Løsningsforslag ukeoppg. 3: 7. - 13. sep (INF1000 - Høst 2011)

Utskrift/ lesing med easyIO, arrayer, løkker *(kapittel 3-4 i læreboka, "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
Øve på bruk av uttrykk, innlesing og utskrift til terminal, løkker, og arrayer.

🔑 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. **Presedensregler — i hvilken rekkefølge utføres regneoperasjonene?**
   Avgjør i hvert av disse tilfellene, uten å bruke datamaskin, hvilken verdi som blir skrevet ut på skjermen.

   For å svare på denne oppgaven må du kjenne reglene for evaluering (utregning) av numeriske uttrykk i Java. Kortversjonen av disse er at uttrykk beregnes fra venstre mot høyre, men multiplikasjoner og divisjoner utføres før addisjoner og subtraksjoner. Du kan lese mer om dette på side 39 i læreboka.

```
1.  System.out.println(3 * 4 + 5);          // 17
2.  System.out.println(3 + 4 * 5);          // 23
3.  System.out.println((3 + 4) * 5);        // 35
4.  System.out.println(1 / 2 * 5);          // 0
5.  System.out.println(1 / 2 * 5.0);        // 0.0
6.  System.out.println(1.0 / 2 * 5);        // 2.5
7.  System.out.println(4 * 2 * 2 * 1);      // 16
8.  System.out.println(4 / 2 / 2 / 1);      // 1   |< Unngå slike uttrykk!,
9.  System.out.println(5 * 4 / 3 * 2 / 1);  // 12  |< bruk heller parenteser!
```

   Hvis du ønsker mer info kan du se følgende oversikt med presedens for alle operatorene. Jo høyere oppe på denne lista operatoren står jo før blir den utført, mens de som står på samme nivå er likestilt og utføres fra venstre til høyre. Tabellen er tatt fra www.janeg.ca/scjp/oper/precedence.html, og er basert på side 378 i *The Java Programming Language*:

Operator type	Operators
Postfix	`[] . (params) expr++ expr--`
Unary	`++expr --expr +expr -expr ~ !`
Creation or Cast	`new (type)expr`
Multiplicative	`* / %`
Additive	`+ -`
Shift	`<< >> >>>`
Relational	`< > >= <= instanceof`
Equality	`== !=`
Bitwise AND	`&`
Bitwise exclusive OR	`^`
Bitwise inclusive OR	`|`
Logical AND	`&&`
Logical OR	`||`
Ternary	`?:`
Assignment	`= += -= *= /= %= >>= <<= >>>= &= ^= |=`

2. **Logiske uttrykk:** *(tabell 2.5, side 41)*
Avgjør hvilken verdi den boolske variabelen *b* får etter hver av disse tilordningssetningene, når vi antar at heltallsvariablene *x*, *y*, og *z* har fått verdier som følger: `int x = 3, y = 4, z = 1;`

```
1.  b = (x < y) && (y < z);        // false
2.  b = (x < y) || (y < z);        // true
3.  b = ! (x < y);                 // false
4.  b = x < y;                     // true
5.  b = x <= y;                    // true
6.  b = (x == y);                  // false
7.  b = (x != y);                  // true
```

Hint: Eksempel på hvordan man kan bruke slike boolske uttrykk i et program:

```
int x = 3, y = 4, z = 1;
if ((x < y) && (y < z)) {
    System.out.println("Det stemmer at x < y < z."); // Blir dette skrevet ut?
}
```

3. **Innlesing fra terminal:** *kap. 3, oppg. 3 (side 72)*
Lag et program som ber om og leser inn to flyttall. Programmet skal deretter regne ut produktet av de to tallene og skrive ut svaret. Her er et eksempel på hvordan en kjøring av programmet kan se ut:

```
Oppgi verdien til x: 9
Oppgi verdien til y: 10
Produktet av x og y er 90.0
```

Hint: Innlesing fra terminal bør gjøres i to steg, først en utskrift som sier til bruker hva hun skal taste inn (det kalles for *ledetekst*), og så kommer selve setningen som henter input fra tastaturet, til dette kan du bruke enten EasyIO eller Javas innebygde funksjoner:

- **EasyIO:** Følg malen fra eksemplet på side 58. Bruk `tast.inDouble()` og ikke `tast.inInt()` slik at bruker kan taste inn et flyttall.
- **Scanner:** ***NB!*** Dette er bare ment **for spesielt interesserte** som vil prøve uten EasyIO.
Se denne tabellen i bloggen med programsetninger i Java 1.5 som tilsvarer EasyIO.

