Föreläsning 1 a)

Introduktion till IOOPM

Imperativ och objektorienterad programmeringsmetodik

- ML: funktionell programmering
  - Ett program är en samling funktioner (f, g, etc.) i en nästan matematiska betydelse
  - Funktionerna kombineras ($f(g)$, etc.) för att konstruera programmets betydelse
  - Programmeringen går åt det deklarativa hållet (vad skall beräknas, hur mellan raderna)

- Imperativ programmering
  - Ett program är en sekvens instruktioner som utförs i ordning och som (i regel) har effekter på ett gemensamt minne
  - Instruktionerna kapslas i regel in i funktioner/procedurer men dessa är mer byggstenar än matematiska objekt
  - Instruktioner "kommunicerar" med hjälp av det gemensamma minnet – om man byter ordning på två instruktioner i sekvens kan programmet få ett helt annat beteende
  - Programmeringen är i termer av hur något skall beräknas, vad är mellan raderna

- Objektorienterad programmering
  - Det vanligaste programmeringsparadigmet sedan ca 20 år
  - Ett program är en samling objekt som kommunicerer med varandra genom att skicka meddelanden
  - Hur ett meddelande skall tolkas bestäms av objektet som mottar meddelandet

- Programmeringsmetodik
  - En metodik är ett system av processer/procedurer som används inom ett visst område
  - Med programmeringsmetodik avser vi på denna kurs användande av verktyg och tekniker för högkvalitativ mjukvaruutveckling, ex.:
    - Testning och testdriven utveckling
    - Tekniker för att debugga kod
    - Utvecklingsmetoder och -filosofier
    - Hur man skriver läsbar kod
  - En metodik är ofärdig – det är något att utgå ifrån, inte något färdigt

Kursen IOOPM 2012

– Upplägg: *uppgiftsdrivet*

– 6 inlämningsuppgifter & 7 laborationsuppgifter

– 2 tentor

– 1 projektarbete

Kursen IOOPM 2013

- Nytt upplägg: *måldrivet*
  * Tydliga mål som skall uppfyllas, valbara uppgifter med vars hjälp man kan uppfylla målen
  * Du kan räkna ut vad som är kvar och vilka betyg du kan få
  * Du måste själv ta ansvar för vilka mål du vill uppfylla, i vilken ordning, och hur du skall demonstrera detta
  * Minst 45 mål (för betyg 3)
- Totalt 15 olika uppgifter att *välja* mellan (plus en obligatorisk)
- 1 frivillig tentamen för högre betyg
- 1 kodprov i slutet av terminen
- 1 projektarbete

Koreografi

– Kursen indelad i tre *faser*, indelade i *sprintar*

  – Fas 0 – imperativ programmering med C (3 sprintar)

  – Fas 1 – objektorienterad programmering med Java (3 sprintar)

  – Fas 2 – projektarbete, verktyg och testdriven utveckling (n sprintar)

– Ca 12 timmar *schemalagd* tid/vecka i fas 0 & 1, ca 4/vecka i fas 2

> **Utöver** schemalagd tid förväntas *en genomsnittlig student* lägga ned **ytterligare 20 timmar i veckan** i fas 0 och 1 och nästan 30 h / v i fas 2.

30 hp = 20 arbetsveckor = 800 h
800 h / 16 kalenderveckor = 50 h per vecka
(Idiotin att läsa 30 hp på 16 veckor ger en 125% arbetstakt)

Undervisningstyper

- Screencasts
  - Programspråk
  - Verktyg
- Föreläsningar
  - Traditionella
  - Inspelade och/eller ”flippade”
  - 1 i veckan i fas 0, ca $1\frac{1}{2}$ i veckan fas 1, och 1 under *hela* fas 2 (ca)
- Laborationer
  - 8 h / vecka i fas 0 och 1; 4 h / vecka i fas 2 (ca)
- Gruppmöten
  - 2 h i slutet av varje sprint plus en extra i slutet = totalt 14 h

- Kick-off för projektet
  - 5-timmars gästföreläsarbonanza

Mål

Målen är uppdelade i tre kategorier med följande fördelning med avseende på nivåerna på de olika målen:

Typ av mål	# 3	# 4	# 5
Kunskapsmål	25	18	9
Verktygsmål	9	2	1
Strukturmål	9	2	1
$\sum = 76$	43	22	11

