Løsningsforslag ukeoppg. 5: 21. - 27. sep (INF1000 - Høst 2011)

Filer; tekst; repetisjon av metoder *(kapittel 3 og 6 i "Rett på Java"* ***3. utg.****)*

> ***NB!*** *Legg merke til at disse er* ***løsningsforslag****. Løsningene dine trenger ikke å være like med disse forslag for å være riktige. Det er vanlig i programmering at samme oppgave kan løses på mange vidt forskjellige måter, og alle fremgangsmåter er ok i INF1000 så lenge de leder fram til riktig resultat og oppfyller kravene som står i oppgaveteksten.*

Mål
Få øvelse i teorien du trenger for å løse Oblig 2 (løkker, arrayer, tekster, og metoder); og i tillegg litt om innlesing og utskrift til fil.

🔑 Oppgave merket med nøkkelsymbol er plukket ut som spesielt representativ for de viktigste temaene fra ukens forelesning, og alle bør ha som minimumsmål å løse denne selvstendig.

Oppgaver

1. **Enkle løkker med tall:**
   (a) Nedtelling: Lag et program med en løkke som teller ned fra 10 til 0. Bruk en teller-variabel med startverdi 10 og redusér den med 1 i hver gjennomgang av løkka. Utskriften skal bli:

```
...10 ...9 ...8 ...7 ...6 ...5 ...4 ...3 ...2 ...1 ...0
```

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

        for (int teller = 10; teller >= 0; teller--) {
            System.out.print(" ..." + teller);
        }
        System.out.println();
   }
}
```

Java-nedtelling: *(kun for spesielt interesserte)* Hvis du vil legge inn 1-sekunds pause før hvert tall skrives ut kan du skrive "`Thread.sleep(1000); // Vent 1000 millisek.`" inne i for-løkka og endre "`(String[] args)`" til "`(String[] args) throws Exception`". Exceptions er *ikke pensum* men omtales i kap. 18. Bare kap. 1 - 9 (og litt av 12) er pensum.

(b) Dobling: Lag en løkke som starter med å skrive ut tallet 2, og deretter dobler tallet, skriver det ut, og gjentar prosessen helt til 10 tall er skrevet ut. Bruk en teller-variabel "`i`" som teller de 10 gangene, og en annen variabel for tallet som ganges med 2 i hver omgang av løkka og skrives ut. Utskriften skal bli:

```
2  4  8  16  32  64  128  256  512  1024
```

```
class PowersOf2 {
   public static void main(String[] args) {

        int potens2 = 1;
        for (int i = 0; i < 10; i++) {
            potens2 = potens2 * 2;
            System.out.print(potens2 + "  ");
        }
        System.out.println();
   }
}
```

(c) Tabell: Lag to nestede for-løkker som gir følgende utskrift. Det skal bare skrives ut ett tall av gangen, som skal være verdien til telleren i den ytre løkka (som går fra 1 til 3 og skal hete *rad*) ganget med telleren i den indre løkka.

```
1  2  3  4
2  4  6  8
3  6  9  12
```

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

        int produkt = 1;
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 4; j++) {
                produkt = i * j;
                System.out.print(produkt + "  ");
            }
            System.out.println();
```

```
        }
    }
}
```

2. **2D-array (to-dimensjonal array) med tall:**
(a) Utskrift: Lag to nestede for-løkker som setter inn følgende verdier i en 2D-array og skriver de ut. Arrayen skal deklareres slik: `int[][] tabell = new int[3][4];`. Se oppgave nr. 1 (c) over for tips om disse verdiene. Husk at indeksene i arrayen starter fra 0, ikke 1. Verdien i `tabell[2][3]` skal være 12.

```
 1  2  3  4
 2  4  6  8
 3  6  9  12
```

```
class Array2D {
   public static void main(String[] args) {

       int[][] tabell = new int[3][4];

       for (int i = 0; i < 3; i++) {
           for (int j = 0; j < 4; j++) {
               tabell[i][j] = (i + 1) * (j + 1);
               System.out.print(tabell[i][j] + "  ");
           }
           System.out.println();
       }
   }
}
```

