Högskolan Kristianstad
291 88 Kristianstad
044 250 30 00
www.hkr.se

Fakulteten för naturvetenskap
Andreas Nilsson

DA556, Programmering i Java

Lektion 1 - Introduktion

Introduktion

Denna lektion kommer att fokusera på att introducera kursen samt ge en kortare introduktion till programmeringsspråket Java och programmering i allmänhet.

Tanken är inte att försöka få in all information om kursen i denna lektionen utan snarare fokusera på ge en kort introduktion. Det är inte heller lärarens ambition att ge en heltäckande bild av Javas historia utan snarare en överblick. Överblicken är tänkt att ge studenten grundläggande förståelse om Javas utveckling över tiden.

Innan du fortsätter

Då mycket praktisk information kring kursen ges i Studiehandledningen[1] uppmanas alla studenter att läsa igenom den innan studenten påbörjar Lektion 1.

Varför programmering?

För några år sedan diskuterades det vid olika tillfällen i media om man borde införa programmering som ett ordinarie ämne i grundskolan, om detta vore en bra idé och hur det i så fall borde göras[2][3]. Det slutade den gången med att programmering inte blev något eget ämne utan att programmering skulle tas upp i andra ämnen, som exempelvis matematik och teknik. Oavsett om programmering ligger som eget ämne eller ej är det viktigt att unga får en förståelse för programmering då samhället blir mer och mer digitalt.

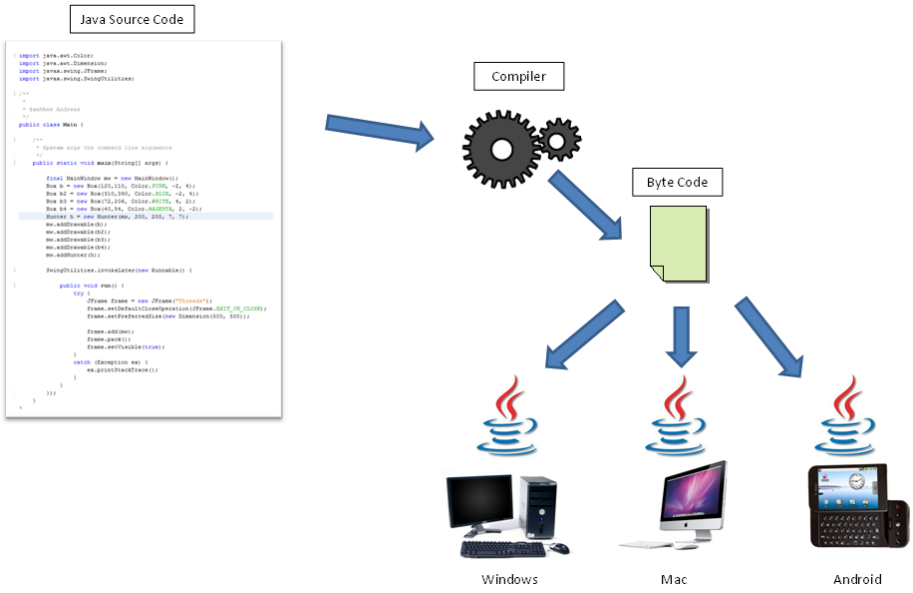
Det är dock inte bara unga som främjas av att lära sig att programmera, alla som är intresserade av hur datorer fungerar, hur program fungerar, eller har ambitionen att själva skapa program för datorer eller applikationer för mobiltelefoner vinner på att lära sig programmera. Genom att lära sig programmera får man även träning i logiskt tänkande som kan användas i många andra sammanhang.

Ett annat sätt att se på varför man ska lära sig att programmera är att titta på hur mycket i vår vardag som styrs av datorer eller hur mycket vi använder datorer i vår vardag. Ett sätt att mäta detta användande på är att titta på hur mycket vi använder Internet. En undersökning som gjordes av Svenskarna och Internet[4] visar att i stort sett alla svenskar, hela 94%, använder Internet och 9 av 10 använder Internet varje dag. När så mycket av det vi gör sker i den digitala världen blir det mer och mer viktigt att vi alla får en bättre förståelse för hur dessa datorer fungerar, eller varför de ibland inte fungerar. Ett sätt att få bättre förståelse för hur datorer och de program som körs på dem fungerar är att lära sig att programmera. Inte bara kan vi få bättre förståelse, vi kan kanske även skapa egna program som kan hjälpa både oss själva och andra i vardagen.

