UiO : Universitetet i Oslo

Oblig 2 (INF1000 - Høst 2010)

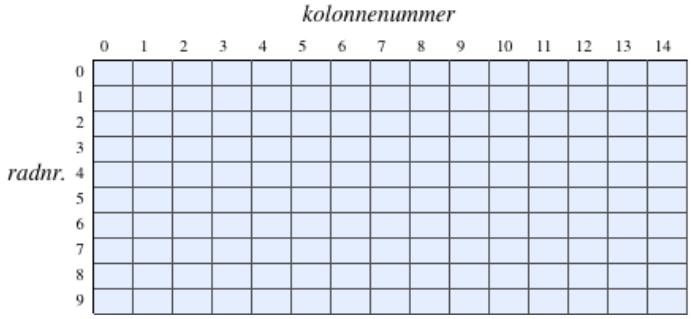
Mål: *Formålet med denne oppgaven er å gi trening i bruk av forgreninger, løkker, arrayer, metoder, programmering av kommunikasjon med bruker via terminal, og som ekstraoppgaver filbehanding og utregning med tekster.*

Leveringsfrist

Fredag 1. oktober kl 16.00, leveres via dette Innleveringssystemet. *Viktig:* Les slutten av oppgaven for detaljerte leveringskrav.

Oppgave

Utenfor kysten av Utopia er det et område med store oljeforekomster under havbunnen, og myndighetene har bestemt seg for å selge rettighetene til å utvinne olje til oljeselskaper. Det aktuelle havområdet er rektangulært og er delt opp i et rutenett med 10 x 15 ruter (se figuren under), hvor radene er nummerert fra 0 til 9 og kolonnene er nummerert fra 0 til 14. Hver rute kalles et utvinningsfelt (eller bare felt). Hvert felt har et entydig navn på formen *radnr-kolnr* hvor *radnr* er et heltall mellom 0 og 9, og *kolnr* er et heltall mellom 0 og 14. For eksempel angir 0-0, 0-1, 0-2, ... 0-14 feltene i øverste rad (fra venstre mot høyre) i figuren under.



Feltene (eller mer presist utvinningsrettighetene til dem) legges ut for salg enkeltvis. Det er ingen grenser for hvor mange felt et oljeselskap kan eie, men et felt kan ikke ha mer enn ett oljeselskap som eier (dvs. hvert felt er enten ikke solgt eller det eies av ett oljeselskap).

Myndighetene i Utopia trenger et datasystem for å administrere oljefeltene. Oppgaven din er å programmere dette systemet, som skal kunne holde rede på hvilke felt som er solgt, til hvilke eiere, og hvor mye olje (målt i antall fat) som er utvunnet i hvert felt. Bruker av systemet tenker vi er en funksjonær som jobber for myndighetene i Utopia. Når et oljeselskap ønsker å kjøpe et gitt felt, ringer de funksjonæren som så bruker ditt program for å registrere kjøpet av feltet. Funksjonæren skal også kunne få ut fra programmet en oversikt over feltene med eiere og en del annen nyttig informasjon, som forklart nedenfor.

Programmet skal være kommandostyrt: det skal kunne ta imot en kommando fra brukeren, utføre kommandoen, ta imot en ny kommando, osv., helt til brukeren ønsker å avslutte. Mer konkret skal programmet oppføre seg slik:

- Det skriver ut på skjermen en meny med kommandoene brukeren kan gi.
- Deretter ber programmet om og leser inn en kommando fra brukeren.
- Programmet utfører den valgte kommandoen.

Programmet skal gjenta de tre trinnene ovenfor helt til brukeren gir kommando om å avslutte. Programmet blir mye ryddigere hvis du lager en egen metode for hver kommando, og derfor er det et krav at din løsning har minst en metode for hver kommando (unntatt Avslutt-kommandoen).

