Introduction

Jo, Heeseung

이 장의 내용

컴퓨터 시스템

프로그램 실행 원리

소프트웨어와 소프트웨어의 역할

컴퓨터의 자료 표현

프로그래밍 언어의 종류와 특징

프로그램 개발

컴퓨터 시스템

컴퓨터 시스템

- 컴퓨터 시스템은 하드웨어와 소프트웨어로 구성

컴퓨터 하드웨어 구성

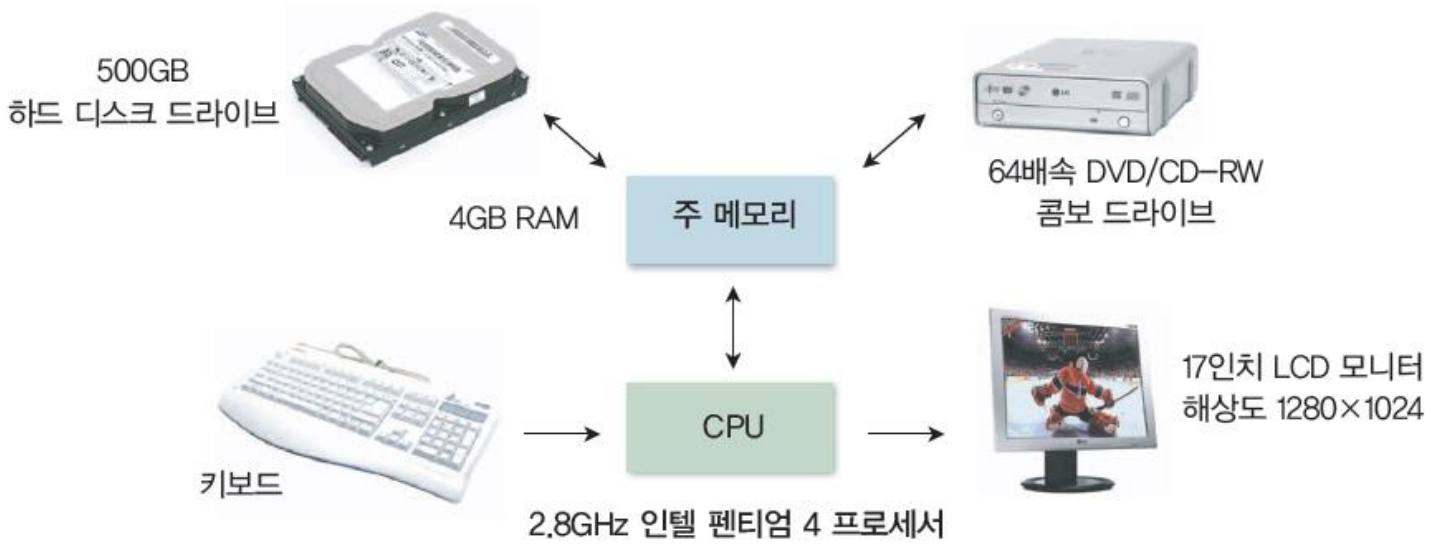

그림 1.1 컴퓨터 시스템 구성과 하드웨어 사양 예

컴퓨터 하드웨어 구성 요소

중앙처리장치(Central Processing Unit; CPU)

- 프로그램의 명령어들을 수행하는 컴퓨터의 두뇌

주 메모리(Main Memory)

- 프로그램과 데이터를 저장하는 휘발성(volatile) 기억장치

보조 기억장치(Secondary Storage)

- 소프트웨어를 비교적 영구적으로 저장하는 비휘발성 기억장치
- 하드 디스크 드라이브(HDD), 솔리드 스테이트 드라이브(SSD)
- 64배속 DVD/CD-RW 콤보(combo) 드라이브

입출력 장치(Input/Output Device)

- 키보드, 마우스, 모니터

프로그램 실행을 위한 구성 요소간 정보이동

프로그램 적재(loading)

- 하드 디스크에 저장된 프로그램을 주 메모리에 읽어 들임

프로그램 실행(execution)

- CPU는 주 메모리로부터 프로그램의 명령어들을 하나씩 읽어 들여서 그 명령어를 한 번에 하나씩 수행

입력(input)

- 명령어가 사용하는 데이터도 역시 주 메모리에 저장
- 보조 메모리 혹은 키보드 같은 입력 장치를 통해 입력

출력(output)

