

제 1 장 프로그래밍언어 개요

- 01 '프로그램'은 무엇일까?
- 02 프로그래밍 언어의 계층과 번역
- 03 왜 C 언어를 배워야 할까?
- 04 프로그래밍의 자료 표현
- 05 소프트웨어 개발
- 06 다양한 '프로그래밍 언어'



학습목표

- ▶ 프로그램과 컴퓨터의 소프트웨어를 이해하고 소프트웨어를 개발하는 프로그래밍 언어에 대해 설명할 수 있다.
 - 일반 용어인 프로그램과 컴퓨터 프로그램의 유사점
 - 하드웨어와 소프트웨어
 - 기계어와 어셈블리어
 - 저급언어와 고급언어
 - 컴파일러와 어셈블러
- ▶ C언어의 중요성과 특징을 이해하고 설명할 수 있다.
 - C 언어의 개발과 역사
 - C 언어의 특징과 중요성
- ▶ 컴퓨터의 자료 표현 방법과 논리 및 문자를 이해하고 설명할 수 있다.
 - 이진수의 표현 방법과 정보의 단위
 - 논리와 문자 표현
- ▶ 소프트웨어 개발 과정과 알고리즘의 표현 방법을 이해하고 설명할 수 있다.
 - 알고리즘의 정의와 기술 방법
 - 소프트웨어 개발 절차
- ▶ 포트란에서부터 엔트리까지 다양한 프로그래밍 언어를 이해하고 설명할 수 있다.
 - 60~70년 초기에 개발된 프로그래밍 언어
 - 비주얼 프로그래밍 언어

스마트폰과 컴퓨터에서의 프로그램

• 프로그램(program)

- 컴퓨터나 스마트폰에서 특정 목적의 작업을 수행하기 위한 관련 파일의 모임
 - 특정 작업을 수행하기 위하여 그 처리 방법과 순서를 기술한 명령어와 자료로 구성
 - 컴퓨터에게 지시할 일련의 처리 작업 내용을 저장
 - 사용자의 프로그램 조작에 따라 컴퓨터에게 적절한 명령을 지시
- 다양한 프로그램
 - 각종 앱이나, MS 워드나 아래한글 등



그림 1-1 일상생활에서 활용하는 스마트폰 프로그램 "앱"과 컴퓨터의 여러 프로그램들

프로그래밍 언어

- **프로그램을 개발하기 위해 사용하는 언어**
 - 사람과 컴퓨터가 서로 의사 교환을 하기 위한 언어
 - 사람이 컴퓨터에게 지시할 명령어를 기술하기 위하여 만들어진 언어
- **다양한 프로그래밍 언어가 개발되어 사용**
 - FORTRAN, ALGOL, BASIC, COBOL, PASCAL, C, C++, Visual Basic
 - Java, Objective-C, JSP, Javascript
 - Python, C#, Go, Swift, Kotlin



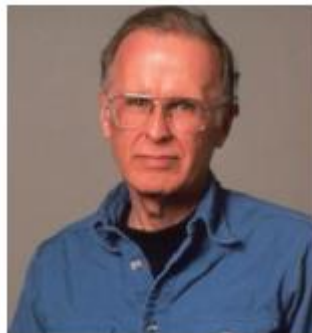
그림 1-5 프로그램을 개발하는 프로그래밍 언어

- 포트란(FORTRAN)

- 최초의 프로그래밍 언어
 - 1950년 중반 IBM에 근무하는 28세의 젊은 과학자인 존 배커스(John Backus)가 개발
- 수식 변환기(FORMula TRANslator)라는 의미의 약자
 - 공학과 과학 분야에서 계산 위주로 사용을 목적으로 개발된 프로그래밍 언어

- 인류 최초의 프로그래머, 오거스타 에이다

- 수학에 천재적 재능, 현대 컴퓨터 프로그래밍 역사의 기원
 - 1833년에 배비지가 고안한 '분석 엔진(Analytical Engine)'에 계산과정을 기술하는 프로그램을 만들어 오늘날 일반적으로 사용하는 컴퓨터의 시조가 되는데 공헌
- 100년 뒤인 1950년대에 업적을 기려 그녀에게 '세계 최초의 프로그래머'라는 호칭이 주어짐
 - 1979년에 미국 국방성에서는 새로 개발한 프로그래밍 언어를 그녀의 이름을 따서 "ADA"라고 명명



