INF1000 Ekstratilbud

Stoff fra uke 1-3
12. September 2012
Siri Moe Jensen

Litt om opplegget

- Målgruppe: De som mangler forståelse for konseptene gjennomgått så langt. Trening får du ved å jobbe med oppgaver, alene eller med andre.
- Mål: Bygge grunnmur! Kommer noen uker nå med mye nytt stoff – så roer det seg.
- *Dere* må jobbe! I plenum, i par og alene
- Vi er mange – dele maskin, ut av rommet i pausen
- Fokus på konsepter og prinsipper – ikke finurligheter, detaljer eller "add-ons"
- Tre hovedbolker:
  - Dataprogrammer, variable og uttrykk
  - Forgreninger og while-løkker
  - For-løkker og arrayer, hvordan "skaper" vi et program?

Hva er poenget med et dataprogram?

Vi vil at maskinen skal gjøre en jobb for oss!

Hva skjer når vi kjører det?

Instruksjonene blir utført

Vil vi alltid at akkurat det samme skal skje?

Sjelden!

Hva vet vi/ vet vi ikke når vi skriver programmet, men først når det kjøres?

Eksempler

- Program som alltid gjør det samme?
  - Siri er flink – ta deg en kopp kaffe!
- Variasjoner?
  - Siri/ Mathias/ Stian/.. er flink!
- Hva avhenger variasjonene av?
  - Bruker (eller data fra fil, måleinstrument, ..)
  - Må representeres i programmet
  - Må hentes inn av programmet og/ eller beregnes

Variable, uttrykk

- Hva slags variasjoner er det snakk om?
  - Alt mulig! Eks:
  - Sovet godt i natt? (på en skala fra 1 til 5)
  - Gjenta (6 ÷ poeng) ganger

Oppgave: Hva skjer – på skjermen, i hukommelsen? Tegn!

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

            int tall1, tall2;

            tall1 = 5;
            tall2 = 10;

            tall1 = tall2;
            tall2 = tall1;

            int x = 2;
            x = tall1 * x;

            System.out.println ("tall1=" + tall1 + "  tall2=" + tall2 + "  x=" + x);
   }
}
```

Skriv et slikt program (bruk variablene):

Java-setning	Hukommelsen			Skjerm
1	t1 3			$java dittprog
2	t1 3	t2		
3	t1 3	t2	t3 7	
4	t1 3	t2 10	t3 7	$java dittprog
5	t1 3	t2 10	t3 7	3 10 7
6 *	t1 3	t2 10	t3 7	20

Hva er et uttrykk?

- ”noe” som leverer en *verdi* av en bestemt *type*
  - *int*
  - *double*
  - *boolean*
  - …
- Denne verdien kan skrives ut, eller tilordnes en variabel, eller brukes i et nytt utrykk.
- Hvis det er logisk verdi (boolean) kan den brukes til å ta en beslutning (i en if-test eller while-løkke)

Eksempler på uttrykk

operatorer

presedens

konvertering

- Heltall (int):
  - -5
  - tall1
  - (tall1 + 5) *12
- Desimaltall (for eksempel double)
  - 5.3
  - 17.999999999 / 2
  - vekt (deklarert som **`double vekt;`**)
- Logiske (boolean)
  - true
  - (true || false)
  - ferdig (deklarert som **`boolean ferdig;`**)
  - (tall1 > 5) && (tall2 == 10)

Oppgaver

For enkelt eller ledig tid? Løs ukeoppgaver

- Program 1
  1. Deklarer heltall **a** og **b**, initialiser med verdi 2 og 5
  2. Deklarer boolsk variabel **test** initialisert til **true**
  3. Gi **test** verdien av det boolske uttrykket (**a** er større enn **b**)
  4. Gi **test** den motsatte verdien av det den har
  5. Skriv ut verdien av **test**
  6. Øk variabelen **a** med verdien av variabel **b**
  7. Skriv ut verdien av **a** og **b**
  8. Skriv ut summen av **a** og **b**
- Program 2
  - Skriv et program som leser inn to heltall, og gir en boolsk variabel verdien **true** om de er like, **false** om de er forskjellige. Skriv ut verdien av alle variable.

Programflyt

Forgreninger (skal - skal ikke?) og while-løkker

Forgreninger og løkker

- Når vi trenger å bestemme hva som skal gjøres mens programmet kjører
- I stedet for at beslutningen tas når vi skriver programmet (legg inn den programsetningen eller la være, gjenta den 1 eller 3 eller 500 ganger), så lar vi programmet bestemme det under kjøring.
- Da må vi i stedet angi hvordan beslutningen skal tas – dette gjøres alltid ved hjelp av et logisk uttrykk, som evalueres under kjøring

Noen tester

Når utføres blokken?

- Eksempler på logiske uttrykk i if-tester:

