TDDC77 Objektorienterad Programmering

Föreläsning 2

Sahand Sadjadee
IDA, Linköpings Universitet

Outline

- Operatorer
- Java Standard Library
- Inmatning

peratorer

LINKÖPINGS UNIVERSITET

Operatorer

- En operator är en symbol som gör en eller flera **operationer**.
- En operator kan ha en, två eller tre **operander**.
- En operand kan vara en **variabel** eller en **literal**.
- Till exempel: `3 + 7, 2 * height //height är en variabel`

LINKÖPINGS UNIVERSITET

operatorer

- Det finns 7 sorters operatorer i Java:
  - **Simple Assignment Operator**
  - **Arithmetic Operators**
  - **Unary Operators**
  - **Equality and Relational Operators**
  - **Conditional Operators**
  - **Type Comparison Operator (ska tas upp senare i kursen)**
  - **Bitwise and Bit Shift Operators (ingår inte i kursen)**

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/opsummary.html

Single Assignment Operator

- Tilldelar ett värde, variabel/literal, på sin högersida till en variabel som ligger på sin vänstersida.
- Har två operander som gör den en binär operator.
- association är från höger till vänster. Det betyder att om det finns flera = i samma instruktion, körs den som ligger på högersidan först.
- Exempel:
  - `int a;`
  - `a = 7;`
  - `int b = a = 5;`

LINKÖPINGS UNIVERSITET

Uttryck - expressions

- Varje operator med sina operander skapar ett uttryck.
- Uttrycket ersätts av ett värde efter operationen är genomförd.
- I Single Assignment Operatorns fall ersätts uttrycket av värdet som tilldelas till variabeln på vänstersidan.
- Exempel: `int b; int a = b = 7;`
- I ovanstående exempel är `b = 7` ett uttryck som ersätts av `7`.
- I resultat körs `int a = 7;` i nästa steg.

LINKÖPINGS UNIVERSITET

Arithmetic operators

- + Additive operator (also used for String concatenation)
- - Subtraction operator
- * Multiplication operator
- / Division operator
- % Remainder operator

Exempel:

```
int a = 7;
int b = 8;
int c = a + b;
```

Arithmetic expressions

Används med heltals och flyttalstyper:

- Addition +, Subtraktion , Multiplikation , Division =, Resten %. t.ex. 18 % 5 ger 3
- Om en eller båda operanderna är flyttal, resultatet blir också ett flyttal; t.ex. 7.5 / 3 ger 2.5
- Om en eller båda operanderna är flyttal, resultatet blir också ett flyttal; t.ex. 7.5 / 3.0 ger 2.5
- Operanden med största typen, storleksmässigt, bestämmer typen på resultatet mellan flyttalstyperna.
- När det kommer till heltal, bestäms den närmaste typen, storleksmässigt, som kan lagra resultatet.

Unary operators

- − Unary minus operator; negates a value
  - Kommer innan ett heltal/flyttalsvärde.
- ++ Increment operator; increments a value by 1
  - Kommer innan eller efter ett heltal/flyttalsvariabel.
- -- Decrement operator; decrements a value by 1
  - Kommer innan eller efter ett heltal/flyttalsvariabel.
- ! Logical complement operator; inverts the value of a boolean
  - Kommer innan ett booleskt värde.

Prioritet

Alla unära operatorer står här.

Prioritet	Operator	associerar
1	unärt: +, -	h → v
2	binärt: *, /, %	v → h
3	binärt: +, -, sträng sammanbindning	v → h

```
r = 77 - 33 * 2
```

Använd parenteser för att tvinga prioritet!