- 프로그램은 실행되는 동안 필요에 따라 모니터와 같은 출력 장치에 정보를 출력

메모리 위치 및 값 저장

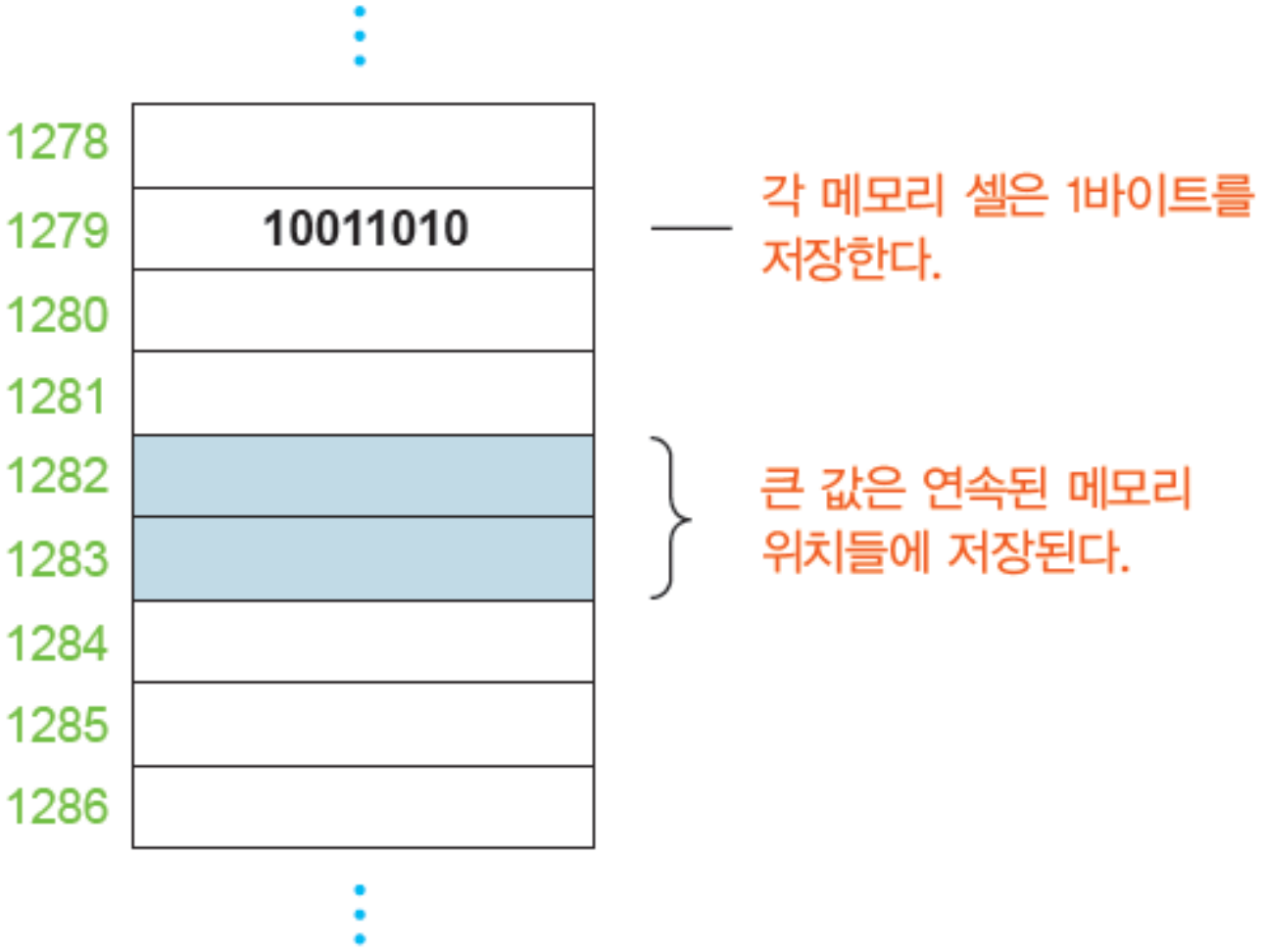

그림 1.2 메모리 위치와 값 저장

이진 저장 단위

단위	기호	2^n 바이트	바이트 수
킬로바이트	KB	2^{10}	1,024
메가바이트	MB	2^{20}	1,048,576
기가바이트	GB	2^{30}	1,073,741,824
테라바이트	TB	2^{40}	1,099,511,627,776

표 1.1 이진 저장 단위

중앙처리장치(CPU)