하드웨어와 소프트웨어

- 컴퓨터 = HW + SW

- 하드웨어

- 중앙처리장치(CPU: Central Processing Unit)
- 주기억장치(main memory)
- 보조기억장치(secondary memory)
- 입력장치(input device), 출력장치(output device)

- 소프트웨어

- 컴퓨터가 수행할 작업을 지시하는 전자적 명령어들의 집합으로 구성된 프로그램

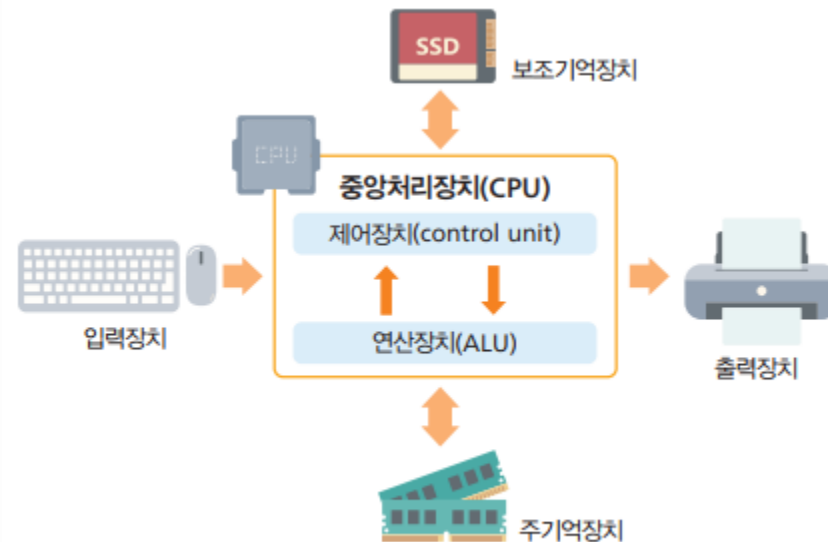


그림 1-7 컴퓨터 하드웨어 구성요소

기계어와 컴파일러

- 기계어(machine language)
 - 컴퓨터가 유일하게 인식할 수 있는 언어
 - 즉 전기의 흐름을 표현하는 1과 흐르지 않음을 의미하는 0으로 표현되는 언어
- 프로그래머와 컴퓨터가 서로 의사 교환
 - 프로그래밍 언어를 사용하는 프로그래머 < = > 기계어를 사용하는 컴퓨터
 - 통역사와 같은 번역기가 필요: 컴파일러
 - 프로그래밍 언어를 기계어로 변환하는 번역기인 컴파일러(compiler)가 필요

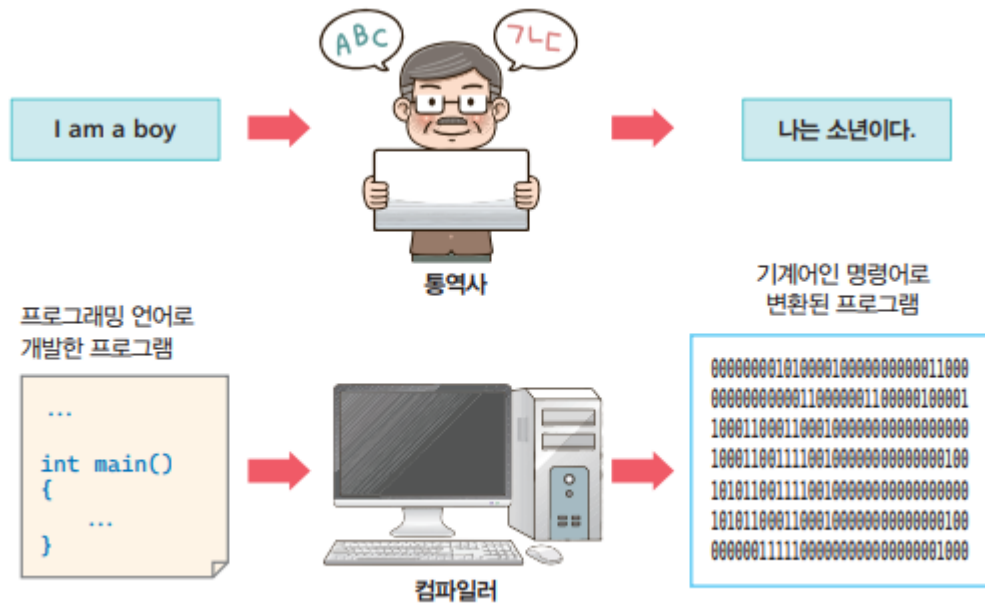
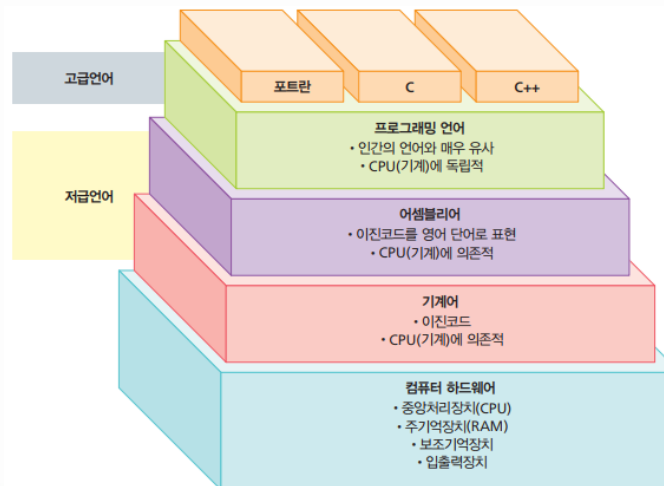