```
if (true) {
        System.out.print
}
```

```
if (!false) {
        System.out.println
}
```

```
if (0 != 0) {
        System.out.println
}
```

```
int i = 5;
if ((i < 10) && (i > 10)) {
        System.out.println
}
```

FORGRENINGER: IF - ELSE

- Når man ønsker alternative setninger utført hvis betingelsen er usann brukes konstruksjonen *if-else*

```
int formue = 20;

if (formue > 100000000) {
    System.out.println (”En formue! ");
} else if (formue > 1000000) {
    System.out.println (”En voksen bankkonto!");
} else {
   System.out.println (”Ingenting å kjøpe husvære for!");
}
```

- Slike if-else-setninger kan kobles i kjede med flere alternative utfall

OPPGAVER (TIPS: LAG NY FIL OG NYTT KLASSENAVN FOR HVERT OPPGAVENR)

- Program 1b
  - Start med Program 1 fra tidligere. Skriv om programmet slik at det kun utfører punkt 7 (skriver ut verdien av **a** og **b**) dersom **a** og **b** er forskjellige. Hvis **a** og **b** har samme verdi skal programmet i stedet utføre punkt 8 (skrive ut summen).
- Program 3
  - Skriv et program som leser inn et heltall og skriver ut verdien som en tekst på skjermen hvis den er mellom 0 og 3. Bruk **`if-else`** eller **`switch`** (se s.79 i RpJ). Hvis verdien er negativ eller større en 3 skrives det ut en feilmelding.

For enkelt eller ledig tid? Løs ukeoppgaver

LØKKER

- Ofte ønsker vi å utføre en eller flere programsetninger mer enn én gang
- Dette gjøres ved hjelp av *løkker*
- Hovedsakelig to typer behov:
  - Utfør setningen(e) inntil en betingelse oppfylles -> while-løkke
  - Utfør setningen(e) et bestemt antall ganger –> for-løkke

WHILE-LØKKER

- Vi kan utføre en blokk med setninger flere ganger ved hjelp av en while-løkke

```
while (<logisk uttrykk>) {
    <setning 1;>
    <setning 2;>
    .....
    <setning n;>
}
```

- Hvis det logiske uttrykket er true, utføres blokken i while-løkka
- Når blokken som hører til løkka er utført, vil programmet returnere til toppen og evaluere det logiske uttrykket på nytt
- Hvis det logiske uttrykket er false, fortsetter utførelsen etter blokken uten at setningene inni utføres.

Eksempel

```
class SkrivLinjer {
    public static void main (String [] args) {
        int k = 1;
        while (k <= 5) {
            System.out.println(
                "Nå har k verdien " + k);
            k = k + 1;
        }
        System.out.println("Nå er k lik " + k);
    }
}
```

Hva skjer her?
Følg verdien av k,
og utfør testen for hvert gjennomløp av løkka

Kjøring


```
> java SkrivLinjer
Nå har k verdien 1
Nå har k verdien 2
Nå har k verdien 3
Nå har k verdien 4
Nå har k verdien 5
Nå er k lik 6
>
```

Evig løkke

- Dersom testen i while-løkka **aldri blir usann** (false), vil utførelsen av while-løkka aldri stoppe. Dette kalles en evig løkke.
- To eksempler:

```
class EvigLokkeOpplagt {
    public static void main (String [] args) {
        while (true) {
            System.out.println("INF 1000");
        }
    }
}
```

```
class EvigLokkeIkkeSaOpplagt {
    public static void main (String [] args) {
        int i = 1, j = 2;
        while (i < j) {
            System.out.println("Nå er i < j (i=" +
                i + ", j=" + j + ")");
            i++; j++;
        }
    }
}
```


Evig løkke - Kjøring

Den kan stoppes med **Ctrl+C**


```
> java EvigLokkeOpplagt
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
INF 1000
. . .
. . .
```

```
> java EvigLokkeIkkeSaOpplagt

. . .
. . .

