Välkomna till Programmeringsteknik I

Föreläsning 1: Introduktion till kursen

- Lärare:
  - Tom Smedsaas och
  - Torsten Andersson samt
  - samt ca 24 assistenter
- Registrering / avregistrering
- Undervisningsformer: föreläsningar och laborationer
- Kursmaterial
  - På nätet: Eget och andras material
  - Kursböcker: Se litteraturlänk på kurssidan
- Kusens hemsida:
  http://www.it.uu.se/edu/course/homepage/prog1/vt19/

Laborationerna

- Datorsystem
  - Linux med ”tunna klienter” på institutionen.
    Konto: UpUnet-S, lösenord A.
  - Egna datorer - även på laborationerna
    - Windows
    - Mac
    - Linux

Obligatoriska moment

1. Tentamen fredagen den ?? mars.

   Tvådelad:

   - del **A** som är obligatorisk för alla och
   - del **B** för betyg 4 och 5.

2. Vissa lektioner (4, 5, 7, 9 och 10) skall redovisas muntligt i labbsal **senast** på tider som står på kurshemsidan.
   Vid dessa ska man också kunna svara för materialet på de föregående lektionerna.

3. Ingen obligatorisk närvaro förutom vid redovisning.

Föreläsningarna

Både

retrospektiv: Vad handlade de närmast föregående lektionerna om?

och

prospektiv: Vad handlar de närmaste kommande lektionerna om?

Begrepp introduceras ofta först på en lektion för att sedan sammanfattas på en föreläsning.

Föreläsningsanteckningarna läggs normalt upp på nätet i förväg (men den kan ändras ända in i det sista.)

Laborationerna

- Varje grupp har 30 schemalagda laborationstillfällen för arbete med och redovisning av kursens nätlektioner.
- OK att gå på andra gruppers tider under förutsättning att det finns plats *men* endast de som är schemalagda på tiden kan påräkna handledning och redovisningstid.
  Tills vidare får de som inte har en tydlig grupptillhörighet (fristående kurs) välja vilken grupp som helst.
- Utnyttja även laborationstiderna 8-10!

Laborationerna

- Assistenternas uppgift är INTE att hitta felen i era program utan att tala om hur man ska bära sig åt för att hitta felen.
- Assistenterna uppgift är INTE att återberätta föregående föreläsningar eller lektioner.
  Om du har missat en föreläsning så måste du läsa på motsvarande material själv!
- Räkna med att du måste lägga ner en hel del tid utöver de schemalagda laborationerna! Kursen ska motsvara 3 veckors heltidsarbete dvs 120 timmar ...

Redovisningar av de obligatoriska lektionerna

- De obligatoriska lektionerna ska redovisas muntligt vid dator för lärare/assistent senast enligt de tider som står på kurssidan.
- Du måste vara beredd på att legitimera dig vid redovisningen.
- Det är tillåtet att samarbeta men varje student måste ha sin version av koden och redovisa individuellt.
- Vid redovisningarna ges individuella frågor och extra kodningsuppgifter.
- Vi prioriterar redovisningar av de enligt schemat aktuella uppgifterna.
- Skriv gärna upp namnet på den du redovisat för - det är lättare att reda ut om någon bokföring missas.
- Vi tar inga redovisningar efter kursens slut.

Vad händer om du missar något?

- Om du inte blir godkänd på en redovisning så har du en vecka på dig att komplettera.
- Om du blir försenad med en lektion så kontakta lärare senast den sista redovisningsdagen. Vi kan bevilja enstaka dispenser.
- Vi ordnar normalt några repetitionstillfällen inför omtentan i augusti. I samband med dessa kan det gå att redovisa någon enstaka uppgift.
- Om du har flera uppgifter kvar får du göra uppgifterna nästa gång kursen ges (troligen period 1, ht 2019)

Se till att du har en fungerande mail och att du läser mailen emellanåt!

Hur ska man göra för att klara kursen?

