GÖTEBORGS UNIVERSITET

INSTITUTIONEN FÖR DATA- OCH INFORMATIONSTEKNIK

DIT260 Avancerad funktionell programmering, 7,5 högskolepoäng

Advanced Functional Programming, 7.5 credits

Avancerad nivå / Second Cycle

Fastställande

Kursplanen är fastställd av IT-fakultetsnämnden 2009-09-19 och senast reviderad 2017-12-19 av Institutionen för data- och informationsteknik. Den reviderade kursplanen gäller från och med 2018-08-19, höstterminen 2018.

Utbildningsområde: Naturvetenskapligt 100 %
Ansvarig institution: Institutionen för data- och informationsteknik

Inplacering

Kursen erbjuds inom flera utbildningsprogram. Den ges även som fristående kurs vid Göteborgs universitet.

Kursen kan ingå i följande program: 1) Datavetenskapligt program (N1COS), 2) Computer Science, Master's Programme (N2COS) och 3) Applied Data Science masterprogram (N2ADS)

Huvudområde	*Fördjupning*
Datavetenskap	A1F, Avancerad nivå, har kurs/er på avancerad nivå som förkunskapskrav

Förkunskapskrav

För tillträde till kursen krävs att studenten har minst 120 hp i datavetenskap eller motsvarande. Specifikt krävs följande kurser, eller motsvarande:

- DIT143 Funktionell programmering, 7,5 hp, eller DIT440 Introduktion till funktionell programmering, 7,5 hp,
- DIT980 Diskret matematik för datavetare, 7,5 hp,
- DIT231 Programming language technology, 7,5 hp.

Följande kunskapsnivå i Engelska krävs; Engelska 6/Engelska B eller motsvarande från ett erkänt internationellt test, t.ex. TOEFL, IELTS.

Lärandemål
Efter godkänd kurs ska studenten kunna:

Kunskap och förståelse
- förklara avancerade typsystemegenskaper, såsom typklasser, generaliserade algebraiska datatyper, funktorer, monader och monadtransformerare, och relatera dem till varandra

Färdigheter och förmåga
- utforma inbäddade domänspecifika språk (EDSL); förklara och exemplifiera deras abstrakta och konkreta syntax och semantik; och implementera dem i Haskell som kombinatorbibliotek
- använda specifikationsbaserade utvecklingstekniker för att formulera och testa egenskaper kring program
- redogöra för korrektheten hos funktionella program och omvandla dem på grundval av sådana resonemang
- analysera och utöka Haskell-program som använder avancerade typsystemegenskaper

Värderingsförmåga och förhållningssätt
- diskutera ovanstående ämnen (dvs typsystemegenskaper, EDSL, specifikationsbaserade tekniker och korrekthet), och hur de relaterar till varandra

Innehåll
Kursens mål är att utforska de kraftfulla mekanismer som funktionella programspråk erbjuder när det gäller att lösa verkliga problem och strukturera större program. Fokus ligger på design av programbibliotek och inbäddade språk.

En stor fördel med funktionella programspråk är att de flesta språkkonstruktionerna kan namnges och därmed återanvändas som högre ordningens funktioner. Funktionella program kan därför ofta konstrueras genom att kombinera konstruktioner från ett funktionsbibliotek. Den här metoden gör det möjligt att snabbt konstruera program med en hög grad av korrekthet. Detta är den centrala idén i kursen.

Vi kan lära oss en hel del genom att studera standardbibliotekens list-funktioner som map, fold osv. Dessa funktioner kan generaliseras så att de fungerar för andra datatyper.

Realistiska funktionella program måste också hantera tillståndsförändringar, avbrott, "backtracking" och andra "icke-funktionella" beteenden. Vi kommer att jobba med hur dessa kan modelleras rent funktionellt. Det matematiska begreppet "monad" hjälper oss

med detta.

Med hjälp av dessa kunskaper kommer vi att konstruera domänspecifika programbibliotek ämnade att lösa problem inom ett ett visst tillämpningsområde. Den här sortens bibliotek kan sägas definiera ett domänspecifikt språk eftersom konstruktionerna som programmeraren använder huvudsakligen består av biblioteksfunktioner. Vi kommer att studera bibliotek för inläsning (parsning), utskrift (pretty printing), grafik, webbprogrammering och interaktion. Kursen kommer också att presentera en del aktuell forskning vilket kan göra att innehållet varierar en del mellan åren. Kursen använder sig huvudsakligen av programmeringsspråket Haskell.

Delkurser

1. **Skriftlig tentamen** *(Written examination),* 3 hp
   Betygsskala: Väl godkänd (VG), Godkänd (G) och Underkänd (U)
2. **Laboration** *(Laboratory work),* 4,5 hp
   Betygsskala: Väl godkänd (VG), Godkänd (G) och Underkänd (U)

Former för undervisning

Föreläsningar, laborationer, handledning och självstudier. Studenterna förväntas lägga mycket egen tid på programmering och självstudier.

Undervisningsspråk: engelska

Former för bedömning

Kursen examineras av 2-3 programmeringslaborationer (U-VG) som normalt utförs i par under kursen, och en individuell skriftlig tentamen (U-VG) i slutet av kursen.

Om student som underkänts två gånger på samma examinerande moment önskar byte av examinator inför nästa examinationstillfälle, ska sådan begäran inlämnas skriftligt till kursansvarig institution och bifallas om det inte finns särskilda skäl däremot (HF 6 kap § 22).

I det fall en kurs har upphört eller genomgått större förändringar ska studenten i normalfallet garanteras tillgång till minst tre provtillfällen (inklusive ordinarie provtillfälle) under en tid av åtminstone ett år med utgångspunkt i kursens tidigare uppläggning.

Betyg

På kursen ges något av betygen Väl godkänd (VG), Godkänd (G) och Underkänd (U). För godkänt betyg på hel kurs krävs godkänt betyg på samtliga delkurser.

Det slutgiltiga betyget för hela kursen är till 60% baserat på laborationerna och till 40% på den skriftliga tentamen.

Kursvärdering

Kursen utvärderas genom möten både under och efter kursen mellan lärare och studentrepresentanter. Därutöver används en anonym enkät för att få skriftlig information. Resultatet av utvärderingen används för att förbättra kursen genom att visa på delar som kan läggas till, förbättras, ändras eller tas bort.

Övrigt

Kursen är samläst med Chalmers.

Kurslitteratur kommer att publiceras senast 8 veckor innan kursstart.

Det är rekommenderat, men inte ett krav, att läsa följande kurser i förväg: DIT602 Algoritmer, samt någon av DIT201 Logik i datavetenskap eller DIT321 Finita automater och formella språk.