Vad är programmering?

Att programmera är i grund och botten att ge instruktioner till en dator. Dessa instruktioner måste ges på ett språk som datorn kan förstå och tolka. Dessa instruktioner översätts sen till körbar kod som datorn kan exekvera. Denna process kallas att man kompilerar sin kod. En schematisk bild av hur denna process kan tänkas se ut kan ses i Figur 1.

Figur 1 - Schematisk bild av programmeringsprocessen



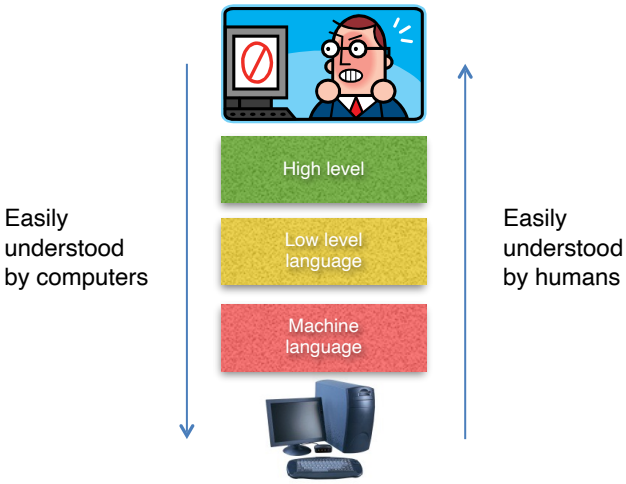
När man inom programmering talar om språk brukar man rätt snart stöta på begrepp såsom abstraktionsnivå, maskinspråk (maskinnära språk eller assembler), lågnivåspråk och högnivåspråk. Enkelt utryckt kan man säga att språk inom programmering är antingen lättare för datorn att förstå eller lättare för en människa att förstå. Ju lättare ett språk är att förstå för en människa desto högre abstraktionsnivå sägen man att språket har.

På den lägsta nivån förstår en dator bara ettor och nollor men då det skulle vara allt för omständligt att programmera med bara ettor och nollor som hjälpmedel har man utvecklat en uppsjö med språk som en programmerare kan ha till sin hjälp. De språk som är enklast för datorn att förstå kallas maskinspråk eller assembler. Detta är språk som historiskt har varit mycket viktiga för att kunna programmera datorer, idag har de dock tappat lite av sin dragningskraft då de tidvis inte är speciellt intuitiva att använda och kräver djup förståelse om hur datorer fungerar på en låg nivå.

Tar vi ett kliv upp i abstraktionsnivå hamnar vi på det som ofta kallas lågnivåspråk. Till dessa hör bland annat ett programmerinsspråket som heter C. Lågnivåspråk är mycket enklare för människor att förstå och gömmer lite av komplexiteten med maskinspråk genom att erbjuda färdiga kodblock som programmeraren kan anropa utan att känna till exakt hur dessa kodblock exekveras. Programmeraren behöver bara hålla koll på vilket kodblock som skall anropas när hen vill ha något speciellt utfört av datorn.

Högnivåspråk är det som är absolut vanligast idag och de absolut flesta programmerare spenderar sina dagar med att skriva kod i ett eller annat högnivåspråk. I denna kursen kommer högnivåspråket Java att användas.

Högnivåspråk har en väldigt hög abstraktionsnivå som döljer de underliggande instruktionerna för programmeraren som i stället använder sig av API (Application Programming Interface) för att få datorn att utföra det som programmeraren vill ha gjort. Fördelarna med högnivåspråk är många men framför allt gör de koden mycket mer lättläst och underlättar utveckling, vidareutveckling och felsökning enormt. Figur 2 ger en grafisk bild av hur programmeringsspråk ofta delas in i olika ”nivåer”.



