INF1000 - Uke 10

Løsningsforslag ukeoppgaver 10

29. oktober 2012

Løsningsforslag

Oppgave 1

Legg inn i et hashmap:

Du har klassen Dyr (se under) Lag en klasse DyreMap, som inneholder en main-metode. Her skal du deklarere et `HashMap<String, Dyr>`. Opprett minst tre objekter av klassen Dyr, finn på navn og idnr. Husk at idnr bør være unike! Less disse inn i HashMappet med idnr som nøkkel.

```
class Dyr {
    String navn;
    String idnr;

    Dyr(String navn, String idnr) {
        this.navn = navn;
        this.idnr = idnr;
    }

    String getIDnr() {
        return idnr;
    }
}
```

Løsningsforslag

```
import java.util.*;

class DyreHashMap {
    public static void main(String[] args) {
        // Oppretter et HashMap med dyr:
        HashMap<String, Dyr> dyr = new HashMap<String, Dyr>();

        // Oppretter tre dyre-objekter og legger dem inn i HashMappet:
        Dyr d1 = new Dyr("Albert", "1234");
        Dyr d2 = new Dyr("Shaun", "2345");
        Dyr d3 = new Dyr("Mathilde", "3456");

        dyr.put(d1.getIDnr(), d1);
```

```
        dyr.put(d2.getIDnr(), d2);
        dyr.put(d3.getIDnr(), d3);

        // Løper gjennom HashMappet og ser at dyrene er lagt inn
        for (Dyr d: dyr.values()) {
            System.out.println(d.navn + " " + d.getIDnr());
        }
    }
}

class Dyr {
    String navn;
    String idnr;

    Dyr(String navn, String idnr) {
        this.navn = navn;
        this.idnr = idnr;
    }

    String getIDnr() {
        return idnr;
    }
}
```

```
KJØREEKSEMPEL:
Albert 1234
Mathilde 3456
Shaun 2345
```

Oppgave 2 [Nøkkeloppg.]

Arrays vs. HashMap

a) Følgende program viser et enkelt banksystem med en array kontoer[], og metoder for å finne en konto vha. navn til eieren og vha. kontonummer. Skriv om programmet slik at det bruker en HashMap i stedet for arrayen kontoer[]. I første omgang lager vi en HashMap, med personnavn som nøkkel og et Konto-objekt som verdi, deklarert slik:

```
HashMap<String, Konto> kontoFraNavn = new HashMap<String, Konto>();
```

Hvilke fordeler og ulemper får vi av å bruke HashMap her? Hva kan variabelen antKontoer erstattes med i programmet? (Anta foreløpig at personnavnene er unike og at hver person bare kan ha én konto i banken.)

```
class Konto {
    int nr; // kontonummer
    String navn; // eier
    int saldo;

    Konto(int nr, String navn, int saldo) {
        this.nr = nr;
        this.navn = navn;
```

```
        this.saldo = saldo;
    }

    void settInn(int innskudd) {
        saldo = saldo + innskudd;
    }
}

class Bank {
    Konto[] kontoer = new Konto[1000];
    int antKontoer = 0;

    public static void main(String[] args) {
        Bank b = new Bank();
    }

    Bank() {
        åpneNyttKonto(530010, "Nils", 4000);
        åpneNyttKonto(720020, "Elin", 8000);
        åpneNyttKonto(910030, "Tina", 9000);

        Konto k = finnKontoFraNavn("Elin");
        System.out.println("Elins kontonr: " + k.nr + ", saldo: " + k.saldo);

        k = finnKontoFraNr(530010);
        System.out.println("Kontonr. " + k.nr + " tilhører " + k.navn);
    }

    void åpneNyttKonto(int nr, String navn, int saldo) {
        Konto k = new Konto(nr, navn, saldo);
        kontoer[antKontoer] = k;
        antKontoer++;
    }

    Konto finnKontoFraNavn(String navn) {
        for (int i = 0; i < antKontoer; i++) {
            if (kontoer[i].navn.equals(navn)) {
                return kontoer[i];
            }
        }
        return null;
    }

    Konto finnKontoFraNr(int kontonr) {
        for (int i = 0; i < antKontoer; i++) {
            if (kontoer[i].nr == kontonr) {
                return kontoer[i];
            }
        }
        return null;
    }
}
```

KJØREEKSEMPEL:

```
Elins kontonr: 720020, saldo: 8000
Kontonr. 530010 tilhører Nils
```