그림 1-10 통역사와 컴파일러

고급언어와 저급언어

- 어셈블리어(assembly language)
 - 기계어를 사람이 좀 더 이해하기 쉬운 기호 형태로 만든 프로그래밍 언어
 - 어셈블리 언어는 기계어보다는 프로그래밍이 훨씬 용이
 - CPU마다 제각각 다름
- 어셈블리 명령어 예
 - LDA(LoaD Address), ADD(ADD), STA(STore Address) 등
 - 명령어를 기호화한 것을 니모닉(mnemonic)이라 함
- 고급언어(High Level Language), 저급언어(Low Level Language)
 - 우리 사람이 보다 쉽게 이해할 수 있도록 만들어진 언어
 - 컴퓨터의 CPU에 따라 달라지며, 특정한 CPU를 기반으로 만들어진 언어



컴파일러와 어셈블러

- **컴파일러(compiler)**

- 고급언어로 작성된 프로그램을 기계어 또는 목적코드(object code)로 바꾸어주는 프로그램

- **어셈블러(assembler)**

- 어셈블리어로 작성된 프로그램을 기계어로 바꾸어주는 프로그램

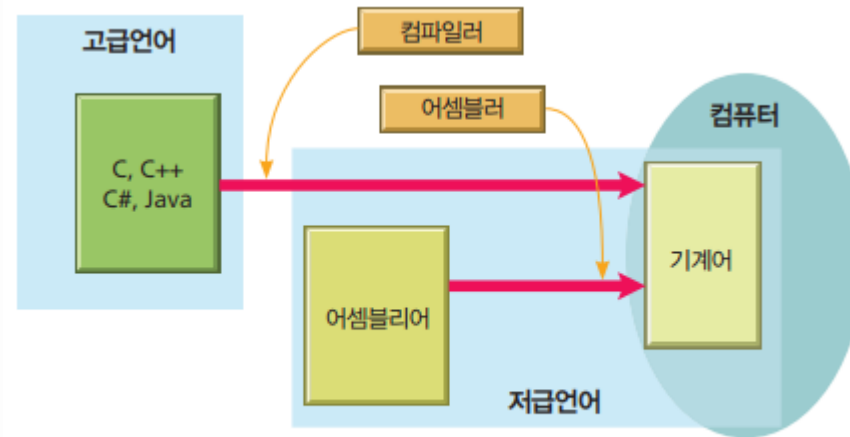


그림 1-15 컴파일러와 어셈블러

C 언어의 역사

- 1972년 데니스 리치(Dennis Ritchie)가 개발

- 미국전신 전화국(AT&T)의 벨 연구소(Bell Lab)에 근무
- 시스템 PDP-11에서 운용되는 운영체제인 유닉스(Unix)를 개발하기 위해 C 언어를 개발
 - 주로 사용 하던 어셈블리 언어 정도의 속도를 내며, 좀 더 쉽고, 서로 다른 CPU에서도 작동되는 프로그래밍 언어로 C 언어 개발
- 영향을 받은 언어
 - 켄 톰슨이 1970년 개발한 B 언어에서 개발된 프로그래밍 언어
 - Algol => CPL(Combined Programming Language) => BCPL(Basic Combined Programming Language) => B => C

- ANSI C

- 표준 C(Standard C)를 지칭
 - 1989년 미국표준화위원회(ANSI: American National Standards Institute)에서 공인