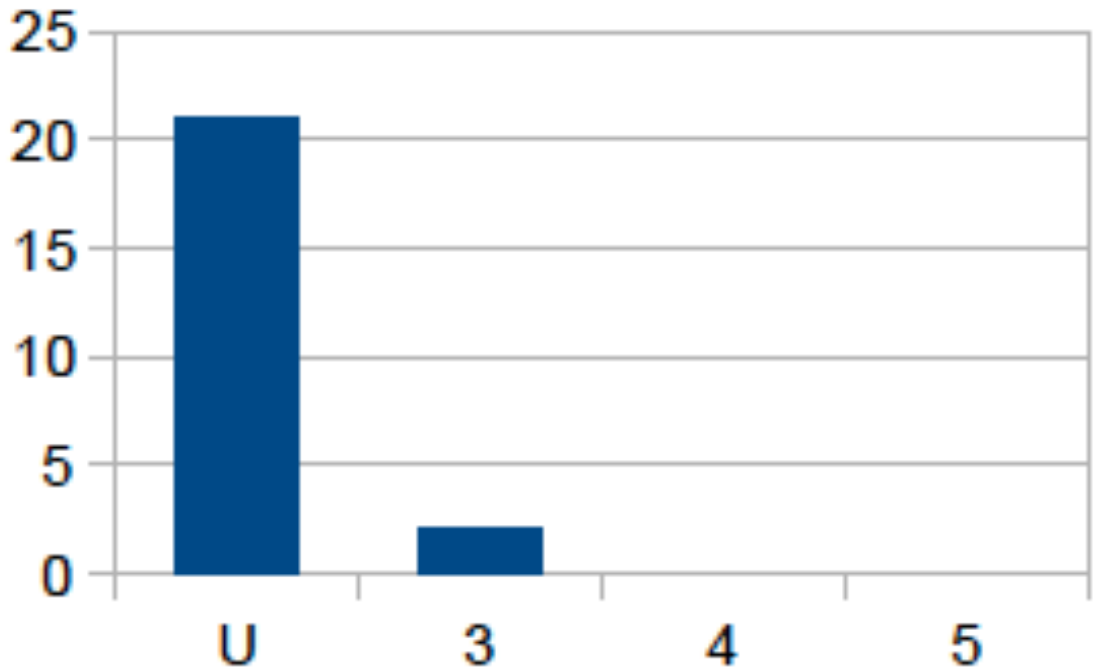
Pedagogisk forskning:

Det enskilt viktigaste momentet är eget arbete vid tangentbordet!

Uppsalaundersökning:

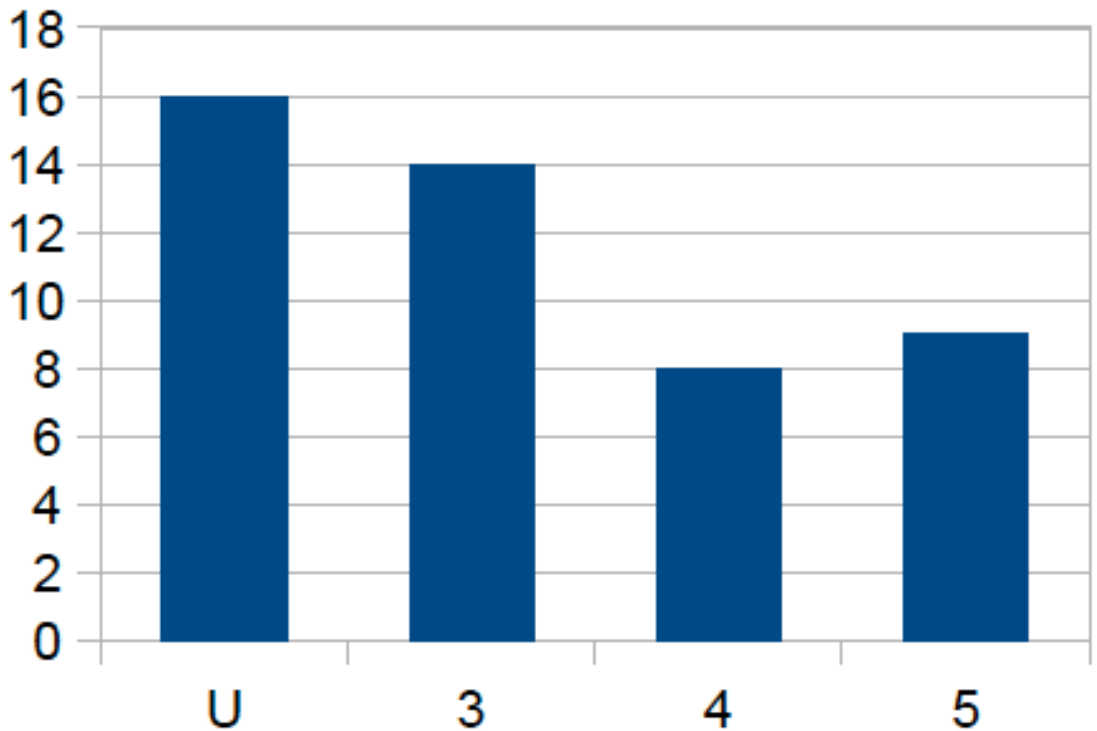
Nybörjarprogrammerare som fick betyg 4 eller 5 på tentan har suttit dubbelt så lång tid som de som blev underkända.
Inga andra faktorer som hade signifikant påverkan.