Løsningsforslag

```
import java.util.*;

class Konto {
    int nr; // kontonummer
    String navn; // eier
    int saldo;

    Konto(int nr, String navn, int saldo) {
        this.nr = nr;
        this.navn = navn;
        this.saldo = saldo;
    }

    void settInn(int innskudd) {
        saldo = saldo + innskudd;
    }
}

class Bank2 {
    HashMap <String, Konto> kontoFraNavn = new HashMap <String, Konto>();
    HashMap <String, Konto> kontoer = new HashMap <String, Konto>();

    public static void main(String[] args) {
        Bank2 b = new Bank2();
    }

    Bank2() {
        åpneNyttKonto(530010, "Nils", 4000);
        åpneNyttKonto(720020, "Elin", 8000);
        åpneNyttKonto(910030, "Tina", 9000);

        Konto k = finnKontoFraNavn("Elin");
        System.out.println("Elins kontonr: " + k.nr + ", saldo: " + k.saldo);

        k = finnKontoFraNr(530010);
        System.out.println("Kontonr. " + k.nr + " tilhører " + k.navn);

        System.out.println("Antall kontoer: " + kontoer.size());
        avsluttKonto(k);
        System.out.println("Fins kontoen nå?: " + kontoer.containsValue(k));
        System.out.println("Antall kontoer nå: " + kontoer.size());
    }

    void åpneNyttKonto(int nr, String navn, int saldo) {
        Konto k = new Konto(nr, navn, saldo);
        kontoFraNavn.put(navn, k);
        kontoer.put("" + nr, k);
    }
```

```
    Konto finnKontoFraNavn(String navn) {
        return kontoFraNavn.get(navn);
    }

    Konto finnKontoFraNr(int kontonr) {
        return kontoer.get("" + kontonr);
    }

    void avsluttKonto(Konto k) {
        kontoFraNavn.remove("" + k.nr);
        kontoer.remove("" + k.nr);
    }
}

KJØREEKSEMPEL:
Elins kontonr: 720020, saldo: 8000
Kontonr. 530010 tilhører Nils
Antall kontoer: 3
Fins kontoen nå?: false
Antall kontoer nå: 2
```

b) Lag en HashMap til, kalt kontoer, hvor du bruker som nøkkel kontonummeret konvertert til String, og fortsatt Konto-objektene som verdi. Vis at metoden finnKontoFraNr() blir enklere nå. Videre tenk deg at vi skal ha en metode for å fjerne en konto. Følgende kode viser hvordan det kan gjøres med arrayer. Hvor mange programsetninger trengs det når vi bruker én HashMap i stedet? Og med to?

```
    void avsluttKonto(Konto k) {
        // Fjerner en konto ved å finne indeksen til kontoen i arrayen
        // kontoer[] og flytte alle kontoene med høyere indeks en plass ned.
        boolean funnet = false;
        for (int i = 0; i < antKontoer && !funnet; i++) {
            if (kontoer[i] == k) {
                funnet = true;
                for (int j = i; j < antKontoer - 1; j++) {
                    kontoer[j] = kontoer[j + 1];
                }
                antKontoer--;
            }
        }
    }
```

Løsningsforslag

Se del a).

c) Disse oppgavene har begrensningen at personnavnene må være unike og at hver person bare kan ha én konto i banken. Hvordan ville man unngått disse begrensninger i et mer avansert system? Hvilke fordeler og ulemper ser du av å bruke HashMap-er i stedet for 2D-arryaer i Oblig 3? (foreslå mulige nøkkel/verdi-kombinasjoner).

Hint: Se avsnitt 9.10 på side 191 i læreboka for forskjellene mellom arrayer og HashMap-er.

Løsningsforlag

I store systemer passer det bedre å ha personnummer som nøkkel eller noe annet som er unikt for hver verdi, i stedet for navn alene. Verdi kan være et Person-objekt, som i sin tur kan ha en objektvariabel av type HashMap som refererer til alle kontoene for personen. Hvis programmet trenger å finne personer ut fra navn så kan man likevel bruke navn som nøkkel i en tilleggs-HashMap, som f.eks. kan ha som verdi en HashMap over personene med det navnet.

Arrayer passer best å bruke når maks. antall elementer er kjent på forhånd, når vi trenger en gitt rekkefølge på dataene, og når indeksene er numeriske. **HashMap** er bedre når maks. antall elementer er ukjent, eller man trenger å slå opp i dataene vha. "indeksersom er noe annet enn tall (f.eks. tekster), og når rekkefølgen på dataene ikke er avgjørende. Med HashMap må dataene være objekter (pekere), ikke primitive typer, men det finnes objekt-typer for alle de primitive: Integer, Double, Character, Boolean.

