

저자 서문

이 책은 프로그래밍 언어라는 방대한 분야의 입문서입니다. 본서는 최신의 함수형 언어와 객체-지향 언어들 몇몇을 포함한 많은 현대 언어들을 상당히 자세하게 다루면서, 이것과 함께 원리들에 대한 전반적인 서술을 합니다. 여타의 많은 입문 교재와는 달리 이 책은 구현과 연관된 주제에 관한 의미 있는 자료나 프로그래밍 언어의 이론적 기반, 그리고 매우 많은 연습문제들을 포함하고 있습니다. 이러한 모든 특징들로 인해서, 이 교재는 컴파일러 과목이나 프로그래밍 언어의 이론적 연구로 나아가기 위한 유용한 가교 역할을 할 수 있습니다. 그러나 이 책은 구체적으로는 '2001년 미국컴퓨터학회/국제전기전자공학자협회-컴퓨터분과 합동 교과과정 연구단 보고서(2001 ACM/IEEE-CS Joint Curriculum Task Force Report)'에 규정된 프로그래밍 언어론의 필수 항목과 '1978년 미국컴퓨터학회 교과과정 CS8 과목(CS8 course of the 1978 ACM Curriculum)'의 대부분을 다루도록 된, 학부 과정의 고급 프로그래밍 언어 개론 과정에서의 교재로 사용하기 위해 고안된 것입니다.

이번의 개정판을 준비하면서 저의 목표는 언어에 구체적인 내용을 1993년의 초판이 나온 이래 프로그래밍 언어의 인기도와 사용에 있어서의 변화에 맞추고, 특정 분야에서 다루는 내용을 개선 및 확장하고, 예제와 연습문제의 서술과 유용성을 향상시키는 것이었습니다. 그리고 이러한 일을 초판의 구성을 가능한 한 유지하면서 달성하는 것이었습니다.

독자들이 어떤 특정한 언어를 알고 있어야 한다고 요구하지는 않습니다. 그러나 적어도 하나의 언어에 대한 경험은 필수적입니다. 자료 구조나 이산 수학 과목을 통해 획득한 어느 정도의 '수학적인 능숙함'도 또한 기대합니다. 이번 판에서 사용되는 주요 언어들에는 C, C++, Java, Ada, ML, Haskell, Scheme, 그리고 Prolog 등이 포함됩니다; 그 밖의 많은 언어들은 보다 간단히 논의됩니다.

개관 및 구성

대부분의 장들은 일부러 내용을 제한하지 않았음에도 서로 거의 독립적입니다. 설명 특정 장이나 절을 건너뛴다고 하더라도, 본문에서의 상호 참조 덕분에 독자나 강의자는 발생할지도 모를 간극을 메울 수 있을 것입니다.

1장은 후속 장들에서 공부할 개념들을 훑어보고, 전형적인 언어로 쓰여진 간단한 예제들 통해 여러 가지 상이한 언어 패러다임들을 소개합니다.

2장과 3장은 각각 프로그래밍 언어의 역사와 언어 설계의 원칙들을 살펴봅니다. 3장은 어찌면 이 책의 마지막 장의 역할을 해야 했을 것이나, 앞에서 다뤄지면 뒤에 나올 주제에 대한 흥미를 불러일으킨다는 점을 알게 되었습니다.

4장은 BNF, EBNF, 구문 다이어그램 등의 사용을 포함하여 구문을 어느 정도 자세히 다룹니다. (4장의) 짝막한 절 하나에서는 순환적 정의(BNF같은)를 풀어야 될 집합 방정식으로 생각하는 관점을 볼 텐데, 이 관점은 이 책을 통해 반복적으로 나타나는 것입니다. 순환적-하강 과상과 과상 도구들의 사용에 대해서도 하나의 절이 배정되어 있습니다.

5장, 6장, 7장, 8장은 프로그래밍 언어의 중심적인 의미 주제들을 다루는데, 그것들은 선언, 할당, 평가; 의미 함수로서의 심벌 테이블과 실행시간 환경; 데이터 타입과 타입 검사; 프로시저 활성화와 매개변수 전달; 그리고 예외상황과 예외상황 처리 등입니다.

9장은 모듈과 추상 데이터 타입을 살펴보는데, 언어의 메커니즘 및 수식적 혹은 대수적 명세를 포함합니다.