Kod	Beskrivning	# 3	# 4	# 5
L	Redovisas i samband med labbtillfälle	32	18	5
G	Redovisas i samband med gruppmöte	12	17	10
T	Redovisas på frivillig tenta	1	2	5
I	Intygas	1		
R	Redovisas genom en rapport	5		
W	Skickas in via epost		1	

Några exempel på mål

Abstraktion	Nivå	Red.
Konsekvent tillämpa procedurell abstraktion för att öka läsbarheten och undvika upprepningar	3	L
Tillämpa objektorienterad abstraktion för att dölja implementationsdetaljer bakom väldefinierade gränssnitt	3	L
Demonstrera förståelse för designprincipen informationsgömning i ett C-program med hjälp av `.c` och `.h-filer`	4	L, G
Arv	**Nivå**	**Red.**
Använda arv, metodspecialisering och superanrop i ett program som drar nytta av subtypspolymorfism	3	L
Använda både överlagrade och specialiserade konstruktorer på lämpligt sätt i programmering	3	L
Förklara hur arvsbegreppet har använts i ett program för att separera genomskärande åtaganden	4	G, T
Förklara hur man kan undvika s.k. ”bräckliga basklasser” vid arv	5	G, T

Arbete

– Allt arbete sker i *par* om två

– Varje sprint roterar *ni* själva paren beroende på intressen och pragmatik

– Varje fas slumpar *vi* om grupperna

Redovisningar av mål i stora drag

– I samband med labbtillfälle

  Paret ansöker om redovisning i vårt webbsystem och får en köplats till en assistent för vilken ni skall presentera er förståelse för målen.

– Vid gruppmötet i slutet av varje sprint

  Paret meddelar labassen vilka mål som skall redovisas och förbereder en demonstration/presentation/etc. som visar målen under gruppmötet.

– Vid den frivilliga tentan

  Eventuella kvarstående mål märkta ”T” kan redovisas på tentan.

– Man får redovisa max 6 mål vid varje gruppmöte och max 4 vid varje labbtillfälle

– Vid den frivilliga tentan får man redovisa max 6 mål.

– Kodprovet kan inte användas för att redovisa mål.

Uppgifter

- Vid varje fas (ibland) sprint släpps nya uppgifter
- Du får välja mellan samtliga släppta uppgifter under hela kursens gång
- Du kommer att behöva programmera i minst C och Java under kursen
- Du väljer vilka uppgifter du vill göra utifrån vilka mål du vill boka av etc.
- Vi rekommenderar att du gör (minst) en uppgift från varje sprint i den ordning de kommer

Högskolepoäng och kursfordringar

5 hp utgår för kodprovet

5 hp utgår för projektuppgiften (fas 2 avklarad)

5 hp utgår för fas 0 avklarad

5 hp utgår för fas 1 avklarad

Avklarade faser

1. Definitionen av fas 0 avklarad:
   (a) Godkända implementationer av två fas 0-uppgifter och minst
   (b) 3 avbockade verktygsmål på nivå 3
   (c) 11 avbockade kunskapmål på nivå 3 (plus Z 77)
2. Definitionen av fas 1 avklarad:
   (a) Godkända implementationer av två fas 1-uppgifter och minst
   (b) 3 avbockade verktygsmål på nivå 3
   (c) 14 avbockade kunskapmål på nivå 3
3. Definitionen av fas 2 avklarad:
   (a) Godkänd projektuppgift inkl. rapport och samtliga projektmål på nivå 3 avbockade
   (b) 3 avbockade verktygsmål på nivå 3

Screencasts

- Dessa ersätter föreläsningar som mest gick ut på att föreläsaren visade syntaxen för olika konstruktioner
- Livekodning har varit uppskattat tidigare år
- Bra med kortade avsnitt, man kan pausa och se om
- Verktygsdelen är ny
- Försök beta av verktyg och C så fort som möjligt under fas 0
- Försök beta av Java så fort som möjligt under fas 1

Föreläsningar

- Vissa föreläsningar hålls som vanligt (som t.ex. denna)
- Andra föreläsningar hålls som en *kombination* av videomaterial och föreläsning
  - Dessa föreläsningar kan komma att innehålla övningar och mer aktivt arbete
  - Att först ta del av videomaterialet är *obligatoriskt* för den som vill delta på föreläsningen
  - Inspelat material, inklusive screencasts kan innehålla frågor – svaren på dessa anonymiseras och presenteras för föreläsare som hjälpmedel inför designen av nästa föreläsning
- Deltagande på föreläsningar är inte obligatoriskt, **men** vissa övningar etc. under föreläsningar kan komma att räknas som avbockade mål

