Chapter 2
역사

1950 년대: 첫번째 프로그래밍 언어
1960 년대: 프로그래밍 언어들의 난리법석
1970 년대: 단순성, 추상성, 프로그래밍 언어론
1980 년대: 객체지향 프로그래밍
1990 년대: 인터넷, API, 스크립트 언어

수학 그리고 프로그램

- 프로그래밍 언어
  - 컴퓨터의 진화의 영향
  - 수학 표현의 발전의 영향
- 수학 그리고 프로그램
  - 수학자들은 어떻게 하느냐 보다는 무엇이냐에 더 중점을 둠
  - 무엇: 특성, 정리
  - 어떻게: 알고리즘, 프로시져, 프로그림
  - B.C.600 년 그리스 이전에는 모든 수학이 프로그래밍이었음

바빌론 프로그램의 예

- 아주 오랜 옛날, 연산이란 게 있었음
  - 바빌론 스타일 표현 [Knuth 1972]

    네모난 저수지가 있는데.
    가로 길이와 높이는 같다.
    그 저수지를 만들려고 판 흙들이 있는 데.
    이 흙들의 부피는 120이다.
    저수지의 가로 길이는 5다. 저수지의 세로 길이는?
    // 너무 평이하므로 번역할 필요 없음
    Add 1 to 5, getting 6.
    Divide 6 into 120, obtaining 20.
    Divide 5 into 20, obtaining the width, 4.
    This is the procedure.

바빌론 프로그램의 추가 설명

- 기하학적인 해결책을 보이고 있음

  Area + Volume = T (120 in the example)

  Area = length × width

  Volume = length × width × height

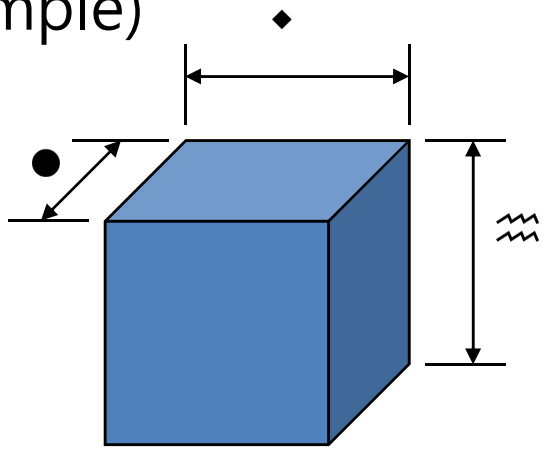
  height = length

  In code:

  ❄ = ● * ◆ + ● * ◆ * ●

  Solving for the width:

  ◆ = ❄ / (● + ● * ●) = T / (1 + ●) / ●



초기의 역사

- 해석 엔진
  - 찰스 배비지 (1830s-40s)
  - 기계적 컴퓨터
  - 입출력 방법: 펀치 카드
- 최초의 프로그래머 (여자임)
  - 어거스타 에이다 러브레이스 백작 부인(Ada Augusta, Countess Lovelace)
  - 시인 바이런의 딸
  - 프로그래밍 언어에서 사용되는 중요한 개념인 루프, 점프, IF문과 같은 제어문, 서브루틴에 관한 개념 고안

1950년대

- 기초적인 프로그래밍 언어
  - 기계어 코드
  - 어셈블리어: 심볼과 니모닉은 있으나 여전히 기계 종속적임 **(중요!)**
- 선구자적인 고수준 언어들
  - 임퍼러티브 언어
    - 포트란 (FORTRAN)
    - 코볼 (COBOL)
    - 알골 (Algol60)
  - 함수 언어
    - 리스프 (LISP)
    - APL

FORTRAN (중요!)

- FORmula TRANslation
- John Backus, IBM (1954~1957)
- 최초의 고수준 프로그래밍 언어
- 목적 코드의 효율화가 궁극적 목적
- 특징들
  - 배열
  - 인덱스 변수에 의해 루프 제어
  - if 문에 의한 브랜치 (branch)
- 아직도 쓰이고 있음 (Fortran I, II, III, IV, Fortran66, Fortran77, Fortran90, Fortran95)

코볼 (COBOL) (중요!)