I **Oblig** 3 er det mest naturlig å bruke arrayer, men man kan også bruke HashMap: da kan f.eks. hele hybelnavnet være nøkkel. Man trenger likevel lett tilgang til etasje- og rom-nr. så disse kan evt. legges til som objektvariabler i Hybel. En annen fremgangsmåte er å ha Etasje som egen klasse og da kan denne ha en array (eller HashMap) med hyblene i etasjen.

Oppgave 4 [Nøkkeloppg.]

HashMap med akronymer

a) Et akronym er et initialord (forkortelse) som blir lest og uttalt som et vanlig ord, altså ikke bokstav for bokstav.Lag et program som leser inn en rekke akronymer og deres tolkning fra fil (du finner en fil med akronymer (og andre forkortelser) på bokas hjemmeside: akronymer.txt). Legg akronymene og tolkningene i en HashMap med akronymet som nøkkel. Brukeren skal oppgi et akronym til programmet og få tilbake en eller flere tolk- ninger av akronymet. Det kan maksimalt være 10 tolkninger til hvert akronym. Denne brukerdialogen skal gå i løkke til brukeren svarer "-".

Tips: Lag en klasse der objekter har følgende attributter: akronymet og en String-array med tolkninger, samt antall tolkninger. Pekere til objekter av denne klassen legges i HashMap med akronymet som nøkkel. Det kan være lurt å lage en metode i denne klassen som legger til en ny tolkning, og som holder orden på hvor mange tolkninger det er i øyeblikket.

```
API   Application Programming Interface
AV  Audio/Video
AV  Authenticity Verification
BASH  Bourne Again Shell [Unix]
EU  Europaunionen
FN  Forente nasjoner
... osv ...
```

Løsningsforslag

```
// Basert på løsningsforslaget i lærebokens hjemmeside.
/*
* Løsning på oppgave 2 i kapittel 9 : Akronymer
* Programmet legges i en fil med navnet : AkronymerMain.java
*
* Løsningen bygger på to klasser som begge fins i denne filen
* Programmet forutsetter at alle akronymene står med store bokstaver
* i datafilen, men brukeren trenger ikke taste de inn med store bokstaver.
*/

import java.util.*; // for å få med HashMap
import easyIO.*;    // for å få med fil- og brukerkommunikasjon

class AkronymerMain {

    // initialiserer akronym-registeret
    HashMap<String, Akronym> akronymer = new HashMap<String, Akronym>();

    public static void main(String[] args) {

        // Oppretter et objekt av denne samme klassen for å kunne kalle
        // objektmetoder fra denne klassemetoden vha. pekeren "am".
        AkronymerMain am = new AkronymerMain();

        // leser inn data fra fil og bygger opp register
        am.byggRegister();

        //går i dialog med bruker
        am.brukerdialog();
    }

    void byggRegister() {
        In fil = new In("akronymer.txt");
        while(fil.hasNext()) {

            // leser akronymet og tolkningen
            String akro = fil.inWord();
            String tolkning = fil.inLine().trim();

            // sjekker om akronymet er registrert tidligere
            if (akronymer.containsKey(akro)){

                // vi skal nå legge til en ny tolkning
                Akronym a = akronymer.get(akro);
                a.add(tolkning);
            } else {

                // vi lager et nytt Akronym-objekt
                Akronym ny = new Akronym(akro,tolkning);

                // vi legger det inn i akronym-registeret
                akronymer.put(akro, ny);
```

```
            }
        }
        fil.close();
    }

    void brukerdialog() {
        In tastatur = new In();
        String svar;
        do {
            System.out.print("Skriv inn et akronym (avslutt med -): ");
            svar = tastatur.inLine();
            svar = svar.trim();
            svar = svar.toUpperCase();
            if (akronymer.containsKey(svar)) {
               Akronym a = akronymer.get(svar);
               a.skrivUt();
            } else if (!svar.equals("-")){
               System.out.println(svar + " fins ikke i akronymregisteret.");
            }
        } while (!svar.equals("-"));
    }
}

class Akronym {
    int antall;
    String akronym;
    String[] tolkninger = new String[10];

    Akronym (String a, String t) {
        akronym = a;
        tolkninger[antall] = t;
        antall = 1;
    }

    void add(String t) {
        if (antall < 10) {
            tolkninger[antall] = t;
            antall++;
        } else {
            System.out.println("For mange tolkninger av " + akronym);
        }
    }

    void skrivUt() {
       System.out.println(akronym + " kan bety:");
       for (int i = 0; i < antall; i++) {
           System.out.println("   " + tolkninger[i]);
        }
    }
}
```

Oppgave 5 [Nøkkeloppg.]