10장, 11장, 12장은 10장의 객체-지향 패러다임을 시작으로 언어 패러다임들을 다룹니다. 10장에서의 개념들을 도입하기 위해 저는 Java를 사용합니다. C++와 Smalltalk은 각각의 절에서 다룹니다. 11장은 함수형 패러다임을 다룹니다. Scheme, ML, 그리고 Haskell 각각을 어느 정도 자세하게 살펴봅니다. 또한 이 장은 람다 산술과 순환적 함수 정의의 이론을 소개합니다. 논리 프로그래밍에 대한 12장은 Prolog에 관한 더욱 자세한 절을 제공하고, 한 절은 OBJ와 같은 수식형 언어에 해당하였습니다.

13장은 정형적 의미론의 주요한 세 가지, 즉 연산적 방법과 표기적 방법 그리고 공리적 방법들을 소개합니다. 이 방법들에 대해 실제적인 맛을 느끼기에 충분한 정도로 상세한 내용을 제공한다는 면에서 이 절은 (프로그래밍 언어론) 입문서로서는 다소 독특한 것입니다.

14장은 주로 Java와 Ada의 예제들을 사용하여 코루틴, 쓰레드, 세마포, 모니터, 메시지 전달과 같은, 프로그래밍 언어에 병렬성을 도입하는 주요 방법들을 다룹니다. 이 장의 마지막 절은 LISP와 Prolog에 병렬성을 도입하려는 최근의 노력을 훑어봅니다.

교재로서의 사용

저는 이 책을 San Jose State University에서 컴퓨터 과학을 전공하는 학부 상급 학생들과 대학원생들을 위한 CS 152 강좌에서 10년 넘게 사용해 왔습니다. 저는 그 과목을 두 가지 완전히 다른 구성으로 가르쳐 왔는데, 그것들은 '원칙' 접근 방식과 '패러다임' 접근 방식 정도로 부를 수 있을 것 같습니다. 한 학기 과정인 경우, 이 방식들의 구성을 제안하자면 다음과 같습니다:

'원칙' 접근 방식 1장, 4장, 5장, 6장, 7장, 8장, 그리고 9장. 만약 시간이 남으면 2장과 3장.

'패러다임' 접근 방식 1장, 10장, 11장, 12장, 13장, 그리고 14장(순서는 딱히 이것을 따를 필요는 없습니다). 만약 시간이 남으면 2장과 3장, 또는 남은 장들에서 선택한 주제들.

두 학기나 두 사분학기(quarter)라면 이 책의 대부분을 다룰 수 있습니다.

각 장의 끝에 있는 연습문제들 일부에 대한 정선된 해답을 www.brookscole.com이나 저자의 웹 사이트인 www.cs.sjsu.edu/faculty/louden에서 찾을 수 있습니다. 이 문제들 중 상당수는 본문에서 논의한 언어로 작성하는 프로그래밍 연습문제(절코 아주 크지는 않은)입니다. 개념적인 연습문제들은 (본문에 다룬) 내용에 대한 이해를 점검하는 단답형부터 다소 긴 서술식 문제와 '사고를 요구하는' 도전적인 질문에까지 걸쳐 있습니다. 독자가 약간만 숙고해 보면 어떤 특정 문제의 잠재적 난이도를 적절히 파악할 수 있을 것입니다. 온라인 해답을 읽음으로써 보다 많은 지식을 얻을 수 있을 텐데, 저는 해답을 본문의 연장이라고 취급해서 때로는 해당 문제를 주는 데 필요한 정도를 넘어서는 추가의 정보를 제공합니다. 가끔은 특정 언어에 관한 문제에 대한 답을 얻기 위해, 독자는 해당 언어의 참조 매뉴얼을 참고하거나 이 책에서 구체적으로 다루지지 않은 언어에 대한 지식을 알 필요가 있을 것입니다. 이 책 전체에 걸쳐서, 저는 코드 예제의 적절한 곳에 줄 번호를 추가하거나 많은 예제의 코

사원 동작을 보여줄 수 있도록 그것에 주 프로그램 드라이버를 추가함으로써 그 유용성을 향상시키려고 하였습니다. 여러 가지 다른 예제들(지면의 제약 또는 기타 이유로 추가 코드를 따뜨린)뿐만 아니라 그리할 (코드) 예제들 전부를 www.brookscole.com이나 앞에 나온 저자의 웹 사이트에서 구할 수 있습니다. 이 웹 사이트들은 또한 이 책에 나온 주요 언어들 전부에 대한 무료이고 다운로드가 가능한 번역기의 링크를 포함하고 있는데, 그것들 중 다수는 제가 예제들을 검사하기 위해 사용한 것입니다. 그 밖의 자료들 역시 그 곳에서 구할 수 있습니다.