- Common Business-Oriented language
- 그레이스 하퍼 (Grace Hopper), 미 해군, 미국 국방성 (1959~1960)
- 산업계에서는 많이 애용되었지만, 학계에서는 무시당해왔음 (장황한 문법 때문인데, 사실 그 당시 가장 영어에 가까운 프로그래밍 언어)
- 특징
  - 레코드 구조
  - 데이터 구조와 실행 블록의 분리
  - "**`picture`**"를 이용한 포맷팅

Algol60

- ALGOrithmic Language
- 국제 위원회에 의해 만들어짐 (1958~1960)
- 알고리즘 구현을 위한 언어
- 특징
  - 구조화된 문장
  - begin-end 블럭
  - 변수의 형 (type) 선언
  - 재귀 호출 → 스택 기반 실행 환경
  - 값 기반 호출
- 문법은 BNF (Backus-Naur Form)에 의해 정의

리스프 (LISP) (중요!)

- LISt Processor
- 존 매카시(John McCarthy), MIT (1958~1960)
- 특징
  - 인공지능에서의 심볼 프로그래밍
  - 하나의 데이터 구조, S 표현식(S-expression)
  - 쓰레기 수집 (garbage collection)
  - 재귀 호출 (recursion)
- 리스프의 방언들이 여전히 쓰이고 있음 (예를 들어, Scheme, Common Lisp)

APL

- A Programming Language
- K. Iverson, Harvard, late 1950s
- Transferred to IBM, early 1960s
- 특징
  - 배열과 매트릭스 연산
  - 초기의 시분할 시스템에서 사용됨 (IBM 360)
  - 단점: 수학 심볼을 위해서 특별한 터미널이 필요

1960년대

- 프로그래밍 언어들이 마구 등장하기 시작 (50쪽 그림 2.1)
- 특화된 목적의 프로그래밍 언어들이 등장
- 주목할만한 언어들
  - PL/I: 범용 언어의 시조
  - Algol68: 직교적 설계 (orthogonal design)
    - 직교성 – 언어가 문맥에 따라 다른 의미를 가지지 않는 것
    - 직교성이 안되는 경우의 예 – C에서의 배열
  - 다른 특화된 언어들
    - SNOBOL
    - Simula67
    - ISWIM
    - BASIC

PL/I

- IBM 360 시리즈를 위한 언어 (1963~1964)
- 다른 모든 프로그래밍 언어를 아우르기 위한 꿈으로 시작된 범용 언어 → 실패함 !
- 설계의 목적
  - FORTRAN, COBOL, Algol60의 모든 특성을 망라함
  - 병행성(concurrency)과 예외 처리(exception handling)
- 단점
  - 언어 번역기를 만들기 어렵다
  - 배우기 어렵고, 에러를 내기 쉽다

Algol68

- 국제 위원회 (1963~1968)
- 특징
  - 완전히 직교적인 언어
  - Algol60에 새로운 특징들이 추가됨
- 단점
  - 너무 복잡함: 거의 읽을 수 없는 난해한 언어
    참조 매뉴얼 → 실패 !

SNOBOL

- StriNg Oriented symBOlic Language
- D. Farber, R. Griswold, and I. Polonsky of Bell Labs (1962~3)
- 스트링을 처리하는 언어
- 강력한 패턴 매칭 능력을 가지고 있음

Simula67

- Kristen Nygaard and Ole-Johan Dahl, Norwegian Computing Center (1965~7)
- 시뮬레이션에 사용되는 언어
- 객체지향 언어에 필수적인 클래스 (class) 개념 도입

ISWIM

- If you See What I Mean
- Peter Landin (1966)
- 현대의 함수 언어의 조상 (say Haskell, ML)
- 특징
  - 느긋한 계산법(lazy evaluation) (중요!)
    - 결과값이 필요할 때까지 계산을 미룸
    - 1. 지연 계산법 – 함수형 언어에서 많이 사용됨
    - 2. 최소 계산법
  - 오프 사이드 규칙 (중요!)
    - 인덴테이션에 의해 변수의 영역이 결정됨
    - 최근의 경우 이런 대표적인 언어는 파이썬(Python)임

BASIC (중요!)