I begge tilfeller så vil programsetningen som leser inn tallet (tast.inInt() eller scan.nextInt()) stoppe kjøringen av programmet midlertidig inntil bruker har tastet inn et svar. Når bruker har gjort det, blir tallet lagret i den angitte variabelen, og kjøringen av programmet fortsetter med neste setning.

Løsningsforslag med EasyIO:
Opprett filen `Produkt.java` med følgende linjer som innhold:

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

        skjerm.out("Oppgi verdien til x: ");
        double x = tast.inDouble();

        skjerm.out("Oppgi verdien til y: ");
        double y = tast.inDouble();

        double produkt = x * y;
        skjerm.outln("Produktet av x og y er: " + produkt);
    }
}
```

```
KJØREEKSEMPEL:
> java Produkt
Oppgi verdien til x: 9.123
Oppgi verdien til y: 10
Produktet av x og y er: 91.22999999999999
```

KOMMENTAR:
Grunnen til at det ikke blir nøyaktig 91.23 er at Java internt lagrer slike double-verdier i det binære tallsystemet, som ikke kan representere helt nøyaktig visse desimalsifre etter komma. Når så Java konverterer tilbake til det desimale tallsystemet ved utskrift, så kan det derfor bli små feil som vist

over. To løsninger på dette er vist i neste oppgave.

Løsningsforslag med Scanner:
Kommer...

4. **Utskrift med 2 desimaler på to måter: easyIO og printf**
Ved utskrift av desimaltall er det ofte ønskelig å kontrollere hvor mange desimaler som skrives ut. Bruker vi `System.out.println(tall)` til å skrive ut en **`double`**-variabel *tall*, får vi med alle desimalene som er lagret i *tall* (med unntak av avsluttende nuller i desimalene). To måter å kontrollere antall desimaler som skrives ut er:

   - **EasyIO:** Hvis pakken EasyIO er installert på maskinen du bruker så kan man få skrevet ut variabelen *tall* med 2 desimaler etter komma, og evt. på akkurat 7 plasser ved hjelp av følgende setninger. I det andre eksemplet så fyller Java på med blanke tegn til venstre slik at hele utskriften tar 7 plasser (eller så mange plasser som er nødvendig):
     ```
     skjerm.outln(tall, 2);
     skjerm.outln(tall, 2, 7);
     ```
     For å kunne bruke dette må programmet ha setningnen "`import easyIO.*;`" øverst, og setningen "`Out skjerm = new Out();`" plassert på et passende sted i programmet før de ovennevnte `skjerm.outln()`-linjene, se eksemplet på side 52 i læreboka. I tillegg må easyIO være installert på datamaskinen (mer om dette kan du lese i bloggen). For info om andre muligheter for formatert utskrift med easyIO se oversikten på side 54. Ledeteksten `"Tall: "` kan skrives ut vha. en egen `skjerm.out("Tall: ")`-setning som du kan plassere før de ovennevnte `skjerm.outln`-setningene.

   - **Printf:** ***NB!*** Dette er bare ment **for spesielt interesserte** som vil prøve uten EasyIO.
     I de vanligste brukte versjoner av Java (5.0, 6, og 7, også kalt 1.5, 1.6, og 1.7) finnes en metode kalt "printf" som gir oss god kontroll over utskriften, følgende setninger gir samme resultat som EasyIO-setningene ovenfor:
     ```
     System.out.printf("Tall: %.2f\n", tall);
     System.out.printf("Tall: %7.2f\n", tall);
     ```
     Koden `\n` (omvendt-skråstrek etterfulgt av n) gir et linjeskift, og kan tas bort i setningene over hvis du ikke ønsker linjeskift. Flere eksempler på bruk av `printf` kan du se her (PDF) [Y.D. Liang *"Introduction to Java Programming"* 5. utg., side 64], og på Wikipedia.

Oppgave: Bruk dette til å endre programmet fra forrige oppgave (oppgave 3 ovenfor) til å skrive ut produktet med 2 desimaler, og igjen med 3 desimaler på 10 plasser.

