Stichting NIOC en de NIOC kennisbank

Stichting NIOC (www.nioc.nl) stelt zich conform zijn statuten tot doel: het realiseren van congressen over informatica onderwijs en voorts al hetgeen met een en ander rechtstreeks of zijdelings verband houdt of daartoe bevorderlijk kan zijn, alles in de ruimste zin des woords.

De stichting NIOC neemt de archivering van de resultaten van de congressen voor zijn rekening. De website www.nioc.nl ontsluit onder "Eerdere congressen" de gearchiveerde websites van eerdere congressen. De vele afzonderlijke congresbijdragen zijn opgenomen in een kennisbank die via dezelfde website onder "NIOC kennisbank" ontsloten wordt.

Op dit moment bevat de NIOC kennisbank alle bijdragen, incl. die van het laatste congres (NIOC2025, gehouden op donderdag 27 maart 2025 jl. en georganiseerd door Hogeschool Windesheim). Bij elkaar zo'n 1500 bijdragen!

We roepen je op, na het lezen van het document dat door jou is gedownload, de auteur(s) feedback te geven. Dit kan door je te registreren als gebruiker van de NIOC kennisbank. Na registratie krijg je bericht hoe in te loggen op de NIOC kennisbank.

Het eerstvolgende NIOC vindt plaats op 18 maart 2027 in Arnhem en wordt georganiseerd door HAN University of Applied Sciences.

Reacties over de NIOC kennisbank en de inhoud daarvan kun je richten aan de beheerder:

R. Smedinga kennisbank@nioc.nl.

Vermeld bij reacties jouw naam en telefoonnummer voor nader contact.

Het programmeren van lussen, een blijvend probleem?

N.M. van Diepen, H. Koppelman
Faculteit Informatica, Universiteit Twente, Enschede

Inleiding

Een belangrijke besturingsstructuur in de meeste programmeertalen is de herhaling of iteratie. In vrijwel iedere inleidende programmeercursus zal er dan ook aandacht aan besteed worden. Omdat cursisten dikwijls fouten maken op dit terrein laten we enkele benaderingen de revue passeren. Wij zullen daarbij ondermeer putten uit onze ervaring in het geven van programmeeronderwijs aan niet-informatici.

Formeel of intuïtief

Een onderscheid dat te maken is tussen de diverse strategieën voor het presenteren van herhalingen is de mate van formaliteit in de aanpak. Dit varieert van het afleiden van een programmalus vanuit een gewenste eindbetrekking tot het opdienen van de syntaxis en het (informeel) aanduiden van de (dynamische) semantiek van een herhalingsstatement. Tussenmogelijkheden zijn bijvoorbeeld het correct bewijzen van lussen met behulp van asserties en het aangeven van de invariante betekenis van variabelen.

Voor informatici of niet-informatici

Bij het geven van een (programmeer)cursus is het van belang nauwkeurig de doelgroep en de doelstellingen in de gaten te houden. Van informatici mag verwacht worden dat zij op zijn minst kunnen redeneren over de correctheid van programma's. Bij programmeeronderwijs voor informatici zal dus ook de aanpak van lussen tamelijk formeel kunnen en moeten zijn. Voor de overige disciplines, die zich met programmeren bezighouden, is het van belang een goede ontwerphouding aan te leren. Het schrijven van correcte lussen moet daar integraal onderdeel van uitmaken.
Wanneer de doelstelling van een cursus is louter het leren van nòg een programmeertaal, dan is bovenstaande uiteraard minder van toepassing.

Informatici

Voor informatici zal een programmeercursus onderdeel uitmaken van een breed vakkenpakket waarvan discrete wiskunde en logica deel moeten uitmaken. Kennis van propositielogica, asserties en bewijstechnieken zijn dan aanwezig. Dit kan dan gebruikt worden bij het correct bewijzen van programma's of programmaonderdelen. Ook programmaconstructie op grond van een gewenste eindbetrekking behoort voor dit type student tot de mogelijkheden en is ook gewenst.

Niet-informatici

Voor niet-informatici, met name technici, is programmeren een hulpvaardigheid, die ten dienste staat van hun vakstudie. Kennis van discrete wiskunde en logica is niet altijd aanwezig, terwijl voor programmeren ook maar een beperkte tijd beschikbaar is. Daarom moet voor deze doelgroep de aandacht niet gericht worden op het correct bewijzen van (met name) lussen. De kostbare tijd moet besteed worden aan het bieden van handvatten voor het ontwerpen van correcte lussen. Er moet dus een leersituatie gecreëerd worden, waarin de lussen 'vanzelfsprekend' correct zijn. Correctheidsbewijzen met behulp van de lusinvariant zijn voor deze doelgroep dan ook volstrekt ongeschikt. Onze ervaring leert dat de vaardigheden, die hiertoe vereist zijn, ontbreken en in het kader van een programmeercursus ook niet snel verworven kunnen worden.

Ontwerpvaardigheden algemeen

Voor het ontwerpen van programma's is het van belang om problemen te decomponeren. De (onafhankelijke) deelproblemen moeten vervolgens gespecificeerd worden. Daarbij mag de implementatie van de oplossing van het deelprobleem nog geen enkele rol spelen. Er moet een strikte scheiding zijn tussen de specificatie (het Wat) en de implementatie (het Hoe). Proceduréle abstractie is dus hoofdvaardigheid.

onderwijsinhoud informatica
onderwijsuitvoering didactiek
onderwijssector hoger onderwijs

Bovendien is het hergebruik van standaardoplossingen, eventueel licht gemodificeerd, een belangrijk programmeerprincipe. Hierbij moet van een probleem herkend worden dat het een instantie is van een standaardprobleem.

Ontwerpvaardigheden met betrekking tot lussen

In de eerste plaats moet bij het programmeren van een lus duidelijk zijn wat het beoogde resultaat is. Het effect van de lus moet gespecificeerd (kunnen) worden. Het idee hierachter is, dat een lus slechts goed geprogrammeerd kan worden wanneer de programmeur precies weet wàt het resultaat moet zijn.
Niet iedere lus levert bij implementatie problemen op. Tellussen (FOR-loops) en lussen waarbij het stopcriterium niet echt afhankelijk is van de te herhalen stappen, zijn niet foutgevoelig. Alleen die LOOPs, waarbij het gaat om het onderscheid tussen eerst-te-onderzoeken/laatst-onderzochte element en tot-aan/tot-en-met, leveren vaak fouten op. Het gaat er dus om die situaties te leren herkennen en hulpmiddelen te bieden voor de problemen die dan optreden. Deze kunnen bestaan uit een combinatie van het aanbieden van standaardschema's voor de oplossing van 'standaardproblemen', het expliciet maken van de (invariante) betekenis van de variabelen, een eenvoudige verificatiemethode en een strategie voor de constructie van lussen.

Slot

Bij het programmeren van lussen kunnen alleen van informatici formele correctheidsbeschouwingen gevraagd worden. Alleen zij bezitten de nodige kennis. Voor niet-informatici is het van belang lastige van eenvoudige lussen te onderscheiden. Bij de lastige lussen moet de aandacht dan gericht worden op procedurele abstractie en op hulpmiddelen om vanuit het gewenste effect een programmalus op te stellen. Hierbij heeft een LOOP-statement zoals in Modula-2 wellicht de meest geschikte structuur. Daarnaast is het van belang om standaardproblemen en standaardpatronen van oplossingen te kennen.