Kommandoene

Brukeren skal kunne gi seks av disse kommandoene:

```
1. Kjøp et felt
2. Liste over solgte felt
3. Lag oversiktskart med statistikk
4. Oppdater oljeutvinning
5. Finn raden med høyest oljeutvinning
6. (Ekstraoppgave): Skriv til fil
7. (Ekstraoppgave): Les fra fil
8. (Ekstraoppgave): Selskap med flest felt  (vanskelig)
0. Avslutt
```

Kommando 1-5 og 0 er krav, mens 6-8 er valgfrie ekstraoppgaver. Det anbefales likevel for alle å løse nr. 6 og 7 fordi det vil lette arbeidet med oblig 3, som er den største obligatoriske oppgaven i kurset. Ekstraoppgave 8 er ment for spesielt interesserte.

1. **Kjøp et felt:**
   Denne kommandoen vil funksjonæren gi hvis et oljeselskap ringer og sier at de ønsker å kjøpe et felt. Programmet skal da spørre om navnet på feltet som ønskes kjøpt og navnet på oljeselskapet som ønsker å kjøpe feltet. Deretter skal programmet sjekke om feltet er ledig (dvs. ikke har noen eier):

   (a) Hvis feltet er ledig skal programmet registrere at det inntastede oljeselskapet nå eier det aktuelle feltet. Programmet skal også skrive ut på

skjermen en beskjed om at kjøpet gikk i orden, f.eks.: `Felt 3-12 er nå kjøpt av Shell`

(b) Hvis feltet ikke var ledig skal programmet først sjekke om selskapet som eier feltet fra før er det samme selskapet som bruker tastet inn. I så fall skal programmet skrive en melding om det, f.eks. `Oljeselskapet Shell eier allerede felt 3-12!`

(c) Hvis verken (a) eller (b) var tilfelle, så betyr det at feltet som bruker tastet inn har en annen eier. Da skal programmet skrive ut nåværende eier av feltet og spørre bruker om kjøpet virkelig skal gjennomføres, dvs. om feltet skal overtas av det nye selskapet. Hvis bruker svarer `ja` skal kjøpet registreres som i del (a). Hvis bruker svarer noe annet enn `ja` skal det gis en melding til bruker om at ingen endring ble registrert.

2. **Liste over solgte felt:**
Programmet skal da gå gjennom alle feltene, og for de feltene som har eier skal programmet skrive ut på skjermen feltnavnet, navnet på oljeselskapet som eier det, og antall fat olje som er utvunnet i feltet. Eksempel på utskrift:

```
Felt 3-12 eies av Shell. Total utvinning i feltet: 0 fat
Felt 5-6 eies av Esso. Total utvinning i feltet: 80 fat
Felt 9-0 eies av Shell. Total utvinning i feltet: 50 fat
```

3. **Lag oversiktskart med statistikk:**
Programmet skal da skrive ut på skjermen et kart hvor feltene er markert med en "`.`" hvis det er ledig og "`x`" hvis det er solgt (dvs. er kjøpt av et oljeselskap). Etter at kartet er skrevet ut skal programmet også skrive ut 3 verdier: (a) Antall felt som er solgt; (b) Summen av alle utvinningene i alle felt; og (c) Gjennomsnittlig utvinning, regnet ut som antall fat per solgte felt. Gjennomsnittet skal avrundes til én desimal. Her er et eksempel på hvordan kartet kan se ut:

```
   0  1  2  3  4  5  6  7  8  9 10 11 12 13 14
0  x  .  .  .  .  .  .  .  .  .  .  .  .  .  .
1  .  .  .  x  .  .  .  .  .  .  .  .  .  .  .
2  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
3  .  .  .  .  .  .  .  .  .  .  .  .  x  .  .
4  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
5  .  .  .  .  .  .  x  .  .  .  .  .  .  .  .
6  .  .  .  .  x  .  .  .  .  .  .  .  .  .  .
7  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
8  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
9  .  .  .  .  .  .  .  .  .  .  .  .  .  .  x

Antall solgte felt: 6   Total utvinning: 560 fat
Gjennomsnittlig utvinning: 93.3 fat pr solgte felt
```

