i programmet) Slike hybler kan derfor med en gang markeres som ledige i systemet. Husk at det er bare en fil du skal lage for alle studentene som skal kastes ut den måneden, mens selve metoden skal kalles en gang for hver av disse studentene.
  Filen ”Torpedo.txt” skal ha følgende format: hybelNavn: StudentNavn: krav;

- **Skriv totalrapport**
  Her står du fritt til å skrive ut tall på skjermen som kan interessere Ærlige Johan, men som et minimum skal du skrive ut summen av all fortjeneste Ærlige Johan har hatt siden programmet startet, hvor mange måneder systemet har vært i virksomhet og hvor mange månedsleier han i gjennomsnitt hver måned har gått glipp av fordi hybler har stått tomme.

Du kan i dette systemet anta at alle studenter har entydige navn. Du skal også gå ut fra at hvis en student flytter inn eller ut i løpet av en måned, så må hun betale Ærlige Johan husleie for hele måneden (det blir enklest slik for Ærlige Johan).

Når programmet avsluttes, skal det skrive til fil status for alle hybler og studenter. Formatet skal være som beskrevet ovenfor til en egen fil ”HaiHus.data”.

Hint:

1. I denne oppgaven er det ikke gitt hvor mange klasser du skal lage, men du skal i alle fall ha 4 klasser: Oblig3, HybelHus, Hybel og Student. Det kan argumenteres for at Student er overflødig siden vi bare vet to ting om hver student: navnet og hvor mye de har innbetalt til Ærlige Johan. Det er vel likevel fornuftig med en egen klasse for Studenter fordi hvis Ærlige Johan ønsker å utvide systemet, kan det tenkes han vil vite mer om hver student som innflyttingsmåned, mobiltelefonnummer, hjemmeadresse,…. Vi kan da bruke følgende programskjelett, som du kan utvikle med mange flere metoder (antall linjer i ’mønsterbesvarelsen’ er ca. 450) :

```
class Student {
      int saldo;
      String navn;
```

```
      Metoder som behandler studentene og deres variable.
     …….
} // end Student

class Hybel {
      Student leietager;
      int utestående;
      Metoder som behandler hyblene og deres variable.
      ……
} // end Hybel

class HybelHus {
     <økonomi-data her>
      Hybel [][] hyblene = new Hybel[4][8];

      HybelHus(String filnavn) {
         <konstruktør for klassen HybelHus,
          les "HaiHus.data" og oppprett
          Hybel og Student-objektene>
       }
      …….
      void kommandoLøkke() {
            ……
      }
} // end HybelHus

class Oblig3 {
  public static void main (String[] args) {

       HybelHus utsyn = new HybelHus("HaiHus.data");
       utsyn.kommandoLøkke();

   } // end main
} // end class Oblig3
```

2. I eksemplet på strukturen i programmet ovenfor er klassen HybelHus utstyrt med en metode (uten **`void, static`** eller noe annet foran) som heter det samme som klassen. Det er en konstruktor (les avsnitt 8.7 og 8.11 i læreboken). En konstruktør er en metode som kalles automatisk når man sier **`new,`** og er en måte å overbringe parametere til objektene vi lager. (Konstruktører er meget nyttige, men ikke nødvendige. Som alternativ kunne vi først ha laget objektet ned **`new`** og så brukt metoder i objektet, eventuelt '.'-operatoren til å sette verdier inn i objektet)

3. I skjelettet av løsningen ovenfor er det en Student-peker **`leietager`** i Hybel-klassen. For å teste om den peker på et objekt, kan man teste:

   ```
   if (leietager == null)
   ```

   `null` er et Java-ord som er pekerverdien 'intet objekt'. Dvs. if-testen slår til hvis **`leietager`** ikke peker på et objekt. **`null`** kan også brukes i tilordningssetninger for eksempel hvis **`leietager`** peker på et objekt, kan vi få fjernet det med:

   ```
   leietager = null;
   ```

4. Vi har behov for å plassere alle Hybel-objektene i en array. I program-eksempelet er det nyttet en todimensjonal array, hvor første indeksen går på gangnummer og andre på hybel innen i en gang. Nå er problemet at hybelen har et bokstavnavn (A,B,…,H)og ikke et tall. Dette kan løses på følgende måte:

```
System.out.print("Gi gang:");
int gang = tast.inInt("\n");
System.out.print("Gi hybelbokstav:");
tast.skipWhite();
bokstav = tast.inChar();
i = (int) (bokstav - 'A');
```

Da vil: **`hyblene[gang-1][i]`** finne riktig peker i arrayen. Og motsvarende, hvis vi vet nummeret 'i' til en hybel i gangen , og vil ha bokstaven , gjør vi det slik:

```
char b = (char) ('A' + i);
```

Begge disse omgjøringene fra char til heltall og motsatt, hviler på at bokstavene A,B,… ligger etter hverandre i kodingen av tegnsettet.