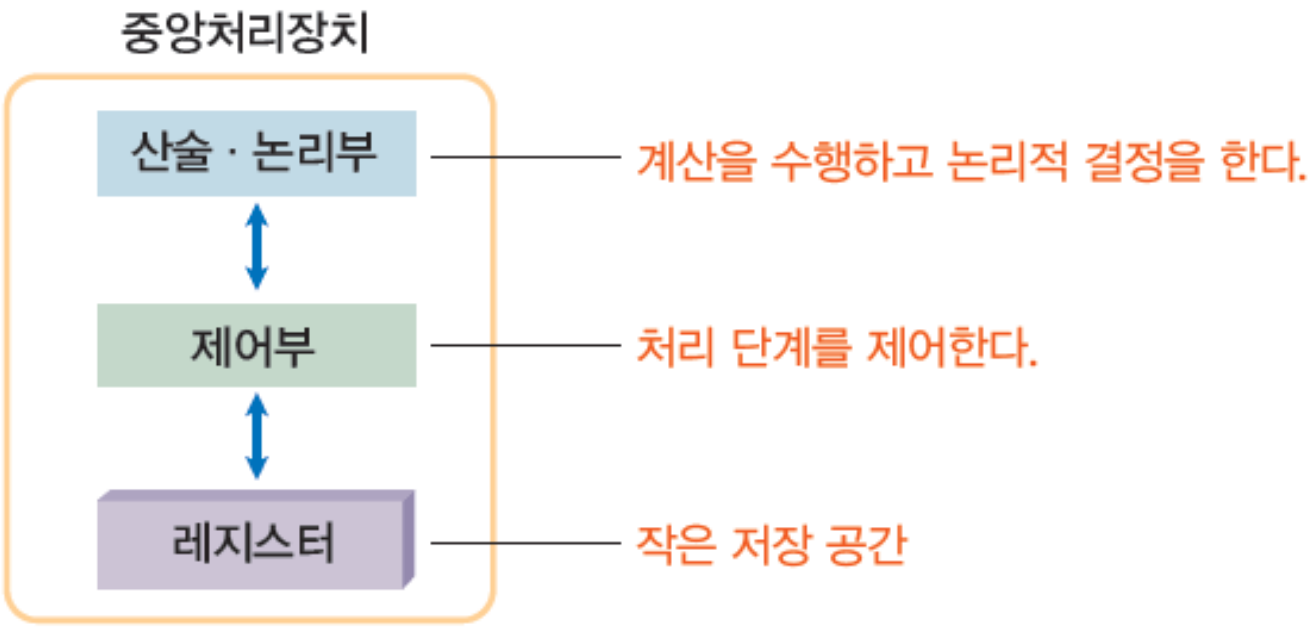

그림 1.3 중앙처리장치의 구성 요소

프로그램 실행 원리

프로그램 실행 원리

폰 노이만 구조(von Neumann architecture)