4. **Oppdater oljeutvinning:**
Hver sjette måned må oljeselskapene gi beskjed til funksjonæren om hvor mye olje de har utvunnet på feltene sine i løpet av de seks siste månedene. Da bruker funksjonæren denne kommandoen for å registrere informasjonen. Programmet skal gå gjennom feltene som har eier, og spørre for hver av disse hvor mye olje som er blitt utvunnet der de siste seks månedene, f.eks. slik:

```
Antall fat utvunnet i felt 3-12 siste 6 mnd. (tidligere total 0 fat): 0
Antall fat utvunnet i felt 5-6 siste 6 mnd. (tidligere total 80 fat): 20
Antall fat utvunnet i felt 9-0 siste 6 mnd. (tidligere total 50 fat): 500
```

(Tallene som står understreket på slutten av linjene er det som bruker taster inn, alt annet skal programmet ditt skrive ut). Programmet skal registrere opplysningene som bruker taster inn ved å *summere* de nye antall oljefat inn i arrayen som holder dataene om utvinning, f.eks. hvis utvinningen i felt 5-6 var 80 oljefat fra før, skal verdien nå bli 100. Sørg for at løsningen din i deloppgave 2 ("Liste over solgte felt") benytter seg av disse opplysningene og viser den oppdaterte utvinningen neste gang bruker utfører kommando 2.

5. **Finn raden med høyest oljeutvinning:**
Programmet skal da finne ut og gi en melding til bruker om hvilken rad som har produsert mest olje, dvs. i hvilken rad er summen av utvinningene (for alle felt i raden) høyest. Hvis det er flere rader som har det samme maksimale antall utvunnede oljefat er det nok at programmet skriver nummeret på én av disse radene. Meldingen til bruker skal vise både radnummeret der høyest sum utvinning ble funnet, og sum utvinning for raden, f.eks. slik:

```
Rad med høyest oljeutvinning: rad 9 (500 fat)
```

6. **(Ekstraoppgave): Skriv til fil:**
Et lite problem med programmet beskrevet ovenfor er at dataene man har registrert mens programmet kjører går tapt hvis det avsluttes. Hvis du ønsker å fikse dette kan du utvide programmet slik at det tar vare på dataene ved å lagre de i en datafil kalt `olje.txt` når brukeren utfører kommando "6". Datafilen kan for eksempel se slik ut:

```
3
0;0;100;Statoil
1;3;200;BP
3;12;50;Shell
```

I dette eksemplet betyr tallet 3 på første linje at det er 3 felt som er solgt, og at det dermed også er 3 linjer til i datafilen, med informasjon om disse solgte felt. Neste linje betyr at Statoil eier felt 0-0 og har utvunnet 100 fat olje der; neste linje angir at BP eier felt 1-3 og har utvunnet 200 fat olje på det feltet, osv. Husk å lukke fila med `fil.close();`

7. **(Ekstraoppgave): Les fra fil:**
   Denne oppgaven er også valgfri, og går ut på å lese datafilen som blir opprettet i foregående oppgave ("Skriv til fil"). Vi tenker oss at man utfører denne kommandoen når man starter programmet, for å hente inn dataene som ble lagret i filen `olje.txt` forrige gang man kjørte programmet. Derfor er det greit om du bare overskriver dataene i arrayene med dataene som leses fra fil. (Men hvis du vil gjøre det annerledes er det også greit, ekstraoppgavene står man fritt til å løse som man vil.)

8. **(Ekstraoppgave): Selskap med flest felt:** *(Vanskelig!)*
   Den som vil ha en større utfordring kan bryne seg på denne ekstraoppgaven: Finn selskapet som har kjøpt flest felt.

0. **Avslutt.**
   Programmet skal da avslutte.

Kjøreeksempel

Nedenfor ser du et eksempel på en kjøring av programmet. Bruker-input er markert med understreking. Dette kjøreeksemplet er bare ment å illustrere hvordan kommunikasjonen med bruker kan foregå; dersom du ønsker å presentere menyen eller andre ting annerledes, så kan du gjøre det.

```
> java Oblig2
*** Utopias Oljefeltadministrasjon ***
Meny:
 1. Kjøp et felt
 2. Liste over solgte felt
 3. Lag oversiktskart med statistikk
 4. Oppdater oljeutvinning
 5. Finn raden med høyest oljeutvinning
 0. Avslutt
Velg kommando: 1

** Kjøp et felt **
Felt som ønskes kjøpt: 3-6
Oljeselskapets navn: Statoil
Felt 3-6 er nå kjøpt av Statoil

Meny:
 1. Kjøp et felt
 2. Liste over solgte felt
 3. Lag oversiktskart med statistikk
 4. Oppdater oljeutvinning
 5. Finn raden med høyest oljeutvinning
 0. Avslutt
Velg kommando: 2

** Liste over solgte felt **
Felt 3-6 eies av Statoil. Total utvinning i feltet: 0 fat

...osv...
```

