HÖGSKOLAN I BORÅS

Data- och programstrukturer
Programming Abstractions and Data Structures

7,5 högskolepoäng
7,5 credits

Ladokkod: NDP011
Version: 1.0
Fastställd av: Institutionsstyrelsen 2007-03-16
Gäller från: HT 2007

Nivå: Grundnivå
Huvudområde (successiv fördjupning): Informatik (G1F), Datalogi (G1F)
Utbildningsområde: Naturvetenskap
Ämnesgrupp: Informatik/Data- och systemvetenskap
Förkunskapskrav: Grundläggande behörighet samt genomgången kurs Programmeringsteknik, 7,5 hp
Betygsskala: Underkänd, Godkänd eller Väl godkänd

Innehåll

Kursen är en fortsättningskurs i strukturerad programmering. Kursens behandlar därför dels generella begrepp och generella lösningar vid strukturerad programmering och dels mer avancerade koncept i programspråket C. Kursen varvar teori med praktik på så sätt att varje huvudavsnitt först behandlas vid föreläsningar och övningar för att därefter övas och prövas på laborationerna. Under laborationsperioderna arbetar studenterna enbart med respektive laboration; dvs. då hålls inga föreläsningar eller övningar. De moment som ingår i kursen är:

- **Rekursion:** Rekursiva algoritmer används flitigt genom hela kursen varför det rekursiva tankesättet introduceras tidigt och i detalj. För att illustrera rekursion används såväl enklare algoritmer som mer avancerade. Två exempel är MiniMax och olika varianter på backtracking. Algoritmen MiniMax är grunden för rationella beslut i tvåpersoners nollsummespel.
- **Komplexitet:** Resonemang om komplexitet genomsyrar hela kursen. Snart sagt varje implementering diskuteras utifrån ett komplexitetsperspektiv och studenten tränas kontinuerligt i att avgöra vilken komplexitet olika operationer har. Som grund för komplexitetsanalys används ”Ordo-notationen” och dess räkneregler. En mycket kort introduktion till distinktionen mellan hanterliga och ohanterliga problem ges. Specifikt diskuteras konsekvenserna av att det finns ohanterliga problem. Problemklasserna P och NP behandlas översiktligt.
- **Sortering:** I kursen gås en mängd sorteringsalgoritmer igenom. För samtliga identifieras och diskuteras dess komplexitet i olika fall. Specifikt analyseras algoritmerna MergeSort och QuickSort samt någon variant av linjär sortering; t.ex. BucketSort.
- **Linjära datastrukturer:** I kursen behandlas länkade listor ingående. Detaljer avseende implementering och användning gås igenom, liksom representationer som kombinerar länkade listor och vektorer. I samband med detta diskuteras även hashning och hashtabeller. Stackar, köer och prioritetsköer behandlas i detalj, med särskilt fokus på hur intern representation påverkar olika operationers effektivitet.
- **Abstrakta datatyper (ADT:er)**. Här behandlas teoretisk motivation för ADT:er, principer för design av ADT:er och hur implementering i C sker. Även generella mekanismer som t.ex. iteratorer behandlas. Samtliga teoretiska koncept illustreras med ett flertal exempel. Ett urval av de ADT:er som gås igenom i kursen är: stack, kö, lista, scanner, textbuffert, symboltabell och binärt sökträd. ADT:er används genomgående för att skapa avancerade applikationer, exempelvis texteditorer, där flera ADT:er typiskt används i en applikation.
- **Rekursiva datastrukturer:** Rekursiva datatyper introduceras med hjälp av listor och i samband med detta diskuteras de rekursiva algoritmer som hanterar listor. Störst tonvikt läggs dock på träd, där fokus ligger på binära sökträd. Implementering av träd med hjälp av länkade strukturer utförs och algoritmer för insättning, borttagning och sökning efter noder, samt traversering av binära sökträd gås igenom. Slutligen diskuteras trädbalansering och algoritmer för detta.
- **Uttrycksträd och interpretatorer:** Syftet med detta avsnitt är tudelat, dels är momentet en tillämpning av träd och dels en introduktion till generella koncept inom programspråk. Träd används för att representera uttryck i formella språk. Begreppet grammatik introduceras och illustreras med ett flertal mindre exempel. Förfarandet att skriva en

parser givet en grammatik presenteras och även evaluering av uttryck gås igenom. Samtliga steg från språkspecifikation till evaluering illustreras med ett exempel där en interpretator för aritmetiska uttryck konstrueras.
- **Avancerade konstruktioner i språket C:** Då kursen även är en fortsättningskurs i språket C, gås ett flertal nya konstruktioner igenom. Framförallt utnyttjas pekare och dynamisk minnesallokering för att skapa olika länkade strukturer. Dessutom presenteras tekniken att använda funktionspekare för att åstadkomma högre ordningens funktioner. Detta används även i ett flertal tillämpningar, t.ex. symboltabeller och binära sökträd. Unioner introduceras också och används främst i olika trädtillämpningar, exempelvis uttrycksträd.