- 프로그램 내장 방식(stored program)의 컴퓨터

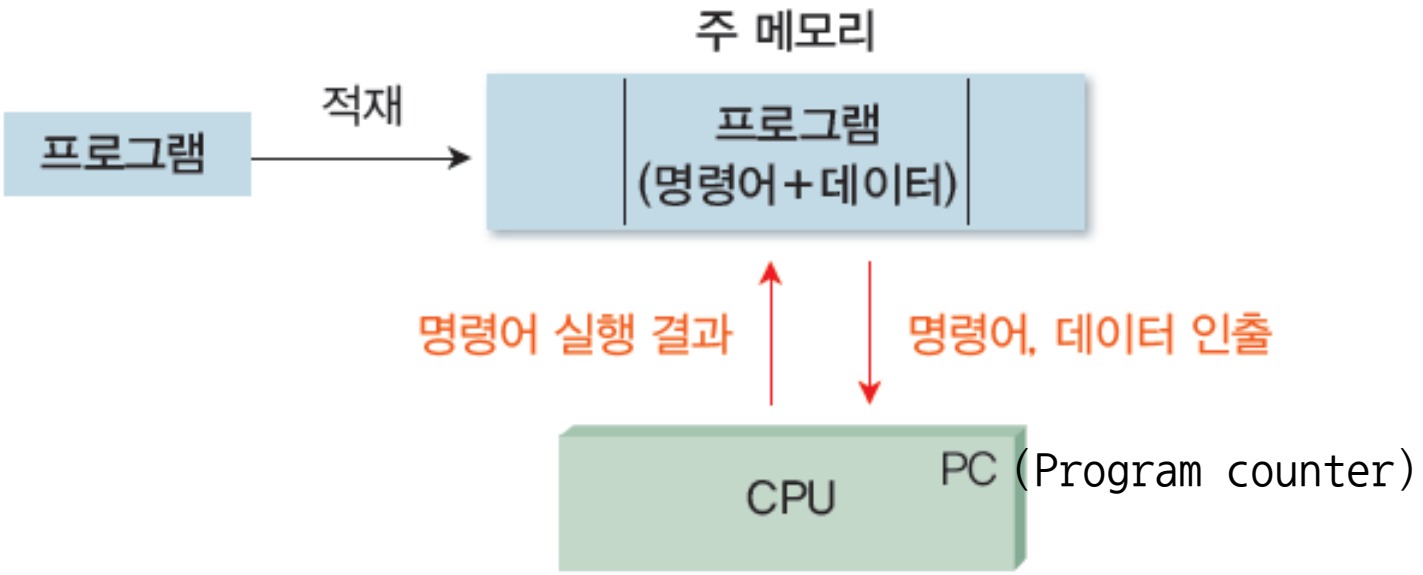

그림 1.4 폰 노이만 컴퓨터 구조

폰 노이만 컴퓨터의 프로그램 내장 방식

프로그램 내장 방식(stored program)

- 명령어와 데이터로 구성된 프로그램을 주 메모리에 적재하고
- CPU가 순차적으로 실행한다는 개념

CPU는 인출-해석-실행(fetch-decode-execute) 주기 반복

- CPU는 주 메모리 내에 저장된 명령어를 한 번에 하나씩 읽어 들여 해석하고 실행

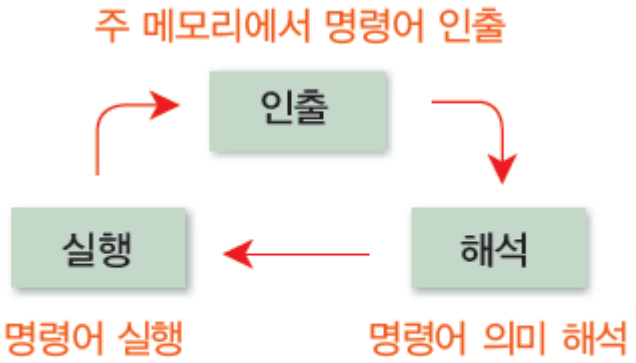

그림 1.5 인출 – 해석 – 실행 주기

컴퓨터의 자료 표현

컴퓨터의 자료 표현

이진수

- 컴퓨터는 이진 값으로 표현된 정보만 저장하고 처리 가능
- 컴퓨터에서는 한 비트가 0과 1을 표현
- N 비트는 최대 2^N개의 경우들을 표현

1비트	2비트	3비트	4비트
0	00	000	0000
1	01	001	0001
	10	010	0010
	11	011	0011
		100	0100
		101	0101
		110	0110
		111	0111
			1000
			1001
			1010
			1011
			1100
			1101
			1110
			1111

표 1.2 이진수

이진수와 십진수

십진수

- 열 개의 숫자(0에서 9)를 이용하여 값을 표현
- 십진수의 각 자리에는 자릿값이 존재
- $182 = 1 \times 10^2 + 8 \times 10^1 + 2 \times 10^0$
  $= 1 \times 100 + 8 \times 10 + 2 \times 1$

이진수

- 두 개의 숫자(0과 1)를 이용하여 값을 표현
- 이진수의 각 자리에는 자릿값이 존재
- $1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
  $= 1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$
  $= 13$

십진수를 이진수로 변환 1

예 $2^4(=16) < 27 < 2^5(=32)$

- $27 = 1 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$
  $= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
  $= 11011_2$

변환 방법

- 27을 $2^4(=16)$로 나누면 몫은 1, 나머지는 11
  첫 번째 비트는 1이고, 11은 나머지 4 비트들로 표현
- 11을 $2^3(=8)$으로 나누면 몫은 1, 나머지는 3
  두 번째 비트는 1이고, 3은 나머지 3 비트들로 표현
- 3을 $2^2(=4)$로 나누면 몫은 0, 나머지는 3
  세 번째 비트는 0이고, 3은 나머지 2 비트들로 표현
- 3을 $2^1(=2)$로 나누면 몫은 1, 나머지는 1
  네 번째 비트는 1이고, 1은 나머지 1 비트로 표현

십진수를 이진수로 변환 2

마지막 비트부터 생각

- $27 = 1 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$
  $= (1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0) \times 2 + 1$