Hint

Disse hint er bare for de som ønsker litt ekstra-hjelp. Du trenger ikke lese dette avsnittet for å løse obligen, men alle må lese siste avsnitt om Leveringskrav. Det kan også bli lagt ut flere ekstra-hint senere.

a. **Generelle tips:** Et godt sted å starte kan være å få kommandoløkken til å fungere. Sørg for at programmet klarer å lese inn kommandoer fra tastatur og vise menyen. Deretter går du videre og programmerer de enkelte kommandoene, i vilkårlig rekkefølge. Når du skriver programmet anbefales det at du kompilerer og prøvekjører det ofte underveis, da unngår du at det samler seg mange feil som blir vanskeligere å rette samlet etterpå. Husk at det er et krav at du lager minst én metode for hver kommando (unntatt Avslutt).

   Hvis noe er uklart i oppgaveteksten kan du gjøre egne forutsetninger, men sørg for å beskrive disse ved hjelp av Java-kommentarer i

programmet. Det er lov å utvide funksjonaliteten eller gjøre programmet mer brukervennlig, men retter må fortsatt kunne taste inn kommando 1-5 som tall, og basis-funksjonaliteten til disse skal være som forklart ovenfor.

b. **Arrayene:** For hvert felt er det to opplysninger som skal tas vare på: (1) navnet på oljeselskapet som eier det; og (2) hvor mye olje som er utvunnet i feltet. Du kan bruke to to-dimensjonale arrayer til å lagre disse opplysningene:

```
String[][] eier = new String[10][15];
int[][] utvunnet = new int[10][15];
```

Da vil f.eks. `eier[1][3]` inneholde navnet på oljeselskapet som eier felt 1-3; og `utvunnet[1][3]` vil være antall fat olje som er utvunnet i det feltet. Det finnes også andre måter å løse oppgaven på, bl.a. kunne vi klart oss med bare én to-dimensjonal array med pekere til objekter av en klasse som vi selv lager, men denne teknikken gjennomgås senere i kurset og bør ikke anvendes i denne oppgaven.

c. **Kjøp et felt:** Når programmet skal lese inn navnet på feltet som skal kjøpes, må du trekke ut radnummeret og kolonnenummeret fra feltnavnet. Hvis f.eks. brukeren skriver 3-6 så må du altså "få tak i" tallene 3 og 6 for å kunne lagre navnet på oljeselskapet i eier-arrayen. Dette kan gjøres på flere måter; en av dem er å bruke setningene:

```
int radnr = tast.inInt("-\n\r ");
int kolnr = tast.inInt("-\n\r ");
```

Her er det en variant av `inInt()` som brukes. Det som står i anførselstegn er de tegnene som skal betraktes som skilletegn - altså de tegnene som `inInt()` skal se på som "blanke tegn" før og etter tallene. De spesielle skilletegnene linjeskift (`"\n"`) og vognretur (`"\r"`) er ikke nødvendige å angi som skilletegn, siden de alltid betraktes som skilletegn i easyIO, men kan godt være med som vist ovenfor. Mellomroms-tegnet kan tas med inne i anførselstegnene for å gjøre programmet mer brukervennlig (da vil programmet også godta at bruker taster inn `"3 - 6"` eller t.o.m. `"3 6"` som feltnavn).

I del (a) må programmet ha en måte å finne ut om et felt er solgt eller ikke. En enkel måte å finne ut om f.eks. felt 3-6 er solgt er å se om `eier[3][6]` er satt til et navn eller ikke. Rett etter at 2D-arrayen `eier` er deklarert er alle verdiene i arrayen lik den spesielle verdien `null`, og du kan derfor sjekke om 3-6 er ledig ved å teste om `eier[3][6] == null`.