- Beginners All-purpose Symbolic Instruction Code
- John Kemeny and Thomas Kurtz at Dartmouth College (1964)
- 시분할 시스템을 위한 언어
- 나중에 마이크로 컴퓨터에서 채택되었음
- ANSI Standard “minimal BASIC” (1978)
- ANSI full Standard BASIC (1988)

1970년대

- 단순함과 추상화
- 언어의 복잡함으로 인해, PL/I 과 Algol68 가 실패했었기 때문임
- 주목할만한 언어들
  - 가장 유명한: Pascal, C
  - ADT (추상 데이터 형) 연구: CLU, Euclid, Mesa
  - 규제되는 LISP: Scheme
  - 다른 것들: ML (Milner, 1978~), FP (Backus, 1978), Prolog (Colmerauer 1972~1982)

파스칼 (Pascal) (중요!)

- Niklaus Wirth (1971)
- Algol68의 기본 개념들만 강조
- 작고, 단순하고, 효율적이고, 구조적인 언어
- 프로그래밍을 가르칠 때 교육용으로 많이 사용되었음
- 다음의 중요한 특성들이 빠짐
  - 분리 컴파일
  - 스트링 다루기
  - 확장가능한 입출력 기능
  - [Kernighan 1981, “Why Pascal is Not My Favorite Programming Language”] 참고

C (매우 중요!)

- 벨 연구소의 데니스 리치(Dennis Ritchie) (1972)
- 형 (type systems) 체계를 단순화함
- 단순한 실행 환경 (함수는 nesting이 안됨)
- 중간 수준 언어: 기계에 더 가깝게 다가간 언어
- 운영체제 프로그래밍에 사용됨 (Unix)

1980년대

- 객체 지향
- 중요한 언어들
  - Ada (John Ichbiah, ~1983)
  - Modula-2 (Niklaus Wirth, 1982)
  - Smalltalk (Alan Kay, ~1980)
  - C++ (Bjarne Stroustrup, 1980~1998)

Ada (Ada83)

- John Ichbiah Team (~1980)
- 미 국방성의 프로젝트 언어
- 특징
  - 패키지(package)를 통한 추상 데이터 타입 (ADT) 지원
  - 태스크(task)를 통한 병렬 프로글래밍 지원
  - 매우 크고 복잡한 언어
  - 1980년대의 PL/I
- 1983년에 국제 표준으로 채택되었음

Modula-2

- Niklaus Wirth (1982, revised in 1985 and 1988)
- 특징
  - 모듈(module): Ada의 패키지 개념
  - 코루틴(coroutine): 제한적으로 병행성 지원
- 교육용 언어 파스칼을 대체함
- 큰 소프트웨어 프로젝트에는 거의 사용되지 않음

Smalltalk (Smalltalk-80) (중요!)

- 제록스 파크(Xerox Corporation's Palo Alto Research Center)의 Alan Kay, Dan Ingalls, et al. at (1972~80)
- 특징
  - Simula 언어의 클래스 (class) 개념이 더 강화됨
  - GUI 윈도우와 마우스를 통한 **환상적인** 프로그래밍 환경 → Mac OS → Windows
- 단점
  - 표시 방법이 일반적이지 않음
  - 효율적인 구현이 어려움

C++ (중요!)

- 벨 연구소의 비야네 스트롭스트룹 (Bjarne Stroustrup)
- C with classes
- 효율적인 객체지향 언어
- 대부분의 플랫폼에서 사용 가능
- 1998 ISO 표준 C++ 제정
- C++0x – 2009년도 현재 개발 중인 새로운 표준

1990년대:이제 기술이 만개하였다

- 거대한 프로그래밍 라이브러리와 API들
- Java (James Gosling, 1995): 이미 만들어진 API들과 함께 등장한 최초의 언어
- Ada 의 라이브러리들도 많아짐(Ada95).
- 시스템 프로그래밍 분야도 커짐: 많은 스크립트 언어들이 도래함 (Perl, Tcl, Javascript, VBScript, Python)
- 함수 언어들도 많아짐:
  - ML (Milner, 1978~1988)
  - Haskell (Hudak, Peyton-Jones, Wadler, 1989~1998)

과연 미래는 어떨까?

- C# (2000): 자바의 자리를 바꿀 수 있을까?
- 5~10년 뒤의 자바는 어떠할까? (대부분의 웹 기반이 아닌 프로그램들은 여전히 C++나 C로 구현되어 있음.)
- 새로운 언어가 등장할까?
- 프롤로그(Prolog)는 어떻게 될까?
- 미래를 예측하는 게 가능하기나 할까?

알아두면 좋은 언어들

- C
- C++
- Java
- C#
- Python
- Ruby