1판과 2판간에 바뀐 것의 요약

1판에서 저는 Scheme, ML, Miranda, C++, Eiffel, Smalltalk, Prolog와 같은 상이한 언어 패러다임을 표현하는 다소 덜 알려진 몇몇 언어들뿐만 아니라, C, Pascal, Ada, Modula-2, FORTRAN을 포함하는 가장 잘 알려진 명령형 언어들을 예제로 사용하였습니다. 2판에서의 가장 큰 변화는 예제에서 Pascal과 Modula-2를 거의 다 C, C++, Java로 대체한 것입니다. 9장에서 ADT에 관한 '역사를 살피는' 절을 제외하면 Modula-2는 사라졌습니다; Pascal로 된 일부 예제는 남아 있습니다. 저는 또한 Ada를 매우 많이 사용하게 되었는데, 특히 C/C++/Java에서 잘 표현되지 않는 기능들(예컨대, 부분범위, 배열과 슬라이스, 데이터 타입의 이름 동치)을 위해서 그렇습니다. Java는 객체-지향 프로그래밍 언어에 관한 10장에서의 주요 예제로서 Simula를 대체하였고, Eiffel에 관한 절은 삭제하였습니다. 저는 함수형 언어에 관한 11장에서 ML과 Haskell에 훨씬 더 많은 지면을 할당하였고, 책 전체에 걸쳐 ML 예제를 자유롭게 추가하였습니다. 끝으로 저는 병렬 프로그래밍 언어에 관한 14장에서 병행성의 기본적인 예로서 Java의 쓰레드를 사용하였습니다. 그 밖의 중요한 변화는 다음과 같습니다:

- 저는 제어에 관한 내용 중에서 프로시저와 환경을 따로 떼어 내어, 이제 7장은 제어 표현과 문장을 다루며 새로운 8장이 제어 프로시저와 환경을 다룹니다. 저는 식을 기초의 미론에 관한 5장의 끝에서 7장의 시작으로 옮겼습니다. 대부분의 경우 식을 평가하는 데 있어서는 어떤 형태로든 명시적 혹은 명시적 제어가 있을 수밖에 없으므로, 이 주제는 기타 제어 주제와 잘 어울립니다.
- 저는 심벌 테이블과 함께 5장에 다중적체를 포함시켰습니다. 왜냐하면 다중적체된 식별자들에 관한 모호함을 명확하게 밝히는 일은 본질적으로 심벌 테이블의 과제이기 때문입니다. 이것이 데이터 타입에 관한 6장과의 '이야기 전개 순서' 문제를 일으키지만—타입 시그니처가 다중적체의 모호성 해소에 사용되는 가장 중요한 속성이기 때문입니다—다중적체의 모호성 해소를 이해하기 위해 필요한 데이터 타입 정보의 양은 그리 많지 않으며, 그 내용은 이런 식으로 서술되는 것이 보다 자연스러워 보이기 때문입니다.
- 저는 타입 검사와 함께 6장에 매개변수 다형성을 포함시켰는데, 거기에서는 또한 Hindley-Milner 다형적 타입 검사의 보다 폭넓은 설명을 하였습니다. 매개변수 다형성은 9장에서 Ada의 패키지를, 그리고 10장에서 C++의 클래스 템플릿을 논의하면서 거듭 나타납니다.
- 저는 ADT와 모듈에 대한 9장을 다시 작성하여 모듈을 좀더 강조하였고 제목도 모듈을 포함하도록 변경하였습니다. 이 주제는 간결하게 서술하기에는 만만찮은 것입니다. 왜냐

하면 ADT와 모듈 메커니즘은 아마도 병행 메커니즘을 제외하면 다른 어떤 기능보다도 일반 프로그래밍 언어들 간에 차이가 큰 것이기 때문입니다. 여기에서 저는 ML과 Ada를 주된 예로 사용하고, C++의 네임스페이스와 Java의 패키지에 관한 일부 내용을 추가로 사용했습니다. ADT와 모듈을 표현하기 위한 클래스의 사용은 객체-지향 프로그래밍에 관한 10장으로 마무리했습니다.