Gruppmötet

- Ger dig möjlighet att redovisa litet mer komplicerade/intrikata mål i mindre grupp
- Du får lyssna och ta del av andras presentationer och komma med återkoppling och hjälp
- Du får hjälp att planera arbetet inför nästa sprint, hjälp att förklara mål, etc.
- Du är redan indelad i en grupp (eller kommer att bli så fort du har loggat in i vårt system (se bild om verktyg))
- Vi slumpar om grupperna varje fas
- I fas 2 skall gruppen göra projektarbetet

Projektet

- Det kommer information om detta senare under projektets gång.
- Det kommer att bildas grupper för projektarbetet
- 2012 hade vi implementation av automatisk minneshantering i C
- 2013 har vi ännu inte beslutat om vilket projekt vi skall göra
- Man kan använda projektet för att bocka av mål
- Målet med projektet är inte att göra en 100%-ig implementation av en specifikation: projektuppgiften är en förevändning för dig att hamna i utvecklade situationer

Kursverktyg

- Information om kursen: `http://wrigstad.com/ioopm`
- Allt utdelat material, alla bilder, frågor, kursmål, etc. i kursens repo:
  - Dess GitHub-sida (se länk från kurswebben)
  - Klona med:
    `git clone git://github.com/TobiasWrigstad/ioopm.git`
- Redovisning, framsteg, etc.:
  `http://wrigstad.com/ioopm/iooPortal`
- Distribution av inspelat material med insprängda frågor
  `http://scalable-learning.com`
- Diskussionsforum och on-line-handledning
  `https://piazza.com/class/hkjhvzyqnp52on`

Föreläsning 1 b)

Introduktion till C

C

- Imperativt programmeringsspråk
  - Satser (kommandon) som utförs i *sekvens*
  - Data (variabler, etc.) som manipuleras
  - Funktioner med sidoeffekter
  - Ofta iterationer (loopar)
- Maskinnära men har stöd för maskinoberoende programmering
  - minnesadresser, adressaritmetik
  - bitmanipulering
  - ...
- Ett litet språk med begränsade standardbibliotek

– Utvecklades ursprungligen av Dennis Richie 1969–1973 för att underlätta implementation av systemprogramvara (bl.a. OS)

– Har influerat många efterkommande programspråk (t.ex. C++ och Java) på gott och ont

C utvecklades för att vara effektivt

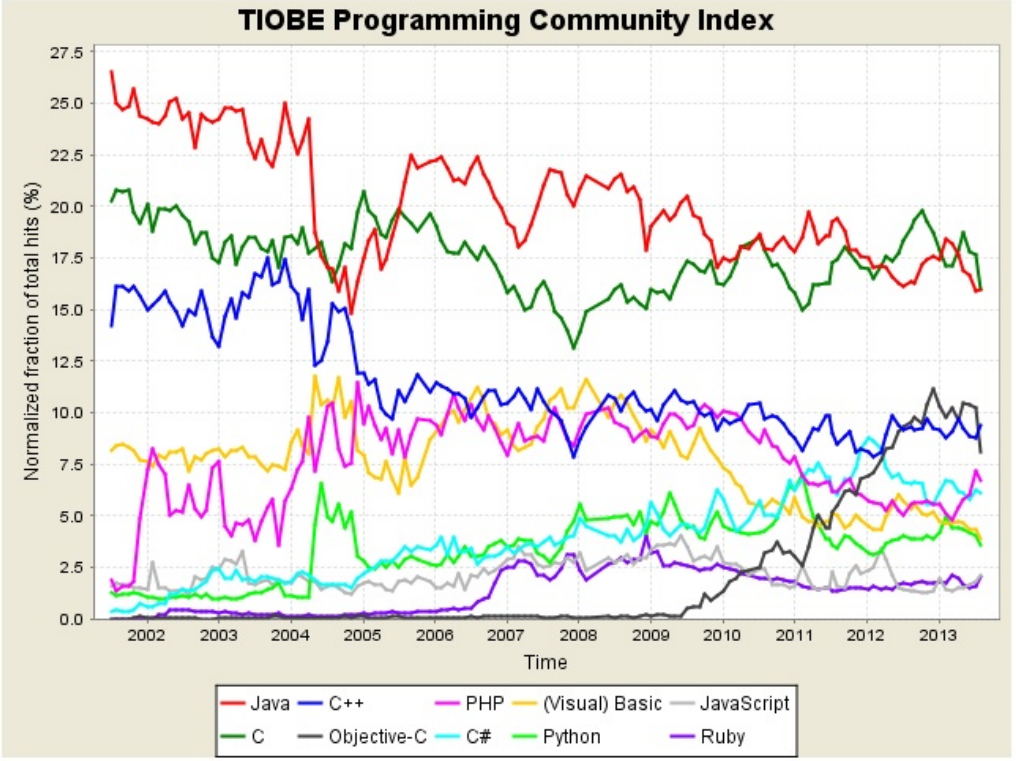
- ...i en era av väldigt begränsad datorkraft, vilket har påverkat vilka funktioner som finns i språket som standard
- Manuell minneshantering utanför stacken
- Direkt minnesåtkomst
- Statiskt, manifest och svag typning
- Inline-assembler
- Ingen exekveringsmiljö och inget metadata under körning