(b) Sum av kolonne: Skriv programkode som beregner summen av verdiene i hver kolonne i ovennevnte array og skriver summene ut. *Dette ligner på en deloppgave i oblig 2!*

```
 6 12 18 24
```

```
       System.out.println("Sum av kolonner:");
       for (int kol = 0; kol < 4; kol++) {
           int sumkol = 0;
           for (int rad = 0; rad < 3; rad++) {
               sumkol += tabell[rad][kol];
           }
           System.out.print(sumkol + " ");
       }
       System.out.println();
```

(c) Søk etter tall: Utvid programmet slik at det ber bruker taste inn et tall og deretter ser om tallet finnes i arrayen. Hvis tallet finnes skal alle array-indeksene der det ligger skrives ut, hvis ikke skal det gis beskjed om at tallet ikke finnes.

```
 Hvilket tall søker du: 8
 Tallet finnes i tabell[1][3]
```

```
class Array2D {
   public static void main(String[] args) {

       int[][] tabell = new int[3][4];

       // (a)
       for (int i = 0; i < 3; i++) {
           for (int j = 0; j < 4; j++) {
               tabell[i][j] = (i + 1) * (j + 1);
               System.out.print(tabell[i][j] + "  ");
           }
           System.out.println();
       }

       // (b)
       System.out.println("\n (b) Sum av kolonner:");
       for (int kol = 0; kol < 4; kol++) {
           int sumkol = 0;
           for (int rad = 0; rad < 3; rad++) {
               sumkol += tabell[rad][kol];
           }
           System.out.print(sumkol + " ");
       }
       System.out.println();

       // (c)
       System.out.print("\n(c) Hvilket tall søker du: ");
       int tall = new easyIO.In().inInt();
       for (int rad = 0; rad < 3; rad++) {
           for (int kol = 0; kol < 4; kol++) {
               if (tall == tabell[rad][kol]) {
                   System.out.println("Tallet finnes i tabell["
                                      + rad + "][" + kol + "]");
               }
```

```
            }
        }
    }
}

KJØREEKSEMPEL:
> java Array2D
1  2  3  4
2  4  6  8
3  6  9  12

(b) Sum av kolonner:
6 12 18 24

(c) Hvilket tall søker du: 4
Tallet finnes i tabell[0][3]
Tallet finnes i tabell[1][1]
```

3. **Tekster: Antall tegn, store bokstaver, og like tekster** *kap. 6, oppg. 1 (side 117)*
   Lag et program som:
   (a) ber brukeren skrive inn en tekst og beregner antall tegn i teksten.
   (b) ber brukeren om å skrive inn en tekst og lager en ny tekst av den gamle med bare store bokstaver, og skriver ut den nye teksten. (*Tips:* Se spørsmål "o" i oppgave 5 nedenfor.)
   (c) ber brukeren skrive inn to ord og tester om disse er like.

```
import easyIO.*;

class Oppg1Kap6 {
    public static void main (String[] args) {
        In tast = new In();
        Out skjerm = new Out();

        // (a)
        skjerm.out("Skriv en tekst: ");
        String tekst = tast.inLine();
        skjerm.outln("Antall tegn: " + tekst.length());

        // (b)
        skjerm.out("Skriv en tekst: ");
        tekst = tast.inLine();
        skjerm.outln("Teksten i store bokstaver: " + tekst.toUpperCase());

        // (c)
        skjerm.out("Skriv to ord adskilt med mellomrom: ");
        String ord1 = tast.inWord();
        String ord2 = tast.inWord();
        if (ord1.equals(ord2)) {
            skjerm.outln("ordene er like");
        } else {
            skjerm.outln("ordene er ikke like");
        }
    }
}

KJØREEKSEMPEL:
> java Oppg1Kap6
Skriv en tekst: abcd
Antall tegn: 4
Skriv en tekst: abcd
Teksten i store bokstaver: ABCD
Skriv to ord adskilt med mellomrom: ab ab
ordene er like
```