- 2로 나누어 보자

$$
\begin{array}{rl}
2\,)\,27 & \\
2\,)\,13 & \text{——— } 1 \\
2\,)\,\ 6 & \text{——— } 1 \\
2\,)\,\ 3 & \text{——— } 0 \\
\ \ 1 & \text{——— } 1
\end{array}
$$

그림 1.6 십진수를 이진수로 변환

16진수(hexadecimal)

- 기수 16인 수 체계
- 0, …, 9, A(10), B(11), C(12), D(13), E(14), F(15)
- $2AC_{16} = 2 \times 16^2 + A \times 16^1 + C \times 16^0$
  $= 2 \times 256 + 10 \times 16 + 12 \times 1$
  $= 684$

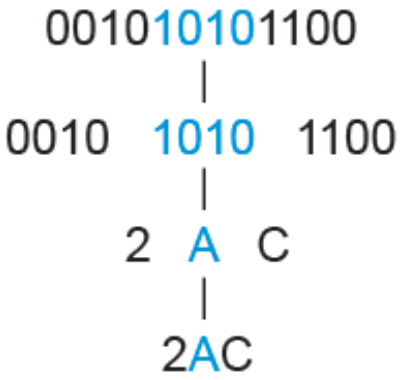

그림 1.7 이진수와 16진수의 관계

소프트웨어

소프트웨어

시스템 소프트웨어(system software)

- 컴퓨터 시스템 효율적인 운영과 관리를 위한 소프트웨어
- 운영 체제, 컴파일러, 디버거, 유틸리티 프로그램

응용 소프트웨어(application software)

- 시스템 소프트웨어가 아닌 거의 모든 소프트웨어를 지칭하는 포괄적인 용어
- 문서 작성 S/W, 데이터 관리 S/W, 스프레드시트,그래픽 S/W, 웹 관련 S/W, 통계 S/W, 게임 S/W 등