Resultat av samma kurs vt 2013

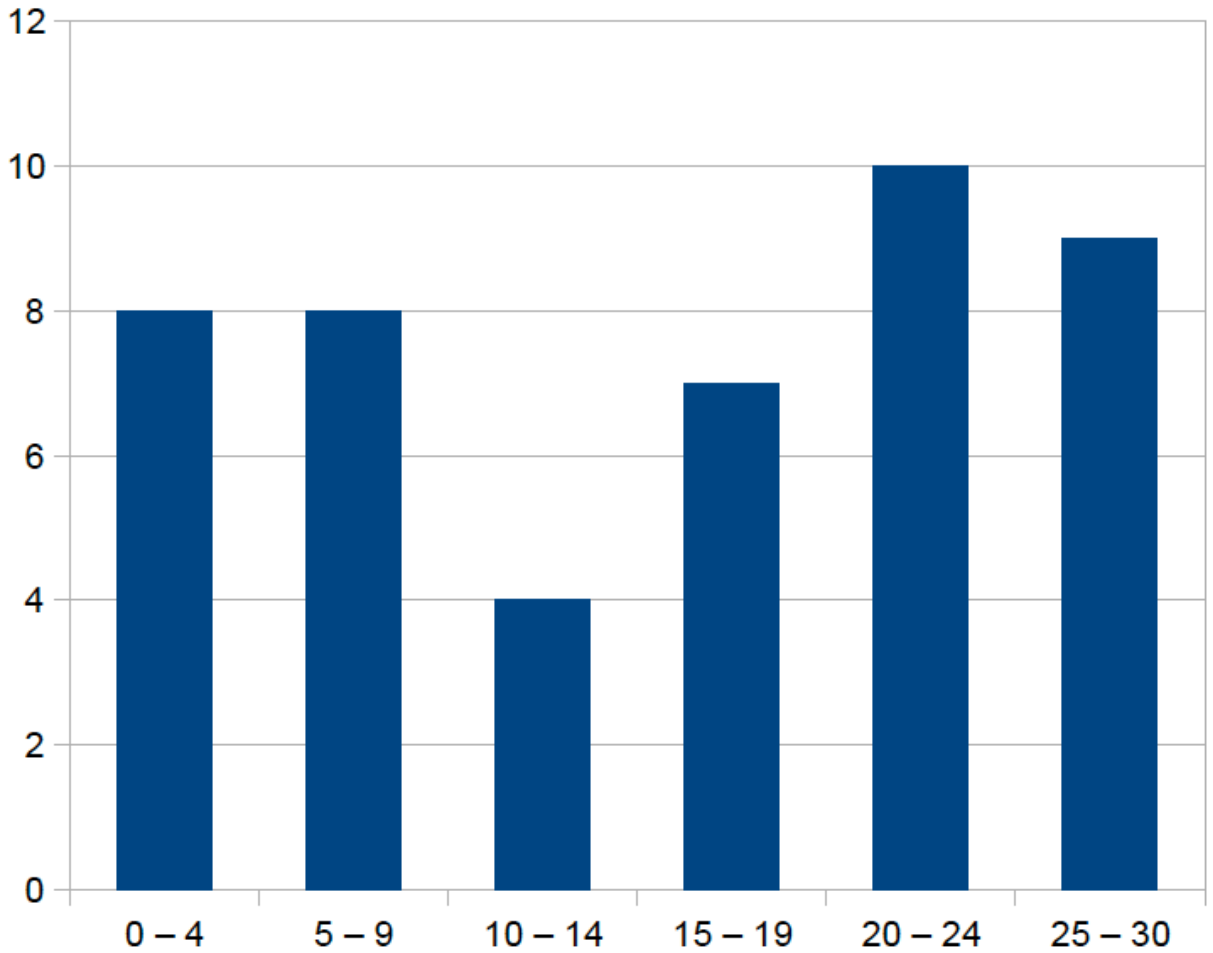


för studenter som inte gjort alla obligatoriska uppgifter

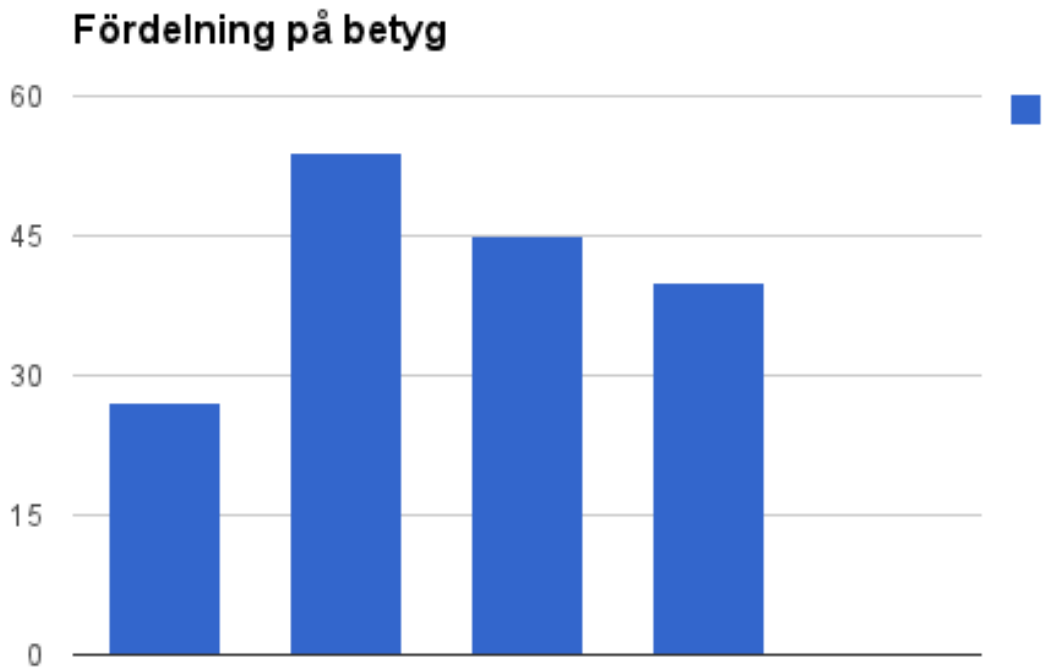
För studenter som var klara med OU 2013



Poängfördelning för dem som var klara med OU



Resultat VT 2014



Ca 85% godkända.

Samma kurs, samma innehåll, samma uppgifter, samma lärare men **individuell** redovisning av uppgifterna.

Slutsatser

- Det är *nödvändigt* att arbeta med de obligatoriska uppgifterna!
- Samarbete är bra *men* alla måste delta *aktivt*!
- Alla måste *skriva* sin egen kod även om man samarbetar!

Föregående kursvärderingar

Generellt sett nöjda studenter.

Två problem som brukar lyftas fram:

- Tentamen på papper — svårt att skriva kod på papper.
  Vi förstår men praktiska problem att ha examination i datorsal.
  Vi försöker hantera problemet med tentans utforming. A-delen så handlar det mer om att komplettera given kod än att skriva egen.

- För få assistenter.
  Vänta inte med redovisning till sista dagen. Vi har successivt utökat antalet laborationstillfällen så det finns gott om tid för det.
  Utnyttja även tiderna 8–10 och 15–17!

Formell kursplan

Mål

Efter godkänd kurs ska studenten kunna:

- redogöra för de grundläggande begreppen klass, objekt, inkapsling och därtill hörande underbegrepp;