5. Når du skal skrive på en fil flere ganger, som ved hver kall på tilkallTorpedo, og ønsker å legge noe til filen (ikke overskrive det som står der allerede), må du åpne fila med 'append' – dvs. at når du sier new, må du ha med en parameter nr. 2 som er 'true'. Det åpnes da en ny fil hvis filnavnet er nytt, men hvis ikke, åpnes den gamle fila på slutten av det som hittil er skrevet – for eksempel:

```
void tilkallTorpedo (String hybelNavn, String student, int krav){
            torpedo = new Out("Torpedo.txt",true);
            < .. skriv neste utkastelse av en student her …>
            torpedo.close();
}// end tilkallTorpedo
```

6. Når man bruker en peker, må man jo først teste om den peker på et objekt før man evt. kan få adgang til noe i objektet. Her er det nyttig å bruke **`&&`** testen (betinget-og) som er slik at den bare tester det som står til høyre for **`&&`** hvis det som står til venstre er sant – eks.:

```
Student s = hyblene[i][j].leietager ;
if (s != null && s.navn.equals("Ola")) {
  // her kommer vi bare hvis s peker på et studentobjekt
  // og navnet i det studentobjektet er lik "Ola"
    ...........
}
```

Vi tester altså først om pekeren er forskjellig fra null, og hvis det er sant, så bruker vi en objekt-variabel i det objektet **`s`** peker på. Eksempelet forutsetter at alle elementene i **`hyblene`** -arrayen peker på et Hybel-objekt og at alle **`navn`** i Studentobjekter peker på et String-objekt (hvis det er usikkert, må vi tilsvarende teste for det).

7. Registrering av hvor mange ledige hybler det er, kan registreres i metoden: månedskjøringHusleie siden den kjøres bare en gang per måned. Samtidig som du her også beregner månedens fortjeneste som legges sammen med Ærlige Johans totale fortjeneste.

8. For å lese inn lysregningen kan du benytte følgende kode for å dele opp linjene:

```
husnr      = lysregning.inInt(":");
gangnr     = lysregning.inInt(":");
bokstav    = lysregning.inChar(":");
kwt        = lysregning.inInt(": \n");
lysregning.skipWhite();
```

Innleveringskrav:

Oppgaven skal utføres og leveres individuelt. Det ferdige programmet med filene

- ✓ ”Oblig3.java” (selve programmet)
- ✓ Logg-fil for kjøring
- ✓ ”HaiHus.data” etter kjøring
- ✓ ”Torpedo.txt” slik den kan se ut når minst en student skal kastes ut

sendes gruppelæreren din på e-post innen leveringsfristen. I tillegg skal det legges ved ett eksempel på kjøring av systemet.

Hvis du har spørsmål eller spesielle behov vedrørende leveringsmåte eller lignende, så snakk med gruppelæreren din. Merk: søknad om utsettelse må leveres i god tid før fristens utløp.

Alle som har levert besvarelse kan i etterkant bli innkalt til hjelpelærer eller forelesere for å forklare sin løsning, og det vil da være en forutsetning at man kan gi en tilfredsstillende redegjørelse for sin egen besvarelse for å få godkjent oppgaven.
Denne innkallingen sendes per e-post til deres respektive UiOmailadresser. Dere MÅ følgelig sjekke disse!!

Du plikter å ha lest og forstått følgende krav til innleverte oppgaver ved institutt for informatikk: http://www.ifi.uio.no/studinf/skjemaer/erklaring.pdf

Oppgaven skal leveres elektronisk.

Første linje i programfilen (.java-filen) skal se slik ut:

// *dittBrukernavn* g-*gruppeNummer* o-3 k-inf1000/h05

hvor *dittBrukernavn* skal erstattes av ditt eget brukernavn (det navnet du oppgir når du logger deg inn på UiO's anlegg), og *gruppeNummer* skal erstattes av nummeret på den øvingsgruppa du er tatt opp på (hvis du har byttet øvingsgruppe, skal du altså oppgi gruppenummeret til den opprinnelige øvingsgruppa). Eksempel: hvis ditt brukernavn er olemann og du skal levere oblig3 på gruppe 17, så skal første linje i programfilen se slik ut:

// olemann g-17 o-3 k-inf1000/h05

Send programfilen (.java-filen) og de to datafilene (se nedenfor) med epost til gruppelæreren din (inf1000-*gruppeNummer*@ifi.uio.no). Resultatet av testkjøringen skal vise utskriften på skjermen

når du starter opp programmet ditt og tester ut de forskjellige delene som du er spurt om å lage i oppgaven.

1. Kjører du Unix, kan du bruke et program som heter *photo* for å lagre på fil alt som skrives ut på skjermen under en testkjøring. Du gir da følgende kommando i xterm-vinduet rett før du starter java-programmet ditt:
   photo *testOblig3.txt*
   Etter at du har startet programmet ditt, testet det ut og fått det til å avslutte, gir du kommandoen *exit* (eller trykker Control-D) for å avslutte photo-programmet. Nå ligger all utskriften fra testen på filen *testOblig3.txt* (du kan kalle filen noe annet hvis du ønsker).
2. Kjører du Windows, kan du starte kommandovinduet (Run/kjør: cmd.exe) ved å klikke på det øvre venstre symbolfelt på ramma, velger edit/rediger og markerer og tar kopi av dette. Deretter limer du dette inn i en editor (under Windows) og lagrer dette. Alternativt kan man under Windows trykke shift+AltGr+ Print Scrn, og da få et bilde av kommandovinduet som kan limes inn i et dokument (Word,...)

Hvis du har spørsmål vedrørende leveringsmåte eller annet, så kontakt gruppelæreren din i god tid før innleveringsfristen. Det er ditt ansvar at oppgaven kommer frem til øvingslæreren på riktig måte innen leveringsfristen.

N.B. Sender du inn innleveringen din fra en e-postkonto fra yahoo eller hotmail, er det en stor sjanse at spamfilteret på Ifi tar besvarelsen og den vil aldri nå hjelpelærerer. Send derfor helst besvarelsen din fra den e-postkontoen du har fått her på Universitetet.