소프트웨어

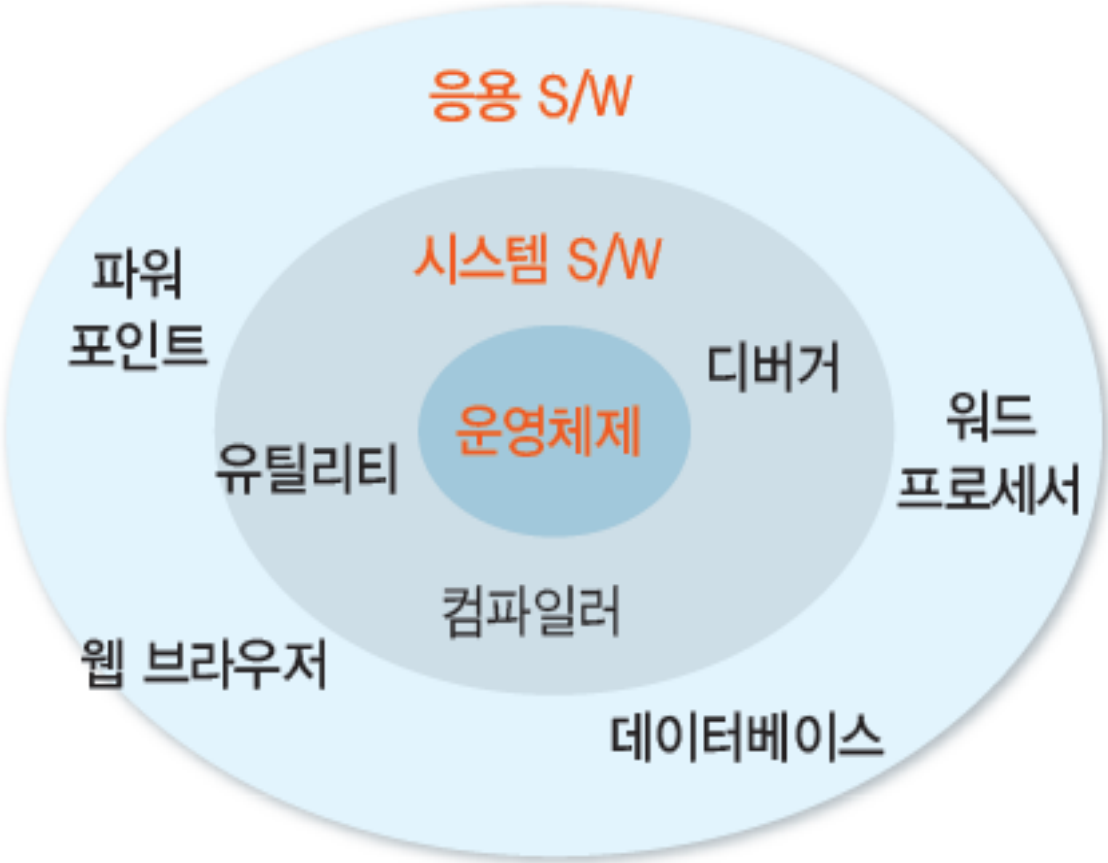

그림 1.8 소프트웨어의 종류

프로그래밍 언어

프로그래밍 언어

프로그램을 작성하기 위한 언어

- 사람이 컴퓨터에게 시키고 싶은 내용을 표현하기 위한 표기법
- 발전 단계에 따라 기계어, 어셈블리어, 고급언어로 분류

기계어

- 이진수 코드로 CPU 종류마다 고유의 기계어

```
1001   0001   0001 위치의 값을 누산기에 저장하라
1100   0010   누산기에 0010 위치의 값을 더하라
1010   0011   누산기의 값을 0011위치에 저장하라
```

어셈블리어

- 기계어의 이진수 코드를 기호화 코드(mnemonics)로 대치한 것

```
LOAD   Y      Y의 값을 누산기에 저장한다
ADD    Z      누산기에 Z의 값을 더하라
STORE  X      누산기의 값을 X에 저장하라
```

프로그래밍 언어

고급 언어

- 영어와 비슷한 구문으로 표현되며, 읽고 쓰기가 보다 쉬움
- 프로그래머가 기계의 세부사항을 알 필요가 없음

```
X = Y + Z
```

주요 고급 언어

- FORTRAN, COBOL, BASIC, C, C++, Java 등

주요 고급 언어

FORTRAN(FORmula TRANslation)

- 1957년에 IBM의 John Backus가 개발한 최초의 고급 언어
- 과학계산용 언어로 프로그램을 단순화된 영어 단어와 수학 공식으로 표현

COBOL(COmmon Business Oriented Language)

- 1960년초에 개발된 사무용 프로그램을 개발을 위한 언어
- COBOL로 작성된 많은 급여, 회계 및 기타 업무용 S/W는 지금도 많이 사용됨

BASIC(Beginner's All-purpose Symbolic Instruction Code)

- 초보자가 배우기 쉬운 프로그래밍 언어로 개발
- 원래 학생들에게 프로그래밍 언어를 가르치기 위해 개발
- 컴퓨터 업계에 소개되면서 비중 있는 프로그래밍 언어가 됨

주요 고급 언어

C 언어

- AT&T 벨 연구소의 Dennis Ritchie가 개발한 시스템 프로그래밍용 언어
- 1970년대 UNIX라는 운영 체제 개발을 위하여 개발된 언어
- 고급 언어이면서도 저급 언어(하드웨어에 가까운 언어)의 특성을 가지고 있음

C++

- AT&T 벨 연구소의 B. Stroustrup에 의해 개발
- C 언어의 기능을 확장하여 만든 객체 지향 프로그래밍 언어

Java

- 1990년대에 선 마이크로시스템 사의 James Gosling이 개발
- 인터넷 환경을 위해 개발된 객체지향 언어로 이동 코드실행 지원
- Java 프로그램은 한번 작성되면 어느 플랫폼에서나 실행 가능