Nå er i < j (i=82155, j=82156)
Nå er i < j (i=82156, j=82157)
Nå er i < j (i=82157, j=82158)
Nå er i < j (i=82158, j=82159)
Nå er i < j (i=82159, j=82160)
Nå er i < j (i=82160, j=82161)
Nå er i < j (i=82161, j=82162)
Nå er i < j (i=82162, j=82163)
Nå er i < j (i=82163, j=82164)

. . .
```

OPPGAVER

- Program 3b:
  - Skriv om starten av Program3 slik at det spør brukeren om et nytt heltall inntil tallet er positivt og mindre enn 4, før det skriver ut noe.
- Program 3c:
  - Skriv om Program 3b slik at det i tillegg til tall mellom 0 og 3 aksepterer (det vil si slutter å spørre om nytt tall) verdien 100.

For enkelt eller ledig tid? Løs ukeoppgaver

21

ARRAYER OG FOR-LØKKER

FOR-LØKKER

- En annen måte å få utført en instruksjon (eller blokk) mange ganger er ved hjelp av en **for-løkke**
- Innebærer bruk av en variabel som teller (indeks)
- (begynn å telle på **; gå ut av løkken når **; øk telleren med ** for hver gjennomføring)
- Brukes for blokker som skal utføres et bestemt antall ganger (kan ligge i en variabel)
- Gir oss tilgang på en "teller" (indeks)

EKSEMPEL PÅ FOR-LØKKE

```
class ForLokkeHvordan {
    public static void main (String [] args) {
        for( int i=0; i<5; i++) {
            System.out.println("Nå er i= " + i);
        }
}
```

Initialisering

Betingelse (logisk uttrykk)

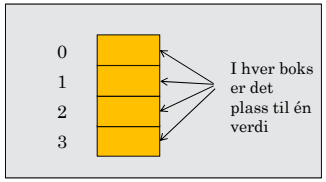
Oppdatering

```
> java ForLokkeHvordan
Nå er i= 0
Nå er i= 1
Nå er i= 2
Nå er i= 3
Nå er i= 4
>
```

24

ARRAYER

- En variabel holder kun én verdi
- En *array* kan holde mange verdier av samme type
- Verdiene i en array med lengde N får hver sin indeks (=posisjon) fra 0 til N-1
- En array med lengde 4 kan tegnes slik:



ARRAY-OBJEKT OG ARRAY-REFERANSE (PEKER)

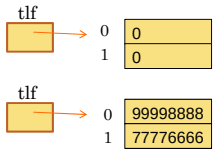
- Selv arrayen vi tegnet kalles et arrayobjekt. For å få tak i dette må vi ha en arrayreferanse (arraypeker).
- Deklarasjon av en arraypeker:
  ```
  int [] heltall;
  ```
- Opprettelse av arrayobjekt med plass til 5 heltall:
  ```
  heltall = new int[5];//lengden kan gis av en variabel
  ```
- Deklarasjon av peker og opprettelse av array samtidig:
  ```
  int [] heltall = {7, 10, 6, 0, 3};   //kjenner verdiene
  ```
  eller
  ```
  int [] heltall = new int[5]; //fyller inn verdiene senere
  ```

BRUK AV ARRAYEN

- Tilordning:

```
int [] tlf = new int [2];

tlf [0] = 99998888;
tlf [1] = 77776666;
```



- Lesing:

```
int mittTlf = tlf [1];
```

mittTlf
77776666

- Få tak i lengden på arrayen:

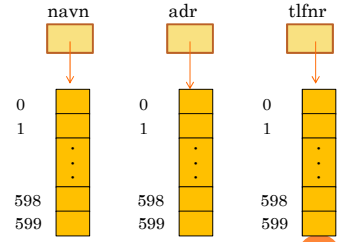
```
int antTlf = tlf.length;
```

antTlf
2

EKSEMPEL - ARRAYER

- Anta at vi ønsker å lagre navn, adresse og telefonnummer for de som følger et bestemt kurs med maksimalt 600 studenter

```
String [] navn = new String [600];
String [] adr = new String [600];
int [] tlfnr = new int [600];
```



OPPGAVER

For enkelt eller ledig tid? Løs ukeoppgaver

- Program 5
  - Skriv et program med en for-løkke som teller seg fra 10 ned til 1, og skriver ut hvert tall på skjermen.
- Program 6
  - Skriv et program som inneholder deklarasjonene nedenfor, og fyller ut med test-verdier for student 0 og 3. Deretter skal programmet skrive ut navn, adresse og telefonnummer for hver student (en linje per student).
  - Hva blir skrevet ut for de studentene du ikke har gitt verdier?

```
String [] navn = new String [6];
String [] adr = new String [6];
int [] tlfnr = new int [6];
```