4. **String og charAt: Ord baklengs** *kap. 6, oppg. 3 (a)-(b), (side 117)*
   (a) Lag et program som skriver ut teksten «Agnes i senga» baklengs. *Hint:* Bruk en for-løkke som teller nedover.
   (b) Modifiser programmet over slik at det først gjør om teksten til kun å inneholde små bokstaver.

```
import easyIO.*;

class Baklengs {
    public static void main (String[] args) {
        Out skjerm = new Out();

        String tekst = "Agnes i senga";

        skjerm.out(tekst + " baklengs: ");
        int antallTegn = tekst.length();
        for (int i = antallTegn - 1; i >= 0; i--) {
            skjerm.out(tekst.charAt(i));
        }
        skjerm.outln();
    }
}

KJØREEKSEMPEL:
> java Baklengs
Agnes i senga baklengs: agnes i sengA
```

5. **Tekster: Hva blir skrevet ut?**

```
import easyIO.*;
class Tekster {
    public static void main(String[] args) {
        Out skjerm = new Out();

        String s1 = "hei";
        String s2 = "Java";
        String[] navn = { "Rune", "Martin", "", "Guro" };

/* a */ skjerm.outln(navn.length + s1.length());                          // 7

/* b */ skjerm.outln(3.1415 + "" + 'x');                                  // 3.1415x

/* c */ skjerm.outln("" + false);                                         // false

/* d */ skjerm.outln("" + ! "abc".equals("abc"));                         // false

/* e */ skjerm.outln("heia" == (s1 + "a"));                               // false

/* f */ skjerm.outln("heia".equals(s1 + s2.charAt(1)));                   // true

/* g */ skjerm.outln(s1.equals("h" + navn[0].charAt(3) + 'i'));           // true

/* h */ skjerm.outln(navn[1].substring(1));                               // artin

/* i */ skjerm.outln(navn[1].substring(1, 4));                            // art

/* j */ skjerm.outln(s2.replace('a', 'i'));                               // Jivi

/* k */ skjerm.outln(navn[1].indexOf("tin"));                             // 3

/* l */ skjerm.outln(navn[2].indexOf("tin"));                             // -1

/* m */ skjerm.outln("A".compareTo("A"));                                 // 0

/* n */ if (s1.compareTo("zz") < 0) skjerm.outln("s1 alfabetisk foran");  // s1 alfabetisk foran

/* o */ skjerm.outln(navn[3].toUpperCase());                              // GURO

/* p */ if ("hei på deg".startsWith(s1)) { skjerm.outln("ja"); }          // ja

/* q */ int x = Integer.parseInt("123"); skjerm.outln(x + 1);             // 124

    }
}
```

6. **Innlesing og utskrift til fil:** *(eksempel side 52 i læreboka)*
(a) Ta utgangspunkt i følgende program, fra side 52 i læreboka, som skriver fire typer verdier til skjerm (tegn, linje, heltall, og desimaltall), og endre programmet slik at utskriften går til en fil i stedet. Kall utfilen "`minfil.txt`".

```
import easyIO.*;

class Utskrift {
    public static void main(String[] args) {
        Out skjerm = new Out();
        skjerm.outln('A');
        skjerm.outln("Canis familiaris betyr hund");
        skjerm.outln(15);
        skjerm.outln(3.1415, 2); // Bruk to desimaler
    }
}
```

Hint:

- **EasyIO:** Utskrift til fil programmeres nesten på samme måte som utskrift til skjerm, bare at man angir filnavn i klargjøringen av "new Out", bruker et annet navn på forbindelsen (f.eks. "utfil" i stedet for "skjerm"), og lukker filen til slutt. Mer detaljert: Klargjør for utskrift til fil med f.eks. "Out utfil = new Out("minfil.txt");" Deretter bruker du utfil.out/outln for utskrift til fil på samme måte som du ville brukt skjerm.out/outln for utskrift til skjerm. Til slutt lukker du filen med utfil.close();. Hvis man glemmer close blir ingenting lagret i filen.
- **PrintWriter/printf:** ***NB!*** Dette er bare ment **for spesielt interesserte** som vil prøve uten EasyIO. Klargjør for utskrift til fil med "import java.io.*;" og "PrintWriter utfil = new PrintWriter(new File("minfil.txt"));". Deretter bruker du utfil.print/println/printf for utskrift til fil på samme måte som du ville brukt System.out.print/println/printf for

utskrift til skjerm. Til slutt utfil.close(); I tillegg bør du ha "try{" i begynnelsen av metoden (før setningen med "new PrintWriter") og følgende på slutten av metoden (du vil lære mer om dette i INF1010, foreløpig er det nok at du bare skriver det helt likt): "} catch (Exception e) { e.printStackTrace(); }".

(b) Utvid programmet slik at det også leser innholdet i filen som ble opprettet i del (a), lagrer det i fire passende variabler (den første skal være av typen `char`, osv.), og skriver til slutt verdiene i variablene ut på skjerm.
Hint: Innlesing fra fil programmeres som innlesing fra tastatur, bare at man angir filnanvet under klargjøring, f.eks. "In innfil = new In("minfil.txt");", og bruker gjerne et annet navn på forbindelsen (f.eks. "innfil" i stedet for "tast"). Hvis du vil prøve uten EasyIO klargjør med "Scanner innfil = new Scanner(new File("minfil.txt"));" og legg try- og catch rundt metoden (se hintet i del (a)).

7. **Lese to arrayer fra fil:** *kap. 5, oppg. 3 (side 99)*
Lag et program som skal behandle data om vekten til elevene i en skoleklasse. Det er 27 elever i klassen. Dataene ligger på fil slik:

```
Jens 52
Marit 43
...
```

(a) Les dataene inn i to arrayer: en navnearray (`String`) og en vektarray (`int`). ***Hint:*** *Se "// Fil til array" på side 62 i læreboka.*
(b) La programmet finne høyeste og laveste vekt og skrive ut navn og vekt på disse.
(c) La programmet beregne gjennomsnittsvekten i klassen.

```
import easyIO.*;

class ElevVekt {
    public static void main(String[] args) {
        In innfil = new In("elevvekt.txt");
        final int ANT_ELEVER = 27;
        String[] navn = new String[ANT_ELEVER];
        int[] vekt = new int[ANT_ELEVER];

        // (a): Leser dataene fra fil inn i to arrayer:
        for (int i = 0; i < ANT_ELEVER; i++) {
            navn[i] = innfil.inWord();
            vekt[i] = innfil.inInt();
        }

        // (b): Finner indeksene i arrayen der vekt er høyest og lavest:
        int høyVektIndeks = 0;
        int lavVektIndeks = 0;
        for (int i = 0; i < vekt.length; i++) {
            if (vekt[i] > vekt[høyVektIndeks]) {
                høyVektIndeks = i;
            }
            if (vekt[i] < vekt[lavVektIndeks]) {
                lavVektIndeks = i;
            }
        }
        System.out.println("Tyngste elev: " + navn[høyVektIndeks] + " "
                           + vekt[høyVektIndeks] + " kg");
        System.out.println("Letteste elev: " + navn[lavVektIndeks] + " "
                           + vekt[lavVektIndeks] + " kg");

        // (c): Gjennomsnittsvekt:
        ElevVekt ev = new ElevVekt();
        double snitt = ev.finnGjennomsnitt(vekt);
        System.out.printf("Gjennomsnittsvekt: %.1f\n", snitt);

    }

    double finnGjennomsnitt(int[] a) {
        int sum = 0;
        for (int i = 0; i < a.length; i++) {
            sum += a[i];
        }
        return (double) sum / a.length;
    }
}

KJØREEKSEMPEL:
Tyngste elev: Jens 53 kg
Letteste elev: Anna 33 kg
Gjennomsnittsvekt: 44.8
```