LINKÖPINGS UNIVERSITET

Inkrementera och kombinera tilldelning med en operator

Flera kombinationer är möjliga i Java:

```
result = 0;
result++;
result+= 15 - 9 / 3;
result*= 2;
result--;
System.out.println(‘‘result= ‘‘ + result);
```

LINKÖPINGS UNIVERSITET

Equality and Relational Operators

Operator	Förklaring
==	lika med
!=	olika från
<	mindre än
<=	mindre än eller lika med
>	större än
>=	större än eller lika med

```
boolean resultat;
int tal1=0, tal2=1;

resultat = (tal1 < tal2);  // true
resultat = (tal1 <= tal2); // true
resultat = (tal1 > tal2);  // false
resultat = (tal1 >= tal2); // false
resultat = (tal1 == tal2); // false
resultat = (tal1 != tal2); // true
```

Equality and Relational Operators

- Har två operander(binär).
- Operanderna får inte vara av typen boolean.
- Association: från vänster till höger.
- Uttrycket är ALLTID av typen boolean.

Conditional Operators

Operator	Förklaring
!p	negationen av p
p && q	sant omm p och q
p || q	sant omm p eller q

```
boolean res;

resultat = predikat && true;
resultat = false && predikat;
resultat = predikat && !predikat;

resultat = predikat || true;
resultat = predikat || false;
resultat = predikat || !predikat;
```

LINKÖPINGS UNIVERSITET

Conditional Operators

- Har två operander(binär).
- Operanderna får **BARA** vara av typen boolean.
- Association: från vänster till höger.
- Uttrycket är ALLTID av typen boolean.

Prioritet

Operators	Precedence
postfix	*expr*++ *expr*--
unary	++*expr* --*expr* +*expr* -*expr* ~ !
multiplicative	* / %
additive	+ -
relational	< > <= >= instanceof
equality	== !=
logical AND	&&
logical OR	||
assignment	= += -= *= /= %=

LINKÖPINGS UNIVERSITET

Standard Library

LINKÖPINGS UNIVERSITET

Java Standard Library

- Innehåller ett tusental klasser som gör olika uppgifter. Till exempel:
  - Grafik
  - Gränssnitt (GUI)
  - Nätverk
  - Matematiska beräkningar
  - Inmatning/utmatning
  - Felhantering
  - …

- Standardbiblioteket ingår i JDK.

JavaSE Javadoc

- Innehåller information om alla klasser som tillhör till Java Standard Bibliotek.
- Overview (Java SE 11 & JDK 11) - Oracle Docshttps://docs.oracle.com/en/java/javase/11/docs/api/index.html

Inmatning

LINKÖPINGS UNIVERSITET

Inmatning med Scanner

- `Scanner` används vid inmatning från tangentbord eller en fil
- Bryta ner inputen i en sekvens av lexikala element eller “tokens”

```java
/* Scan.java
 * Programmet demonstrera användandet av Scanner programm
 */
import java.util.Scanner;
class Scan
{
    // Metoden bryter ner en sträng i flera tokens och skriva ut de
    static public void main(String[] args){
     String source="1 TDDC 54 true 4.5 ett två tre!";
     Scanner scan = new Scanner (source);

     String token1 = scan.next();
     String token2 = scan.next();
     int token3 = scan.nextInt();
     boolean token4 = scan.nextBoolean();
     double token5 = scan.nextDouble();
     String token6 = scan.nextLine();

     System.out.println("token1 = " + token1);
     System.out.println("token2 = " + token2);
     System.out.println("token3 = " + token3);
     System.out.println("token4 = " + token4);
     System.out.println("token4 = " + token5);
     System.out.println("token4 = " + token6);
    }
}
```

Inmatning med Scanner

- `System.in` representerar “standard input stream”, här tangentbordet
- Kan också läsa in från en fil.

```java
/* ScanStandard.java
 * Programmet demonstrera hur man kan läsa in
 * från den standard input, ofta konsolen
 */
import java.util.Scanner;
class ScanStandard{
    /* Metoden läser in avstånd och tid från användaren,
     * räknar upp hastigheten och skriva ut den
     */
    static public void main(String[] args){
     Scanner scan = new Scanner (System.in);

     System.out.println("What was the distance in kms?");
     Double distance = scan.nextDouble();

     System.out.println("What time in minutes did it take?");
     Double time = scan.nextDouble();

     System.out.println("The average speed was: " + (distance*60/time) + "km/h");
    }
}
```

Tack för att du lyssnade!

LINKÖPINGS UNIVERSITET