- analysera problem och designa lösningar genom att använda ovanstående begrepp;
- använda programmeringsspråket Java genom att
  - förklara vad ett givet program utför
  - skriva och använda klasser som innehåller instansvariabler, metoder och konstruktorer
  - skriva program med flera samverkande klasser;
- använda en programutvecklingsmiljö;
- testa och felsöka program.

Formell kursplan forts

Innehåll

1. Programmering
2. Algoritmer
3. Objektorienterad problemlösning
4. Datatyper
5. Programmeringsteknik

Programmeringsspråket Java

- Generellt
- Objektorienterat
- Syntaktiskt likt språk som C++ och C#
- Väldefinierat
- Stor mängd fördefinierade standardiserade komponenter (”*klasser*”):
  - Internet
  - Grafik
  - Användarinterface
  - ...
- Implementerat på alla vanliga datorsystem
- Portabelt
- Fritt att ladda ner från nätet

Programmeringsmiljöer

Minimalt:

- en *editor* för att redigera programtexten,
- en *kompilator* för att översätta programmet till instruktioner som är mer lämpade för datorn att tolka och
- en *javamotor* som är det som utför instruktionerna,

Integrerade miljöer (”IDE”) som innehåller alla dessa delar. Exempel:

- DrJava
- Eclipse
- NetBeans
- ...

Fastnat för DrJava i denna kurs