```
import easyIO.*;
class Produkt2 {
    public static void main(String[] args) {
        In tast = new In();
        Out skjerm = new Out();
        double x;
        double y;
        double produkt;

        skjerm.out("Oppgi verdien til x: ");
        x = tast.inDouble();

        skjerm.out("Oppgi verdien til y: ");
        y = tast.inDouble();

        produkt = x * y;

        // easyIO:
        skjerm.out("Produktet av x og y er: ");
        skjerm.out(produkt, 2);
        skjerm.out(" = ");
        skjerm.outln(produkt, 3, 10);

        // printf:
        skjerm.out("Produktet av x og y er: ");
        System.out.printf("%.2f", produkt);
        System.out.printf(" = %10.3f\n", produkt);
    }
}

KJØREEKSEMPEL:
> java Produkt2
Oppgi verdien til x: 9.123
Oppgi verdien til y: 10
Produktet av x og y er: 91.23 =     91.230
Produktet av x og y er: 91.23 =     91.230
```

5. **Valuta-omregning:** *kap. 3, oppg. 4 (side 73)*
Lag et program som leser inn et kronebeløp **fra tastaturet**, og omregner beløpet til amerikanske dollar, euro, og svenske kroner; og skriver ut resultatet på skjermen **med to desimaler**. Omregningen er etter følgende kurser: 1 USD = 5.435 kr; 1 EUR = 7.682 kr; 1 SEK = 0.8462 kr. (*NB! Vekslingskursene er oppdatert 6. sep 2011, det står andre kurs og valutaer i læreboka.*) [Valutakurser]. ***Hint:*** Hvis brukeren taster 100 skal resultatet i USD bli $18.40.

```
import easyIO.*;

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

        // Valutakurser 29. aug 2008:
        final double NOK_PR_USD = 5.435;
        final double NOK_PR_EUR = 7.682;
        final double NOK_PR_SEK = 0.8462;

        // Leser input fra bruker:
        System.out.print("Angi antall kr: "); // Ledetekst
        double antallKr = tast.inDouble();

        double antallUSD = antallKr / NOK_PR_USD;
        double antallEUR = antallKr / NOK_PR_EUR;
        double antallSEK = antallKr / NOK_PR_SEK;

        // 2 desimaler vha. easyIO:
        skjerm.out("Det tilsvarer i USD: $");
        skjerm.outln(antallUSD, 2);

        skjerm.out("  i euro: EUR");
        skjerm.outln(antallEUR, 2);

        // 2 desimaler vha. printf:
        System.out.printf("  i svenske kroner: SEK %.2f\n", antallSEK);
    }
}

KJØREEKSEMPEL:
> java Valuta
Angi antall kr: 100
Det tilsvarer i USD: $18.40
  i euro: EUR13.02
  i svenske kroner: SEK 118.18
```

6. **For-løkke:** *kap. 4, oppg. 4 (side 85)*
Lag et program som skriver ut på skjermen omkretsene til sirkler med radiusene r = 1, 2, ..., 10 (omkretsen *O* beregnes etter formelen $O = 2\pi r$. Sett $\pi = 3.14$). Utskriften skal følge mønsteret:

```
Radius = 1 gir omkrets = 6.28
Radius = 2 gir omkrets = 12.57
..osv..
```

```
import easyIO.*;

class Omkrets1 {
    public static void main(String[] args) {
        Out skjerm = new Out();
        double omkrets;

        for (int radius = 1; radius <= 10; radius++) {
            omkrets = 2.0 * 3.14 * radius;
            skjerm.out("Radius = " + radius + " gir omkrets = ");
            skjerm.outln(omkrets, 2);
        }
    }
}

KJØREEKSEMPEL:
> java Omkrets
Radius = 1 gir omkrets = 6.28
Radius = 2 gir omkrets = 12.57
Radius = 3 gir omkrets = 18.85
Radius = 4 gir omkrets = 25.13
Radius = 5 gir omkrets = 31.42
Radius = 6 gir omkrets = 37.70
Radius = 7 gir omkrets = 43.98
Radius = 8 gir omkrets = 50.27
Radius = 9 gir omkrets = 56.55
Radius = 10 gir omkrets = 62.83
```

7. **While-løkke:** *(b): kap. 4, oppg. 5 (side 85)*
(a) Gjenta forrige oppgave, men bruk while-løkke i stedet.

```
import easyIO.*;
class Omkrets2 {
    public static void main(String[] args) {
        Out skjerm = new Out();
        double omkrets;

        int radius = 1;
        while (radius <= 10) {
            omkrets = 2.0 * 3.14 * radius;
            skjerm.out("Radius = " + radius + " gir omkrets = ");
            skjerm.outln(omkrets, 2);
            radius++;
        }
    }
}
```

(b) Som forrige oppgave, men utskriften skal nå først stoppe når omkretsen overstiger 1000. Tips: Bruk `while`-løkken til å kontrollere *O*.

```
import easyIO.*;
class Omkrets3 {
    public static void main(String[] args) {
        Out skjerm = new Out();
        double omkrets = 0;

        int radius = 1;
        while (omkrets <= 1000) {
            omkrets = 2.0 * 3.14 * radius;
            skjerm.out("Radius = " + radius + " gir omkrets = ");
            skjerm.outln(omkrets, 2);
            radius++;
        }
    }
}
```

8. **Summerings-løkke:** *kap. 4, oppg. 6 (side 85)*
(a) Skriv et program som leser ett heltall *n* fra terminal, og som deretter summerer tallene fra 1 til *n*, dvs. 1 + 2 + ⋯ + *n*, og skriver ut hver av mellomresultatene (1; 1+2; 1+2+3; osv.), og sluttsummen. Utskriften skal følge dette mønsteret (når *n* = 5):

```
  1    1
  2    3
  3    6
  4   10
  5   15
```

```
import easyIO.*;

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

        skjerm.out("Angi n: ");
        int n = tast.inInt();

        int sum = 0;
        for (int i = 1; i <= n; i++) {
            sum = sum + i;
            skjerm.out(i, 3);
            skjerm.outln(sum, 5);
        }
    }
}

KJØREEKSEMPEL:
> java Heltallssum
Angi n: 5
  1    1
  2    3
  3    6
  4   10
  5   15
```

(b) Det finnes også en formel som gir summen av tallene 1, 2, 3, ..., *n* direkte: *n* * (*n* + 1) / 2. Utvid programmet slik at det til slutt sammenligner siste sum med resultatet av formelen, og gir en melding til bruker på om de to svarene var like eller ikke.

```
        if (sum == n * (n + 1) / 2) {
            System.out.println("Formelen stemmer!");
        } else {
            System.out.println("Formelen stemmer ikke.");
        }
```

9. **Løkker: Hva blir skrevet ut?**
Anta at følgende programsetninger utføres. Hva skrives ut på skjermen?