8. **Finn feil:**
Når vi prøver å kompilere og kjøre dette programmet får vi feilmeldingene vist nedenfor. Hva er feil?

```
1  class FinnFeil {
2      int[] fat = new int[5];
3      System.out.println("Feilplassert");
4
5      public static void main(String[] args) {
6          FinnFeil ff = new FinnFeil();
```

```
7           ff.metode();
8       }
9
10      void metode() {
11          for (int i = 0; i < 5; i++) {
12              fat[i] = 10 * i;
13          }
14
15          System.out.println(fat[5]);
16          System.out.println(fat2);
17          System.out.println(fat[1], 2);
18      }
19  }
```

Hva betyr disse feilmeldingene, og hvordan retter vi feilene?

```
(a)
  FinnFeil.java:3: <identifier> expected
      System.out.println("Feilplassert");
                         ^
  FinnFeil.java:3: illegal start of type
      System.out.println("Feilplassert");^

(b)
  Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
          at FinnFeil.metode(FinnFeil.java:15)
          at FinnFeil.main(FinnFeil.java:7)

(c)
  FinnFeil.java:16: cannot find symbol
  symbol  : variable fat2
  location: class FinnFeil
          System.out.println(fat2);
                             ^
(d)
  FinnFeil.java:17: cannot find symbol
  symbol  : method println(int,int)
  location: class java.io.PrintStream
          System.out.println(fat[1], 2);
                     ^
```

9. **Metode med inn og ut-parametre:** *kap. 7, oppg. 2 (side 133)*
Lag en metode som regner ut hypotenusen *c* i en rettvinklet trekant når vi går ut fra Pytagoras formel: $c^2 = a^2 + b^2$ der *a* og *b* er lengden på katetene - de to andre sidene i trekanten. Vi bruker *a* og *b* som parametre til metoden. Du trenger da å kalle kvadratrotmetoden i `Math.sqrt(`*`double_verdi`*`)` i den metoden du lager. Returner verdien *c* som verdien på metoden. Test metoden ved å kalle den i en dobbel for-løkke for alle kombinasjoner av *a* og *b* med heltallsverdiene fra 1.0 til og med 6.0, og skriv ut svarene.

```
double finnHypotenus(double a, double b) {
    // ...
    return c;
}
```

Hint: Kallet på metoden kan se slik ut: `double c = finnHypotenus(a, b);` ...eller: `System.out.println(finnHypotenus(a, b));`

```
class Hypotenus {
    public static void main(String[] args) {
        HypotenusMetoder h = new HypotenusMetoder();
        h.start();
    }
}
class HypotenusMetoder {
    void start() {
        for (double a = 1; a <= 6; a++) {
            for (double b = 1; b <= 6; b++) {
                double c = finnHypotenus(a, b);
                System.out.printf("a=%.1f, b=%.1f  gir c=%.3f\n", a, b, c);
            }
        }
    }

    double finnHypotenus(double a, double b) {
        double c = Math.sqrt(a * a + b * b);
        return c;
    }
}

KJØREEKSEMPEL:
a=1.0, b=1.0  gir c=1.414
a=1.0, b=2.0  gir c=2.236
a=1.0, b=3.0  gir c=3.162
a=1.0, b=4.0  gir c=4.123
a=1.0, b=5.0  gir c=5.099
a=1.0, b=6.0  gir c=6.083
a=2.0, b=1.0  gir c=2.236
a=2.0, b=2.0  gir c=2.828
a=2.0, b=3.0  gir c=3.606
a=2.0, b=4.0  gir c=4.472
a=2.0, b=5.0  gir c=5.385
a=2.0, b=6.0  gir c=6.325
a=3.0, b=1.0  gir c=3.162
```

```
a=3.0, b=2.0  gir c=3.606
  ...osv...
a=6.0, b=5.0  gir c=7.810
a=6.0, b=6.0  gir c=8.485
```