Figur 2 - Abstraktionsnivåer på programmeringsspråk

Kursens lärare

Kursens lärare kommer att vara Andreas Nilsson, kontaktuppgifter till honom finns i Studiehandledningen[1] med det går också bra att nå honom via direktmeddelande i Canvas.

Det är också Andreas Nilsson som kommer att rätta samtliga inlämningsuppgifterna.

Kursens upplägg

I Studiehandledningen[1] finns bland annat information om vilka delprov kursen har och hur dessa examineras. Studiehandledningen har även ett förslag till generell studieplanering som kan ses som en riktlinje för var man som student bör befinna sig i kursen vid ett visst tillfälle. Utöver vad som nämns i Studiehandledningen kommer här några saker studenten bör fundera kring, ta ställning till eller mer generellt känna till.

1) Varje student bör ta fram en egen studieplanering som passar den enskilda studenten. Denna plan bör tas fram omgående för att på så vis lättare kunna anpassas till eventuella perioder när studenten inte kan ägna så mycket tid år kursen.

2) Studenten bör spendera tillräckligt med tid på kursen redan från början för att inte hamna i tidsnöd på slutet.

3) Var frågvis! Om det finns något studenten upplever som svårt så finns det troligtvis fler som upplever samma sak som svårt. Använd gärna forumet för att samarbeta, hjälpa varandra och glöm inte att genom att hjälpa någon annan förstärker man sina egna kunskaper. Tveka inte heller att kontakta läraren med frågor eller funderingar.

4) Studenten måste ta ansvar för sitt eget lärande, om något område intresserar studenten mer eller om något område upplevs som besvärligt, ta hjälp av boken, biblioteket, vänner eller Internet för att läsa mer inom det aktuella området. Tänk på att böcker kan upplevas som lättlästa av vissa och svårbegripliga av andra. Skulle studenten fastna rekommenderas det att läsa om samma sak i en annan bok. Utnyttja gärna biblioteket för att få tillgång till fler böcker.

5) Även om det är tillåtet att diskutera uppgifterna och hjälpa varandra ska studenten lösa samtliga uppgifter och inlämningsuppgifter individuellt. Det är inte tillåtet att kopiera lösningar från någon annan, detta kan leda till att studenten anmäls till disciplinnämnden.

6) Hela kursen ges via Internet. Kursens material kommer att publiceras på Canvas efter hand som kursen fortlöper. Skulle något material inte vara tillgängligt i tid ombeds studenten att ta kontakt med undervisande lärare omgående.

I övrigt är kursen byggd på antagandet att studenten tar eget ansvar för sina studier och hör av sig ifall problem uppstår, om något behöver förklaras ytterligare eller om studenten bara har några frågor. Åter igen, studenten uppmanas att aldrig tveka kring ifall det är rätt eller fel att höra av sig till läraren. Det är alltid rätt!

Java, en kort historik

Som nämnts tidigare kommer kursen att använda sig av programmeringsspråket Java som är ett högnivåspråk.

Även om det är svårt att mäta placerar sig Java ofta i topp när det gäller vilket programmeringsspråk som är populärast eller mest använt. Några av anledningarna till att Java är så populärt är att det är plattformsoberoende vilket innebär att man kan skriva ett program i Java som sen kan köras på flera olika plattformar så som exempelvis Windows, Linux, Unix och Mac. Detta är en stor fördel då det förkortar den tid det tar att utveckla program och det underlättar underhållet av program då det bara finns en kodbas att hålla reda på.

En annan anledning till att Java används så mycket är att man kan använda Java inom en rad områden. Java kan användas oavsett om ens program är tänkt att köras på en server, på en desktop dator, som en del i ett inbyggt system, eller på en mobil enhet.

Javas historia började redan 1991 när en herre vid namn James Gosling som jobbade på Sun Microsystems började jobba med att ta fram ett sätt att programmera hemelektronik. Hans arbetsgivare blev inte speciellt imponerade och Java kom aldrig riktigt till sin rätt i denna roll. Däremot föll bitarna på plats några år senare när James Gosling kom på att Java kunde användas för att köra applikationer via Internet. Nedan ges en mycket överskådlig inblick i Javas historia i form av en punktlista.

