

EDA016 Programmeringsteknik för D

Läsvecka 1: Introduktion

Björn Regnell

Datavetenskap, LTH

Lp1-2, HT 2015

1 Introduktion

- Om denna kurs
- Vad är programmering?
- Vi programmerar
- Grundläggande programkonstruktioner i Java
- Sammanfattning

Vad och hur?

■ *Vad* ska du lära dig?

- Grundläggande principer för programmering
- Konstruktion av enkla algoritmer
- Tänka i abstraktioner
- Imperativ och objektorienterad programmering
- Programspråket Java
- Utvecklingsmiljön Eclipse: implementera, testa, felsöka

■ *Hur* ska du lära dig?

- Genom praktiskt eget arbete: Lära genom att göra!
- Genom studier av kursens teori: Skapa förståelse
- Genom samarbete med dina kurskamrater

Kurslitteratur

- "Objektorienterad programmering och Java" av Per Holm
- Kurskompendium med övningar och laborationer
- Bokpaket säljs på KFS
John Ericssons väg 4
<http://www.kfsab.se/>



Personal

Kursansvarig:

Björn Regnell, bjorn.regnell@cs.lth.se

Kurssekreterare:

Lena Ohlsson

Exp.tid 09.30 - 11.30 samt 12.45 - 13.30

Handledare:

Gustav Cedersjö, Doktorand

Maj Stenmark, Doktorand

Anton Klarén, D09

Maria Priisalu , D11

Anders Buhl, D13

Erik Bjäreholt, D13

Fatima Abou Alpha, D13

Cecilia Lindskog, D14

Emma Asklund, D14

Kursmoment — varför?

- **Föreläsningar:** skapa översikt, ge struktur, förklara teori, svara på frågor, motivera varför
- **Övningar:** bearbeta teorin med avgränsade problem som mestadels löses med papper & penna, förberedelse inför laborationerna
- **Laborationer:** lösa programmeringsproblem praktiskt, obligatoriska uppgifter; lösningar redovisas för handledare
- **Resurstider:** få hjälp med övningar och laborationsförberedelser av handledare
- **Samarbetsgrupper:** grupplärande genom samarbete och dialog
- **Kontrollskrivning:** obligatorisk, diagnostisk, kamraträttad; kan ge samarbetsbonuspoäng till tentan
- **Inlämningsuppgift:** du visar att du kan skapa ett större program självständigt; redovisas för handledare
- **Tenta:** Skriftlig tentamen utan hjälpmedel, förutom **snabbpreferens**.

Nytt för i år

Årets kurs är i flera avseende väsentligt annorlunda and förra årets upplaga, så lita inte på allt som era äldre kursare säger :)

- Övningar blir resurstider i datorsal
- Inlämningsuppgift utan skriftlig rapport
- Samarbetskultur och grupplärande
- Nya övningar
- Nya laborationer
- Nya föreläsningar
- Höjda ambitioner: Fler ska klara tentan med högre betyg och fler ska lära sig mycket mer än vad som "råkar" komma på tentan...

Detta är bara början...

Exempel på efterföljande kurser som bygger vidare på denna:

■ Årskurs 1

- Programmeringsteknik – fördjupningskurs
- Utvärdering av programvarusystem
- Diskreta strukturer

■ Årskurs 2

- Objektorienterad modellering och design
- Programvaruutveckling i grupp
- Algoritmer, datastrukturer och komplexitet
- Funktionsprogrammering

Förkunskapsenkät

Samarbetsgrupper

En typisk kursvecka i första läsperioden

- Gå på föreläsningar måndag–tisdag
- Jobba själv med boken, övningar, labbförberedelser
- Träffas i samarbetsgruppen och hjälp varandra att förstå mer och fördjupa lärandet
- Gå till resurstider och få hjälp och tips av handledare, onsdag–torsdag
- Genomför obligatorisk laboration på fredagen

Se detaljerna och undantagen i [schemat i TimeEdit](#)

Resurstider

Laborationer

Att göra Vecka 1

- Läs kursboken kapitel 1-3.2
- Gör övning 1: Hello World, Hello Args, javac, editera–kompilera–exekvera–felsök, värden, uttryck variabler och tilldelning
- Gör förberedelserna till laboration 1
- Genomför laboration 1: Quiz — träna på att editera, läsa, ändra och felsöka i färdig programkod
- Funktionsprogrammering

Vad är programmering?

Programming unplugged



Att skapa koden som styr världen

Vad är en kompilator och ett programspråk?

Vad är Java?

Vad är objektorientering?

- Det finns många olika **programmeringsparadigm** (sätt att programmera på), till exempel:
 - **imperativ programmering**: programmet är uppbyggt av sekvenser av olika satser som påverkar systemets tillstånd
 - **objektorienterad programmering**: en sorts imperativ programmering där programmet består av objekt som samlar data och operationer på data
 - **funktionsprogrammering**: programmet är uppbyggt av samverkande (matematiska) funktioner som undviker föränderlig data och tillståndssändringar
 - **logikprogrammering**: programmet är uppbyggt av logiska uttryck som beskriver olika fakta eller villkor och exekveringen utgörs av en bevisprocedur som söker efter värden som uppfyller fakta och villkor

Grundläggande principer i imperativ programmering

- **Sekvens:** Ett program innehåller sekvenser av *satser*. Ordningen mellan dessa har helt avgörande betydelse.
- **Alternativ:** Systemet reagerar på vad som händer och kan välja olika vägar genom programmet beroende på *variablers* värde
Java: **if**-sats eller **switch**-sats
- **Repetition:** Göra saker om och om igen
Java: **while**-loop eller **for**-loop
- **Abstraktion:** Kapsla in (komplexa) programdelar och sätta namn på dessa så att de enkelt går att återanvända utan att vi behöver "rota i inandömet".
Java: klasser och metoder

Hello World!

Vårt första java-program:

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hej och välkomna!");  
    }  
}
```

Hello World! – Vad betyder egentligen allt detta?

```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hej och välkomna!");  
    }  
}
```

- **public** Denna programdel är synlig "utåt" och kan användas av andra delar.
- **class** Ett slags "kodbyggblock" som samlar olika programdelar. All java-kod måste finnas i en klass. Det finns tusentals färdiga klasser att använda direkt i Java och man kan lätt skapa egna klasser. Klammerpar { } anger början och slut.
- **static** Denna programdel skapas direkt vid programmets start och det finns exakt *en* sådan här per klass.
- **void** Berättar för kompilatorn att inget värde returneras från denna programdel.
- **main** Berättar var exekveringen av programmet börjar.
- **()** Parentespar berättar för kompilatorn att vi här kan ha parametrar.
- **String[] args** Möjliggör indata till programmet i form av flera textsträngar. Parametern args måste finnas i main, men vi använder den inte i detta program.
- **System.out.println** Den färdiga klassen System kan bl.a. skriva ut text. Textsträngar avgränsas av citationstecken. Semikolon avgränsar satser.

Översikt av några grundläggande programkonstruktioner i Java

- värden (values): data som programmet kan använda

42 "hej" 42.0 **true**

- uttryck (expressions)

41+1 "hej"+1 43.5-1.5 **!true**

Värden och uttryck

Code @ LTH