소프트웨어 개발

소프트웨어 개발과정

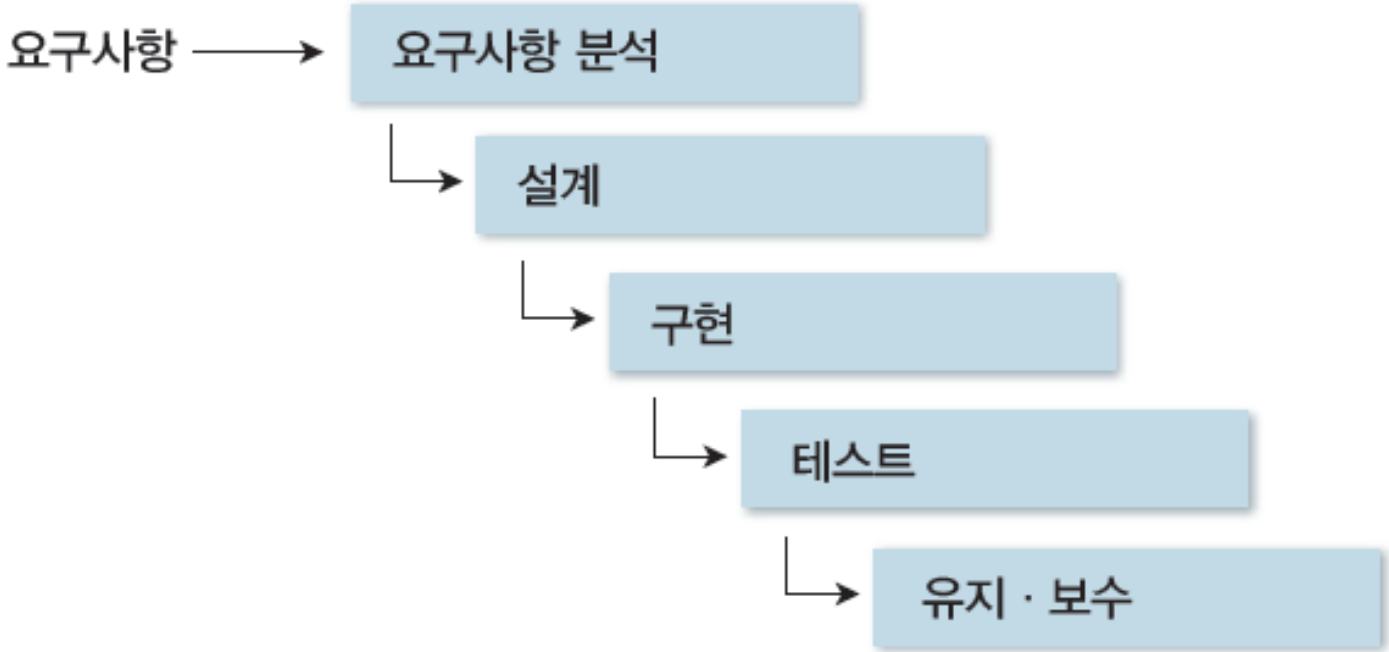

그림 1.9 소프트웨어 개발 과정

소프트웨어 개발과정

요구사항 분석(requirement analysis)

- 사용자의 요구사항 즉 소프트웨어가 해야 할 일을 파악하는 단계
- 사용자는 요구사항을 명세서(specification) 형태로 제시

설계(design)

- 요구 사항을 구체화하는 단계로 소프트웨어의 구조를 결정
- 파악된 문제를 해결하는 알고리즘(algorithm)을 흐름도(flow chart)나 의사코드(pseudo code) 형태로 표현

구현(implementation)

- 설계된 내용을 구체적인 프로그래밍 언어로 작성하는 단계
- 프로그램을 작성하는 것을 코딩(coding)한다고 함

소프트웨어 개발과정

테스트(testing)

- 프로그램의 오류를 찾아내고 이를 수정하는 단계
- 프로그램 오류는 컴파일시간 오류, 실행시간 오류, 논리 오류 등이 있음

유지보수(maintenance)