- 저는 이 책에서(2장에 있는 잘못된 절을 빼고) 전반적으로 Perl과 JavaScript 그리고 Tcl과 같은 스크립트 언어들을 언급하지 않았습니다. 비록 이러한 언어들의 사용이 특히 웹 응용에 대해 광범위하며 증가세에 있고 학생들의 흥미도 높지만, 저는 아무래도 그 언어들은 이 책을 위해서는 다소 너무 특수-목적용이라고 생각합니다. 그러나 관심을 가진 강의자가 언어의 어떤 주요 기능이든, 이 언어들로 작성된 그것의 예제를 제시하는 것을 막을 생각은 없습니다.
- 저는 또한 Visual Basic이나 여러 가지 JavaBean 도구와 같은 '시각적' 언어 또는 컴포넌트 조립 도구는 아무 것도 다루지 않았습니다. 제 생각에는 이 '언어들'은 GUI나 소프트웨어 공학 강좌에서 보다 잘 배운다고 봅니다. 마찬가지로 XML과 SGML 그리고 HTML과 같은 다양한 마크업 언어들은 단지 지나가며 언급합니다.

감사의 글

저는 여러 해에 걸쳐 의견과 수정 그리고 제한을 이메일로 보내준, 너무 많아서 언급할 수 없는 그 모든 분들에게 사의를 표하고 싶습니다. 특히 이 판의 감수자들에게 많은 유용한 제안과 의견에 대한 감사를 드립니다: Arizona State University의 Leonard M. Faltz, College of St. Elizabeth의 Jesse Yu, Abilene Christian University의 Mike Frazier, ITESM Campus Estado de Mexico의 Ariel Ortiz Ramirez, Indiana State University의 James J. Ball, Grinnell College의 Samuel A. Rebelsky, 그리고 University of Iowa의 Arthur Fleck 등입니다.

저는 또한 병행성에 대한 14장을 읽고 의견을 준 것에 대해 Tel-Aviv University의 Eran Yahav에게, 개관하는 장인 1장과 함수형 프로그래밍에 대한 11장에 의견과 제한을 준 것에 대해 University of Texas의 Hamilton Richards에게 감사합니다.

San Jose State University에서 저의 CS 152 강의를 수강한 많은 학생들이 이 판과 초판에 적·간접적으로 기여한 것에 대한 고마움을 간직하고 있습니다: San Jose State University의 제 동료들 중에서 초판의 각 장을 읽고 의견을 준 Michael Beeson과 Cay Horstmann 그리고 Vinh Phat에 감사하고; 초판의 감수자들인 American River College의 Ray Fanslau, Ohio University의 Larry Irwin, California Polytechnic State University의 Zane C. Motteler, University of Illinois-Urbana의 Tony P. Ng, Shippensburg University of Pennsylvania의 Rick Ruth, 그리고 University of North Texas의 Ryan Stansifer에게 감사합니다.

물론 이 책에 있는 결점과 오류는 전적으로 저의 책임입니다. 독자들로부터의 오류 보고와 기타 의견은 louden@cs.sjsu.edu로 기쁘게 받겠습니다.

저는 특히 길으로 보기에 끝이 없을 것 같은 개정 작업 동안에 보여준 격려와 헌내에 대해 Brooks/Cole 사의 컴퓨터 과학 담당 편집자인 Kallie Swanson에게 감사합니다. 제일 처음 제가 이 책을 써야 한다고 확신을 갖게 해준 Marjorie Schlaikjer에게는 특별한 사의를

표하는 것이 마땅합니다.

마지막으로 저의 아내 Margreth와 두 아들 Andrew와 Robin에게, 늘 그렇듯이 헌내와 지원 그리고 사랑에 대한 감사의 마음을 전합니다. 그들이 없었다면 이 모든 일의 대부분이 결코 일어나지 않았을 것입니다.