C – fortfarande relevant

Position Aug 2013	Position Aug 2012	Delta in Position	Programming Language	Ratings Aug 2013	Delta Aug 2012	Status
1	2	↑	Java	15.978%	-0.37%	A
2	1	↓	C	15.974%	-2.96%	A
3	4	↑	C++	9.371%	+0.04%	A
4	3	↓	Objective-C	8.082%	-1.46%	A
5	6	↑	PHP	6.694%	+1.17%	A

Tiobe index, augusti 2013

C – fortfarande relevant om 10 år?



Demo

1. Hello, world
2. Kvadrater

Att skriva, bygga och köra ett C-program

1. **Skriv programmet i en kraftfull texteditor**
2. **Källkoden *kompileras* till *objektkod***
3. (I samband med detta kör vi också ett antal verktyg för felkontroll mer om det senare)
4. **Objektkoden *länkas* med biblioteksfunktioner till ett exekverbart program**
5. (Nu bör vi köra programmets tester för att hitta alla fel vi gjort)
6. **Nu kan programmet köras!**
7. (Och när programmet kraschar använder vi en debugger för att undersöka programet)
8. (Och när programmet går för långsamt använder vi profileringsverktyg för att förstå varför)
9. (Och när programmet läcker minne använder vi särskilda verktyg för att spåra läckage)
10. (Och så använder vi verktyg för att generera dokumentation från vår kod)

– Det finns en uppsjö ”integrerade utvecklingsmiljöer” (IDE:er) för C, t.ex. Netbeans, Eclipse, Xcode, m.fl. som integrerar många av dessa verktyg

– På denna kurs är det *obligatoriskt* att använda Emacs för utveckling i C och separata verktyg för allt annat; vi skall möta Netbeans senare.

Emacs

– Det ingår i en programmerares verktygslåda att *behärska* minst ett kraftfullt verktyg för editering och textmanipulering

– IDE:er är dåliga på textmanipulering och har en massa ”bells and whistles” som vi ofta inte behöver

– Att använda separata verktyg tyddliggör de olika processerna/stegen i utveckling på ett sätt som gör det lättare att använda IDE:er i framtiden (eller andra editorer och verktyg)

– Emacs är valt på pragmatiska grunder, inte för att vi vill frälsa er för just Emacs (pröva gärna t.ex. Vi också, men *efter* kursen!)

Föreläsning 1 c)

Vad skall du göra nu?

Översikt för denna vecka = fas 0/sprint 0

1. Se till att logga in på iooPortal (du får du ett gruppmedlemsskap)
2. Läs igenom kurswebben, bokmärk alla länkar
3. Hitta denna veckas labbpartner (måste vara en annan gruppmedlem)
4. Gör uppgiften fas 0/sprint 0
5. Studera kursmålsdokumentet i kursens repo
6. Titta på videomaterialet för fas 0/sprint 0 (ca 5 timmar)
7. Studera vidare med hjälp av online-material eller föreslagna kursböcker
8. Sök hjälp på laborationerna (tisdag 13–17 och torsdag 13–17), eller på Piazza
9. Redovisa dina första mål under laborationerna
10. Delta på gruppmötet på fredag, redovisa och planera din framtid!