```
class Ukeoppg2_8 {
    public static void main(String[] args) {

//(a)
        int a = 1;
        while (a < 5) {
            a = a + 1;
        }
        System.out.println("a = " + a);
```

```
a = 5
```

```
//(b)
        int b = 11;
        while (b < 14) {
            b++;
            System.out.println(b);
        }
```

```
12
13
14
```

```
//(c)
        int c = 1;
        while (c < 10) {
            c = -2 * c;
        }
        System.out.println("c = " + c);
```

```
c = 16
```

```
//(d)
        for (int d = 0; d < 3; d++) {
            System.out.println(d);
        }
```

```
0
1
2
```

```
//(e)
        for (int e = 1; e <= 3; e++) {
            for (int f = 1; f <= 2; f++) {
                System.out.println(e + " " + f);
            }
        }
```

```
1 1
1 2
2 1
2 2
3 1
3 2
```

```
//(f)
        for (int ytre = 0; ytre < 2; ytre++) {
            System.out.print("[");

            for (int indre = 0; indre < 3; indre++) {
                System.out.print(".");
            }
            System.out.println("]");
        }
```

```
[...]
[...]
```

10. **En enkel array:**

```
int[] a = new int[20];
```

(a) Når setningen over utføres, skjer det både en deklarasjon og en oppretting av et array-objekt. Forklar hvilken del av setningen som gjør hva, og vis hvordan setningen kunne vært splittet opp i to setninger: en deklarasjonssetning og en setning som oppretter array-objektet.

```
int[] a;
a = new int[20];
```

(b) Her er eksempler på bruk av ovennevnte array. Hva blir skrevet ut på skjermen?

```
int[] a = new int[20];
a[0] = 100;
a[1] = a[0] * 2;
System.out.println(a[1]);
a[0]++;
System.out.println(a[0]);
System.out.println(a.length);
```

```
200
101
20
```

11. **Array med tall:**

(a) Lag et program som ber bruker taste inn tre heltall og lagrer disse i en array kalt *tall*:

```
int[] tall = new int[3];
```

(b) Sum av array: Utvid programmet slik at det regner ut summen av tallene ved hjelp av en løkke, og skriver ut resultatet.
(c) Minste verdi: Utvid programmet slik at det finner og skriver ut det minste tallet i arrayen.
(d) Lave verdier: Legg til programkode som skriver ut alle verdiene i arrayen som er mindre enn 10.
(e) Søk: Legg til programkode som skriver ut en beskjed om verdien 5 finnes eller ikke finnes i arrayen.

```
import easyIO.*;
class Ukeoppg3_3 {
    public static void main(String[] args) {
        In tast = new In();
        Out skjerm = new Out();

        int[] tall = new int[3];

        // (a) Lese 3 tall fra tastatur:
        for (int i = 0; i < 3; i++) {
            // Ledetekst:
            skjerm.out("Angi tall[" + i + "]: ");

            // Les et tall fra tastatur og lagre det i arrayen tall[]:
            tall[i] = tast.inInt();
        }

        // (b) Sum av array:
        int sum = 0;
        for (int i = 0; i < 3; i++) {
            sum += tall[i];
        }
        skjerm.outln("(b) Sum av tallene = " + sum);

        // (c) Minste verdi:
        int minste = tall[0];
        for (int i = 0; i < 3; i++) {
            if (tall[i] < minste) {
                minste = tall[i];
            }
        }
        skjerm.outln("(c) Minste verdi = " + minste);

        // (d) Lave verdier:
        skjerm.out("(d) Lave verdier (< 10): ");
        for (int i = 0; i < 3; i++) {
            if (tall[i] < 10) {
                skjerm.out(tall[i] + " ");
            }
        }
        skjerm.outln();

        // (e) Søk:
        boolean funnet = false;
        for (int i = 0; i < 3; i++) {
            if (tall[i] == 5) {
                funnet = true;
            }
        }
        skjerm.out("(e) Verdien 5 finnes ");
        if (! funnet) {
            skjerm.out("ikke ");
        }
        skjerm.outln("i arrayen");
    }
}

KJØREEKSEMPEL:
> java Ukeoppg3_3
```

```
Angi tall[0]: 2
Angi tall[1]: 7
Angi tall[2]: 11
(b) Sum av tallene = 20
(c) Minste verdi = 2
(d) Lave verdier (< 10): 2 7
(e) Verdien 5 finnes ikke i arrayen
```

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