1991:	James Gosling på Sun Microsystems påbörjar utvecklingen av Java
1993 - 1994:	Sun kommer på att man kan använda Java till att köra program på Internet
1994:	Sun släpper den första webbläsaren som kan köra Java Applets, HotJava Browser
1995:	Sun släpper Java 1.0
2002:	Version 1.4 släpps. Många tillägg gjorde att detta blev en mycket omtyckt version
2004:	Version 5 släpps. Bland annat lägger man till Generics, Enum och Collections.
2006:	Java blir open source
2009:	Oracle Corporation köper Sun Microsystems
2011:	Version 7 släpps. JVM får support för dynamiska språk, nytt paket för filhantering

2014:	Version 8 släpps. Innehåller bland annat support för Lambda
2017:	Version 9 släpps. Introducerar moduler
2018:	Version 10 släpptes. Local-Variable Type Inference lades till
2018:	Version 11 släpptes. Ändringar för säkerhet och Garbage Collection
2019:	Version 12 och 13 släpptes. En ny typ av switch introduceras
2020:	Version 14 och 15 släpptes. Text Blocks lades till
2021:	Version 16 och 17 släpptes. Pattern matching och sealed classes lades till

Även om listan ovan är väldigt överskådlig kan man se att nya versioner har släppts efter hand som nya funktioner lagts till i språket. Sedan en tid tillbaka försöker man släppa nya versioner av Java lite oftare för att på så vis hålla språket modernt.

Förslag på vidare studier

För vidare studier kommer här några tips:

1) Udacity har publicerat en kort, väldigt översiktlig men bra video om vad programmering är på Youtube[5]. I videon nämner de programmeringsspråket Python men detta kan ersättas med Java som är det språk vi kommer att använda.
2) En lite längre introduktionsvideo om programmering kommer från Eli the Computer Guy[6]. Videon rekommenderas på grund av att den är välgjord och informativ. Speciellt de första tjugo minuterna och sammanfattningen är bra.
3) Lite mer information om Java som plattform kan man få genom att läsa igenom vad Oracle skriver i sin nu några år gamla introduktion till Javateknologien[7].
4) Även om Wikipedia inte alltid ses som en tillförlitlig källa kan jag rekommendera dess sida om Java[8] då den är både informativ och ger en trevlig överblick. För tillfället är de olika exempel som ges av mindre vikt, läs fram till att exemplen börjar.

Referenser

[1] Studiehandledning. (2022) Andreas Nilsson. [Online] Tillgänglig från: https://hkr.instructure.com (Kursen DA556B) [Tillgänglig: 2022-02-17]

[2] Sjätteklassare med programmering på schemat. (2015) Lars Anders Karlberg. [Online] Tillgänglig från: https://www.nyteknik.se/digitalisering/sjatteklassarna-med-programmering-pa-schemat-6394723
[Tillgänglig: 2022-02-17]

[3] Stockholms stad inför dataslöjd på skolschemat. (2015) Karin Wanngård, Maria Rankka, Martin Lorentzon. [Online] Tillgänglig från: https://www.dn.se/debatt/stockholms-stad-infor-dataslojd-pa-skolschemat/ (prenumeration)
[Tillgänglig: 2022-02-17]

[4] Svenskarna och Internet 2021. (2021) IIS. [Online] Tillgänglig från: https://svenskarnaochinternet.se/rapporter/ [Tillgänglig: 2022-02-17]

[5] What is programming - Intro to Computer Science. (2014) Udacity. [Online] Tillgänglig från: https://www.youtube.com/watch?v=tAR3ZAwKc78
[Tillgänglig: 2022-02-17]

[6] Introduction to Programming. (2011) Eli the Computer Guy. [Online] Tillgänglig från: https://www.youtube.com/watch?v=lJnvq0A_7WQ [Tillgänglig: 2022-02-17]

[7] Lesson: The Java Technology Phenomenon. (2015) Oracle. [Online] Tillgänglig från: https://docs.oracle.com/javase/tutorial/getStarted/intro/index.html
[Tillgänglig: 2022-02-17]

[8] Java (Programming Language). (2012) Community based. [Online] Tillgänglig från: https://en.wikipedia.org/wiki/Java_(programming_language)
[Tillgänglig: 2022-02-17]