10. **Mer om metoder**
Fullfør følgende program, som viser bruk av metoder. Angi også hva programmet skriver ut.

```
import java.util.Scanner; // Tilsvarer: import easyIO.*;

class Metoder {
    public static void main(String[] args) {
        TestMetoder tm = new TestMetoder();
        tm.start();
    }
}

class TestMetoder {
    Scanner tast = new Scanner(System.in); // Tilsvarer: In tast = new In();

    void start() {
        // Kaller en enkel metode:
        metode1();

        // Kaller en metode med én inn-parameter:
        skrivTredoblet(123);

        // Leser to tall fra tastatur, og overfører de til en metode
        // som finner og skriver ut det høyeste av de to tall:
        skjerm.out("Skriv to tall (f.eks. 7 4): ");
        int tall1 = tast.nextInt(); // Tilsvarer: ... = tast.inInt();
        int tall2 = tast.nextInt(); // Tilsvarer: ... = tast.inInt();
        finnHøyesteAv2(tall1, tall2);

        // Kaller en metode som multipliserer de samme to tall lest
        // inn ovenfor, og returnerer resultatet hit.
        int resultat = multipliser( /* Fyll inn resten. . . */ );
        skjerm.outln("Resultat multiplisert: " + resultat);

        // Metode med array som inn-parameter:
        double[] verdier = { 0, -3, 5, 10, -20, -7.7, 1.2, -0.01 };
        //int antNeg = finnAntallNegativeTall(verdier);
        //skjerm.outln("Arrayen verdier[] har " + antNeg + " negative tall.");
        // <- Ta bort "//" i de to linjene over.
    }

    void metode1() {
        skjerm.outln("Dette er metode1");
    }

    void skrivTredoblet(int x) {
        int tredoblet = x * 3;
        skjerm.outln("Tredoblet resultat = " + tredoblet);
    }

    void finnHøyesteAv2(int a, int b) {
        // Hva mangler her?
        // . . .

        skjerm.outln("Høyest av de to tall er:" /* . . . */ );
    }

    int multipliser( /* Fyll inn resten. . . */) {
        // . . .

        return  0 ; // . . . Erstatt 0 med resultatet av x ganger y.
    }

    // Skriv metoden "finnAntallNegativeTall" her, som har en array med
    // double-verdier som inn-parameter, finner ut hvor mange av verdiene
    // i arrayen er negative tall, og returnerer det antallet (som en int).

    // . . .
}
```

```
import easyIO.*;

class Metoder {
    public static void main(String[] args) {
        TestMetoder tm = new TestMetoder();
        tm.start();
    }
}

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

    void start() {
        // Kaller en enkel metode:
        metode1();

        // Kaller en metode med én inn-parameter:
        skrivTredoblet(123);
```

```
        // Leser to tall fra tastatur, og overfører de til en metode
        // som finner og skriver ut det høyeste av de to tall:
        skjerm.out("Skriv to tall (f.eks. 7 4): ");
        int tall1 = tast.inInt();
        int tall2 = tast.inInt();
        finnHøyesteAv2(tall1, tall2);

        // Kaller en metode som multipliserer de samme to tall lest
        // inn ovenfor, og returnerer resultatet hit.
        int resultat = multipliser(tall1, tall2);
        skjerm.outln("Resultat multiplisert: " + resultat);

        // Metode med array som inn-parameter:
        double[] verdier = { 0, -3, 5, 10, -20, -7.7, 1.2, -0.01 };
        int antNeg = finnAntallNegativeTall(verdier);
        skjerm.outln("Arrayen verdier[] har " + antNeg + " negative tall.");
    }

    void metode1() {
        skjerm.outln("Dette er metode1");
    }

    void skrivTredoblet(int x) {
        int tredoblet = x * 3;
        skjerm.outln("Tredoblet resultat = " + tredoblet);
    }

    void finnHøyesteAv2(int a, int b) {
        int høyeste = a;
        if (b > a) {
            høyeste = b;
        }
        skjerm.outln("Høyest av de to tall er: " + høyeste);
    }

    int multipliser(int a, int b) {
        int produkt = a * b;
        return produkt;
    }

    // Skriv metoden "finnAntallNegativeTall" her, som har en array med
    // double-verdier som inn-parameter, finner ut hvor mange av verdiene
    // i arrayen er negative tall, og returnerer det antallet (som en int).
    int finnAntallNegativeTall(double[] verdier) {
        int antNegative = 0;
        for (int i = 0; i < verdier.length; i++) {
            if (verdier[i] < 0) {
                antNegative++;
            }
        }
        return antNegative;
    }
}

KJØREEKSEMPEL:
> java Metoder
Dette er metode1
Tredoblet resultat = 369
Skriv to tall (f.eks. 7 4): 100 55
Høyest av de to tall er: 100
Resultat multiplisert: 5500
Arrayen verdier[] har 4 negative tall.
```