I del (b) skal du sammenligne tekst-verdier med hverandre og finne ut om de er like. Vanligvis bruker man "==" i Java for å teste om to verdier er like, men dette fungerer på en litt annen måte med tekst-verdier. For å sjekke om to tekster er like bruker man i stedet metoden `.equals()`, slik: `"if (tekst1.equals(tekst2)) {..."`. Dette tester om *innholdet* i to tekst-pekerne er likt. Mer om dette kan du lese i avsnitt 6.4.1 på side 105 i læreboka. Ved sammenligning av en tekst mot den spesielle verdien `null` derimot, så skal man bruke "==" (eller `"!="`), se (a) ovenfor.

I del (c) trenger du et ja/nei-spørsmål. Det kan programmeres på mange måter, f.eks. vha. en `inWord()`-setning som leser inn `"ja"` eller `"nei"` fra bruker, eller vha. `inChar("\n ")` og betrakte `'j'` som ja-svaret, og alt annet som nei.

d. **Liste over solgte felt:** Denne listen er nyttig for å sjekke at de andre deloppgavene fungerer, og kan programmeres f.eks. rett etter at del (a) av "Kjøp et felt" er gjort. For å forenkle arbeidet kan du først programmere utskriften til å bare vise 0 som utvinning i alle feil, og så, når du har programmert deloppgave 4 ("Oppdater oljeutvinning") legger du til visning av riktig utvinning for feltene.

e. **Lag oversiktskart med statistikk:** Denne oppgaven kan løses på mange måter. Det enkleste er kanskje å programmere kartet alene først, og så når du har gjort det, kan du legge inn i samme kode som tegner kartet en teller-variabel som holder rede på hvor mange "x"-er det var, og til slutt legger du til summering av oljeutvinningene.

f. **Oppdater oljeutvinning:** Husk at du i denne oppgaven skal *addere* den innleste verdien til den verdien som allerede er registrert i systemet, ikke bare erstatte verdien med den nye som bruker tastet inn. Verdien som lagres i arrayen skal altså være det totale antall fat olje som er utvunnet i feltet.

g. **Finn raden med høyest oljeutvinning:** Finn helst din egen måte å løse denne deloppgaven på! Hvis du står helt fast så er to mulige måter å løse deloppgaven på: (1) Å først lage en éndimensjonal tilleggs-array der du lagrer summene for radene. (2) En annen løsning er å løpe gjennom feltene vha. to nestede løkker og hele tiden under gjennomløpene holde rede på sum utvinning i nåværende rad, maksimal radsum funnet så langt, og radnummer der sistnevnte ble funnet.

h. **(Ekstraoppgave): Les fra fil:** Når du begynner å programmere lesing av datafilen bør du legge inn testutskrifter som skriver ut på skjermen det som faktisk blir lest fra filen, slik at du kan kontrollere at datafilen blir lest riktig. Dette kan du gjøre f.eks. ved å bruke følgende to programsetninger i stedet for bare å lese dataene fra fil med `int x = fil.inInt(" ;");`

```
int x = fil.inInt(" ;");  skjerm.outln("x=" + x);
```