Uke 3: Array med tall

a) Lag et program med en for-løkke som ber bruker taste inn fem heltall og lagrer disse i en array kalt tall:

```
int[] tall = new int[5];
```

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.

Løsningsforslag

```
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
```

Oppgave 6

Metoder (10 poeng) oppgave 5 fra eksamen h09

a) Nedenfor er det deklarert en metode sum(int[] array), som skal beregne summen av tallene i en int-array. Vil metoden kompilere? Hvis nei, forklar hva som må endres for at metoden skal kompilere.

```
public void sum ( int[] array ) {
    int sum = 0;
    for (int i=0; i < array.length: i++) {
        sum += array { i };
    }
    return sum;
}
```

Løsningsforslag

Tre syntaksfeil:

- I linje 3 brukes ':' istedenfor ';' etter betingelsen i for-løkka.

- I linje 4 brukes ''/'' i stedet for '['/']' om paranteser for å angi plass i arrayen.
- I linje 6 returneres en int-verdi, selv om metoden er deklarert void. Endre metodens returtype til int.

b) Deklarerer en metode `snitt(int[] array)`, som skal beregne gjennomsnittet av tallene i parameteren array av type int-array. Snittet skal beregnes med flyttallsdivisjon og returverdien skal være av type double. Du skal skrive hele metoden, inkludert signaturen.

Løsningsforslag

```
double snitt (int[] array) {
    double snitt = 0.0;
    for (int i=0; i < array.length; i++) {
        snitt += array[i];
    }
    return snitt / array.length ;
}
```

c) Vi kan beregne en gjenstands snitthastighet ved å dividere tilbakelangt distanse på tid. Deklarerer en metode som tar to double-parametre distanse og tid, og som returnerer hastigheten som en double-verdi. Hvis verdien til tid er 0, skal metoden returnere -1. Du skal skrive hele metoden, inkludert signaturen. Finn på et passende metodenavn.

Løsningsforslag

```
double snittHastighet(double distanse, double tid) {
    if (tid == 0) {
        return -1;
    } else {
        return distanse / tid ;
    }
}
```

d) I sjø- og luftfart oppgis ofte hastigheter i knop. Én knop er definert som én nautisk mil pr. time, som tilsvarer 1852 meter pr. time. Deklarer en metode som tar en double-parameter knop og returnerer den tilsvarende hastigheten i km/t som en double-verdi. Du skal skrive hele metoden, inkludert signaturen. Finn på et passende metodenavn.

Løsningsforslag

```
double beregnKmT(double knop) {
    return knop * 1.852;
}
```

Oppgave 7

Klasser og objekter, Oppgave 11 eksamen h05 (25 poeng)

I programmet nedenfor skal du lage en konstruktør til klassen Pyramide som har pyramidens bredde, lengde og høyde som tre double-parametreparameter. Du skal også lage en objektmetode i klassen Pyramide som regner ut volumet og returnerer denne verdien (du skal bruke formelen: volum = 0.333*høyde*bredde*lengde). Du skal også skrive programkode i main som oppretter to pyramider, en med lengde =30.1, bredde=30.1 og høyde = 22.9 og en med samme grunnflate, men med dobbelt så stor høyde som den første pyramiden. Fra main skal du så kalle på volumberegningsmetoden i hvert av de to objektene og skrive ut en linje for hver pyramide med høyde, lengde og bredde samt volumet.

Løsningsforslag

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

        // skriv kode her som lager to Pyramide-objekter og
        // skriver ut deres høyde, lengde og volum
        Pyramide[] p = new Pyramide[2];
        p[0] = new Pyramide(30.1, 30.1, 22.9);
        System.out.println("Pyramide " + 30.1 + " x " + 30.1+ " x " + 22.9 +
                           "har volum " + p[0].volum());
        p[1] = new Pyramide(30.1, 30.1, 2*22.9);
        System.out.println("Pyramide " + 30.1 + " x " + 30.1 + " x " + 2*22.9 +

    } // end main
} // end class PyramideTest

class Pyramide {
    double hoyde, lengde, bredde;

    // skriv konstruktør her
    Pyramide(double lengde, double bredde, double hoyde){
        this.lengde = lengde;
        this.bredde = bredde;
        this.hoyde = hoyde;
    }

    // skriv objektmetode her som beregner og returnerer volumet
    double volum(){
        return 0.333 * hoyde * bredde * lengde;
    }
} // end class Pyramide
```

Hvis du har spørsmål eller kommentarer til dette løsningsforslaget, send en mail til *tuvakt@ulrik.uio.no*.