11. **Fibonacci-tallene**
(a) **Løkker:** Lag et program som skriver ut de 15 første tall i Fibonaccifølgen. Følgen er definert ved at de to første tall er 0 og 1, og hvert neste tall er summen av de to foregående. Utskriften skal bli:

```
 0  1  1  2  3  5  8  13  21  34  55  89  144  233  377
```

Mer info: Fibonacci-tallene forekommer mye i naturen, bl.a. i tregrener, blomster, kongler og kaniner.

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

        int nestSiste = 0;
        int siste = 1;
        // Skriver ut de to første tall:
        System.out.print(nestSiste + "  ");
        System.out.print(siste + "  ");

        // Skriver ut 3. til 15. tall i følgen:
        for (int i = 3; i <= 15; i++) {
            int tmp = siste; // Midlertidig variabel
            siste = siste + nestSiste;
            nestSiste = tmp;
            System.out.print(siste + "  ");
        }
        System.out.println();
    }
}
```

(b) **Metoder:** Endre strukturen i Fibonacci-programmet du lagde ovenfor slik at det følger malen fra Oblig 2 (vist

også i oppgave nr. 8 fra forrige uke), med en kontrollklasse øverst, etterfulgt av en hjelpeklasse for metodene. Bruk tre metoder i hjelpeklassen: en ordreløkke som skriver ut følgende meny, og en metode for hvert av de 2 menyvalgene:

```
1. Skriv ut de 15 første tall i Fibonaccifølgen
2. Test om et tall hører til følgen
```

Metoden for ordre 2 ber bruker taste inn et tall, og går så i en løkke som genererer Fibonacci-tallene frem til det bruker-inntastede tallet. Deretter gis det melding til bruker om tallet hørte til følgen eller ikke.

```
import easyIO.*;

class Fibonacci {
    public static void main(String[] args) {
        FibonacciMetoder fib = new FibonacciMetoder();
        fib.ordreløkke();
    }
}

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

    void ordreløkke() {
        int ordre = 0;

        while (ordre != 3) {
            // Meny:
            skjerm.outln("1. Skriv ut de 15 første tall i Fibonaccifølgen");
            skjerm.outln("2. Test om et tall hører til følgen");

            // Leser kommando fra bruker:
            skjerm.out("Velg kommando (3=Avslutt): ");
            ordre = tast.inInt();

            switch (ordre) {
                case 1: skriv15Fib(); break;
                case 2: testFibTall(); break;
                default: break;
            }
            System.out.println();
        }
    }

    void skriv15Fib() {
        int nestSiste = 0;
        int siste = 1;
        System.out.print(nestSiste + "  ");
        System.out.print(siste + "  ");

        for (int i = 3; i <= 15; i++) {
            int tmp = siste; // Midlertidig variabel
            siste = siste + nestSiste;
            nestSiste = tmp;
            System.out.print(siste + "  ");
        }
        System.out.println();
    }

    void testFibTall() {
        skjerm.outln();
        skjerm.out("Skriv et tall for å teste om det er Fibonacci-tall: ");
        int inntastet = tast.inInt();

        int nestSiste = 0;
        int siste = 1;
        int i = 2; // i: Plass i Fib.følgen til nåværende "siste"
        // Så lenge "siste" er < enn inntastet generér neste Fib.-tall:
        for ( ; siste < inntastet; i++) {
            int tmp = siste;
            siste = siste + nestSiste;
            nestSiste = tmp;
        }
        if (siste == inntastet) {
            System.out.println(inntastet + " er Fibonacci-tall nr. " + i);
        } else if (inntastet == 0) {
            System.out.println(inntastet + " er Fibonacci-tall nr. 1");
        } else {
            System.out.println(inntastet + " er IKKE et Fibonacci-tall.");
        }
    }
}