i. **Programstruktur:** Nedenfor er det en programskisse du kan bruke som utgangspunkt. Du behøver ikke å følge det, men dette er i alle fall et forslag til en ryddig start på programmet. Hvis du følger denne skissen så trenger du ikke endre noe i den første klassen (`class Oblig2`), det er nok at du gjør alle endringene dine i klassen *Olje*. Det vil også bli lagt ut et forslag til prekode i bloggen som ikke benytter easyIO.

```
import easyIO.*;
// Her kan du skrive evt. kommentarer om besvarelsen din.

class Oblig2 {
    public static void main(String[] args) {
        System.out.println("*** Utopias Oljefeltadministrasjon ***");
        Olje ol = new Olje();
        ol.kommandolokke(); // Kjører metoden kommandolokke() i klassen Olje.
        System.out.println("-- Programmet avslutter --");
    }
}

class Olje {
    // Klargjør for innlesing/utskrift med easyIO, gjelder for hele klassen:
    In tast = new In();
    Out skjerm = new Out();

    // Her kan du deklarere arrayene eier[][] og utvunnet[][]:
    // ...

    // Metoden "kommandolokke()": Tar imot kommandoer fra bruker og utfører disse.
    void kommandolokke() {
        int kommando = -1; // Tilfeldig startverdi.

        while (kommando != 0) {

            // Legg her setninger som skriver ut menyen, f.eks. linjer med
            // println eller skjerm.outln(), eller kall på en egen metode.
            System.out.println(" 1. Kjøp et felt");

            // Her kan du skrive ut en ledetekst, f.eks. "Velg kommando: "
            // ...
            // Og så leser du input som bruker taster inn:
            kommando = 0;  // (Erstatt 0 med kode som leser fra tastatur.)

            switch (kommando) {
                case 1: kjopEtFelt(); break;
                case 2: listeOverSolgteFelt(); break;
                case 3: lagOversiktskartMedStatistikk(); break;
                // ... fyll inn case-er for de to andre kommandoene her.

                default: break;
            }
        }
    }

    // Metoder for hver kommando:

    void kjopEtFelt() {
        // Programmér kjøp av felt her:

        // - Be bruker taste inn rad-kol., og oljeselskap.

        // - Sjekk om arrayen eier[][] allerede har et oljeselskap i den
        //   angitte rad-kol.:  (a) Hvis ikke => registrér kjøpet, osv...

    }

    void listeOverSolgteFelt() {
        // Programmér kjøp av felt her:

    }

    void lagOversiktskartMedStatistikk() {

    }

    // ...osv... (minst 2 metoder til)
}
```

Leveringskrav

Du skal bare levere én fil, som må ha filendelsen "`.java`" (altså ikke `.tgz` eller noe annet), og filnavnet bør være `Oblig2.java`. Java-nøkkelordet "`package`" skal *ikke* brukes i programmet. Hvis du bruker norske bokstaver (æøå) i filen bør tegnsettet ("*encoding*") til filen være utf-8 eller iso-8859-1. Tips om dette kan du finne i bloggen.

Hvis du har kommentarer til gruppelærer angående besvarelsen din, f.eks. de som nevnes i følgende avsnitt, så skriver du disse i en Java-kommentar øverst i programmet før du leverer. Det er lov å levere flere utgaver av besvarelsen din, det er bare den siste du leverer innen fristen som blir rettet. Husk å velge oblignummer 2 i innleveringssystemet i alle innleveringer du gjør av oblig 2.

Du kan *diskutere* med andre studenter hvordan dere skal løse oppgavene, men det er ikke lov å *kopiere* noe Java-kode fra dem, selv om du endrer på koden etterpå, og det er heller ikke lov å hente programbiter fra andre besvarelser, for eksempel fra Internet. Hver student skal skrive sitt eget program. Dette er nærmere forklart i følgende krav til innleverte oppgaver ved Ifi, som du plikter å ha lest og forstått før du leverer din besvarelse:

> http://www.ifi.uio.no/studier/studentinfo.html#krav

Alle innleveringer vil bli kontrollert med Joly-algoritmen, så hvis du samarbeidet mye med andre eller hentet mye hjelp fra andre kilder enn lærebøker, websidene til kurset, eller gruppelærere, bør du nevne disse kildene i besvarelsen din og hvilke deler av programmet det gjelder, som forklart i lenken ovenfor.

Tilbakemelding på obligen din vil du få via e-post og Godkjentsystemet wwws.ifi.uio.no innen to uker etter leveringsfristen. Mer informasjon om tilbakemelding og prosedyrer rundt innlevering kan du finne i Reglement for obligatoriske oppgaver ved Ifi. Hvis du skal levere forbedret utgave av obligen etter fristen, så skal du fortsatt bruke samme innleveringssystem og velge oblignummer 2 der.

Hvis du har spørsmål, kommmentarer, eller finner feil i oppgaveteksten kan du skrive disse i kurs-bloggen.

Lykke til!