Mål

Kursens mål är att studenten ska ha tillägnat sig en teoretisk och praktisk fördjupning inom imperativ programmering. Studenten ska efter genomgången kurs kunna tillämpa samtliga ingående teoretiska moment (se under innehåll) och därmed specifikt kunna:

- självständigt kunna konstruera avancerade applikationer med textbaserat gränssnitt utifrån en given kravspecifikation.
- såväl utnyttja som implementera rekursiva algoritmer och rekursiva datatyper.
- redogöra för den grundläggande teorin bakom komplexitetsanalys.
- genomföra enklare komplexitetsanalys av iterativa algoritmer.
- implementera och redogöra för egenskaperna hos de viktigaste sorteringsalgoritmerna.
- ha goda kunskaper och utifrån teorin redogöra för och motivera användning av abstrakta datatyper. Kunna utnyttja abstrakta datatyper för att erhålla en programdesign som uppfyller de viktigaste kriterierna för god programvara ochredogöra för och motivera nyttan av koncepten ”inkapsling”, ”generiska datatyper” och ”programmering mot gränssnitt”. Kunna skapa inkapslade och generiska datatyper i språket C och välja och utnyttja lämpliga abstrakta datatyper vid programkonstruktion och givet en kravspecifikation kunna designa ett gränssnitt för en abstrakt datatyp.De ska kunna implementera en abstrakt datatyp med val av lämplig konkret representation. Kunna beskriva, använda och implementera de datastrukturer och abstrakta datatyper som kursen behandlar samt beskriva, använda och implementera algoritmer för hantering av de datatyper som ingår i kursen.
- givet en enkel grammatik och semantik kunna konstruera en interpretator för uttryck i det specificerade språket.
- redogöra för och använda avancerade konstruktioner i språket C.

Undervisningsformer

Föreläsningar, övningar och laborationer med tillhörande handledning. Undervisningen bedrivs normalt på svenska, men undervisning på engelska kan förekomma.

Examinationsformer och betygsskala

Examination på kursen består av 4 obligatoriska laborationer och en tentamen. Tentamen genomförs skriftligen.

Laborationsmomentet ger 2,5 högskolepoäng.
Tentamen ger 5,0 högskolepoäng.

Laborationerna poängsätts och för att vara godkänd på laborationsmomentet krävs att studenten på de fyra laborationerna totalt uppnår en viss poängsumma. Laborationernas poängsättning framgår nedan.

Lab 0 – 20 laborationspoäng.
Lab 1 – 70 laborationspoäng.
Lab 2 – 80 laborationspoäng.
Lab 3 – 80 laborationspoäng.

Laborationsmomentet betygssätts som antingen godkänt eller underkänt. För betyget godkänt fordras 150 laborationspoäng av de 250 möjliga.

På tentamen ges ett av betygen väl godkänt, godkänt eller underkänt.

För att vara godkänd på kursen krävs godkänt betyg på laborationsmomentet samt godkänt på tentamen. För betyget väl godkänt krävs godkänt betyg på laborationsmomentet samt väl godkänt på tentamen

Kursens färdighetsmål examineras både på laborationerna och på den skriftliga tentamen. I laborationerna löser studenterna i par komplexa uppgifter vilka innebär att realisera kravspecifikationer och resonera om komplexitet. På tentamen prövas studentens teoretiska förståelse och förmåga att konstruera mindre program eller programavsnitt.

Studentens rättigheter och skyldigheter vid examination är enligt riktlinjer och regelverk vid Högskolan i Borås.

Kurslitteratur och övriga läromedel

Se bilaga.

Studentinflytande och utvärdering

Kursutvärderingen är skriftlig. Sammanställning av kursutvärderingen samt eventuella planerade förändringar till nästa kursgenomförande delges studenterna.

Övrigt

Kursen ingår i Systemarkitekturutbildningen och Systemvetarutbildningen.

Kurser ersätter Data- och programstrukturer (NDPG01).

Bilaga: Litteraturlista för Data- och programstrukturer (NDP011)

Robert, Eric S, *Programming Abstractions in C – A Second Course in Computer Science,* ISBN: 0201545411