KJØREEKSEMPEL:
> java Fibonacci
1. Skriv ut de 15 første tall i Fibonaccifølgen
2. Test om et tall hører til følgen
Velg kommando (3=Avslutt): 1
0  1  1  2  3  5  8  13  21  34  55  89  144  233  377

1. Skriv ut de 15 første tall i Fibonaccifølgen
2. Test om et tall hører til følgen
Velg kommando (3=Avslutt): 2

Skriv et tall for å teste om det er Fibonacci-tall: 233
233 er Fibonacci-tall nr. 14
```

```
1. Skriv ut de 15 første tall i Fibonaccifølgen
2. Test om et tall hører til følgen
Velg kommando (3=Avslutt): 2

Skriv et tall for å teste om det er Fibonacci-tall: 234
234 er IKKE et Fibonacci-tall.
```

12. **Ukens nøtt 1: Nærmeste Fibonacci-tall** *(middels vanskelig)*
Utvid ordre 2 i foregående oppgave slik at den også sier hvilket Fibonacci-tall er nærmest det bruker-inntastede tallet. Hvis to er like nærme, f.eks. hvis bruker tastet inn "4", skriv ut begge (i dette tilfellet "3" og "5").

13. **Ukens nøtt 2: Sudoku hjelpeprogram** *(middels vanskelig)*
(a) Lag et program som leser inn en Suduko-oppgave fra fil og lagrer de forhåndsutfylte tallene i en 2-dimensjonal array. Deretter går programmet i en løkke som spør brukeren om et rad- og et kolonnenummer (i området 1-9, eller 0 for å avslutte). Programmet skal så svare brukeren med hvilke tall (1-9) som er mulige kandidater for plassering i den angitte rad/kolonne-plassen, ved å finne ut hvilke av sifrene 1-9 ikke er allerede brukt i samme rad, kolonne, eller 3×3-omsluttende boks.

    Input-filen er på 9 linjer, med 9 tall per linje adskilt med mellomrom, og hvor 0 angir plassene som ikke har forhåndsutfylt siffer i Sudoku-oppgaven. Her er et eksempel på en slik fil (med middels vanskelig oppgave). Finn gjerne andre oppgaver fra aviser eller nettet.

```
6 0 7 0 0 0 0 8 0
0 0 0 1 0 4 0 7 0
0 0 5 0 0 8 0 3 0
8 0 0 3 0 0 7 0 0
4 0 0 5 0 6 0 0 8
0 0 1 0 0 2 0 0 6
0 8 0 4 0 0 5 0 0
0 9 0 2 0 3 0 0 0
0 7 0 0 0 0 1 0 3
```

    (b) Utvid deretter programmet slik at det går gjennom *alle* ikke-utfylte ruter (i stedet for å be brukeren taste en), og for de rutene som bare har ett kandidatsiffer setter du sifferet inn i arrayen. Gjenta prosessen med de gjenværende ikke-utfylte ruter helt til to påfølgende gjennomkjøringer ikke finner nye tall å sette inn. Skriv ut resultatbrettet til slutt.

14. **Tips til Emacs: Linjenummer på hver linje**
Du kan få linjenummer på hver linje i Emacs ved å lime inn litt kode i din "~/.emacs"-fil, som forklart i veiledningen i Emacs-oppgavene fra Forkurs i informatikk.

Tibakemelding om dette oppgavesettet kan du skrive i bloggen eller sende på mail til josek *[a]* ifi.uio.no