- 개발된 소프트웨어를 실제 상황에서 운영하면서 상황 및 변화된 요구에 따라 소프트웨어를 적절하게 수정하는 단계

프로그램 작성 및 실행

프로그램 개발 과정을 돕기 위한 소프트웨어 도구

- 편집기, 컴파일러, 디버거, 인터프리터 등
- 통합개발환경(IDE): 이들을 통합하여 제공하는 개발환경

프로그램 작성 및 실행 과정

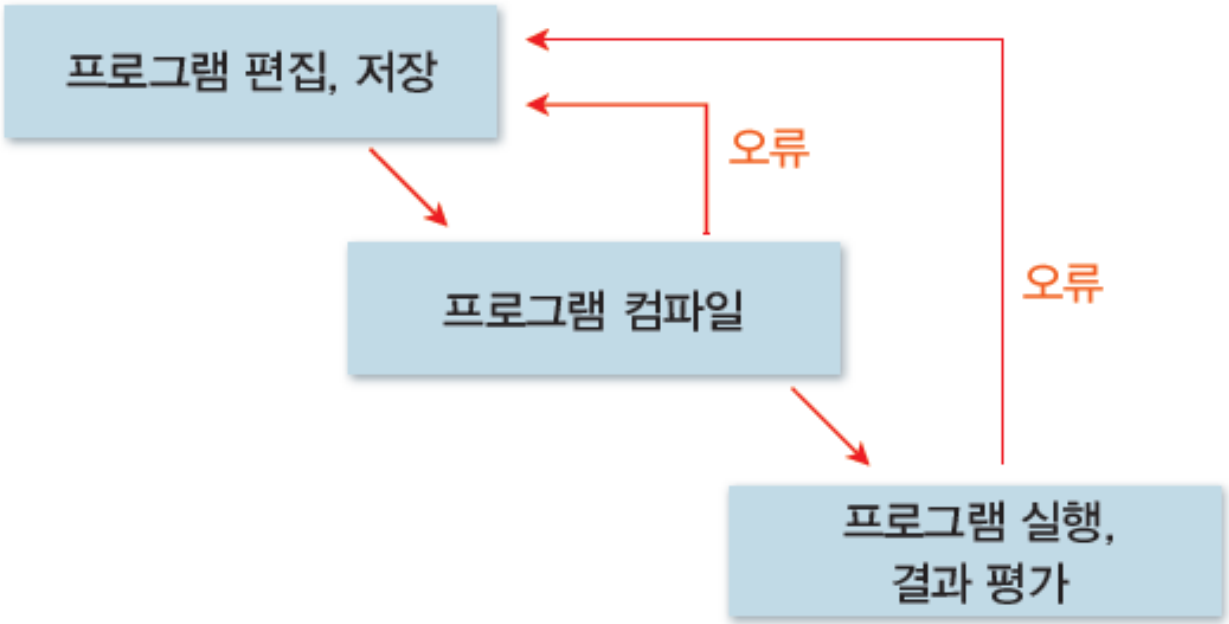

그림 1.10 프로그램의 편집과 실행

오류의 종류

컴파일 오류(compile error)

- 컴파일 과정에서 생긴 오류
- 컴파일러는 프로그램의 구문, 데이터, 의미 없는 문장 등을 검사
- 이런 경우 편집기로 돌아가서 오류를 수정한 후 다시 컴파일 해야 함

논리 오류(logical error)

- 성공적인 컴파일 후 프로그램을 실행하면서 잘못된 결과를 내는 경우
- 원하는 결과를 내도록 프로그램 수정 후 다시 컴파일해서 실행해야 함

실행 오류(run error)

- 프로그램실행 중에 예상치 못한 이유로 비정상적으로 종료
- 예: 어떤 값을 0으로 나누려고 하면 프로그램은 실행을 멈추고 종료

디버깅

디버깅(debugging)

- 프로그램의 결함을 찾고 수정하는 과정을 디버깅이라고 함

QnA

- 디버깅이란 도대체 무슨 말인가요?

컴파일러 및 인터프리터

인터프리터

- 별개의 컴파일 단계 없이 한 번에 한 문장씩 번역(혹은 해석)하여 실행

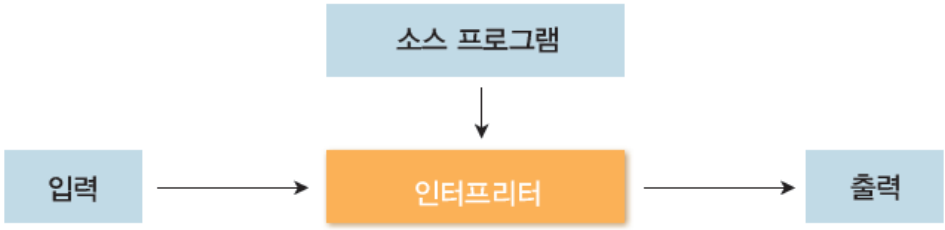

그림 1.12 인터프리터 역할

컴파일러

- 고급 언어로 작성된 코드를 동등한 의미의 저급 언어로 된 코드로 번역하는 소프트웨어

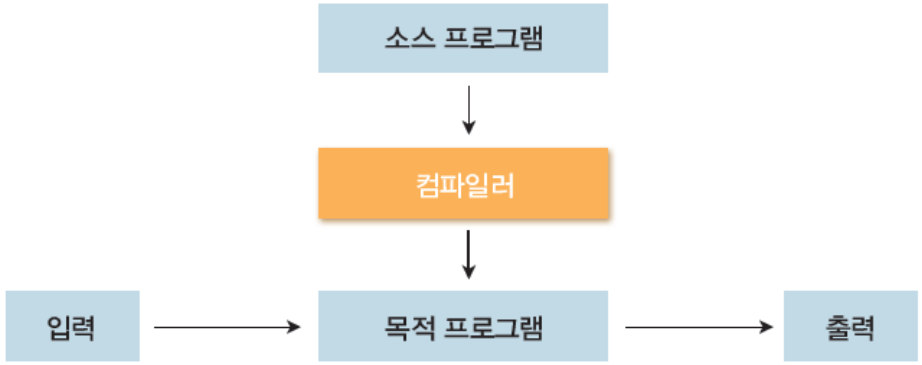

그림 1.11 컴파일러 역할