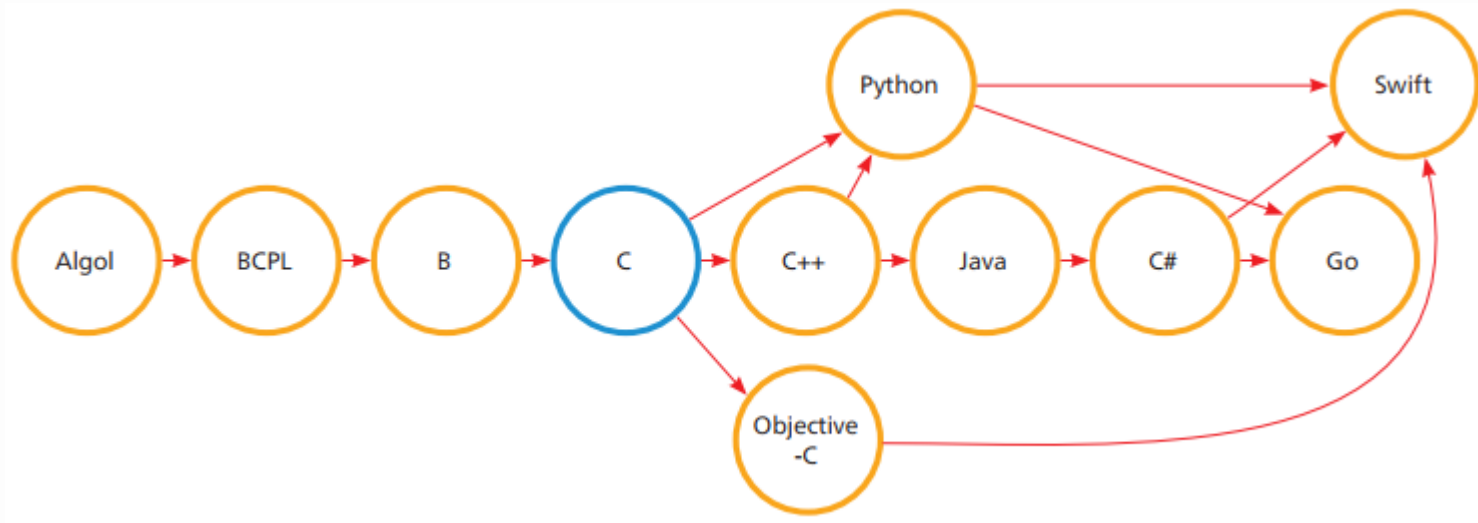


그림 1-17 C 언어에 영향을 미친 언어와 C 언어의 발전

C 이후의 프로그래밍 언어 발전

- **프로그래밍 언어 C++**

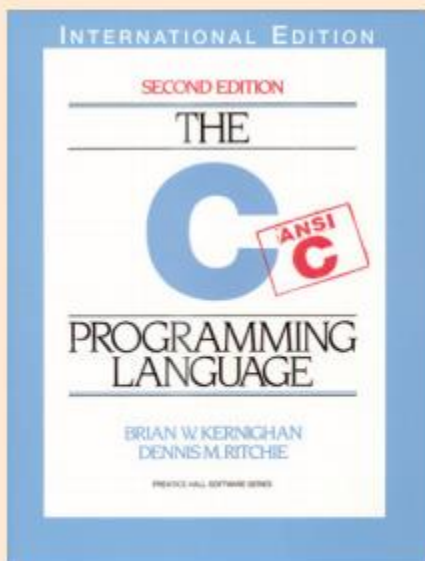
- 1983년 미국전신전화국에 근무하는 비야네 스트롭스트롭(Bjarne Stroustrup) 개발
 - C 언어에 객체 지향 프로그래밍 개념을 확장
- C와 C++
 - 92년, 유럽에서 개발된 파이썬 언어에 영향





TIP 초기 C 언어의 바이블

1978년 프로그래밍 언어 C를 개발한 데니스 리치는 동료 브라이언 커니건(Brian Kernighan)과 함께 너무나 잘 알려진 "The C Programming Language" 라는 책을 출판했다. 흔히 이 책을 K&R C라고 부르는데, 이 K&R C는 ANSI C가 나오기 전까지 C 언어의 표준 문서 역할을 했다. K&R C는 이후 ANSI C 버전으로 2판(cm.bell-labs.com/cm/cs/cbook/index.html)이 다시 출간되었다. K&R C는 초보자에게는 다소 어려운 것이 사실이나 아직도 C의 바이블로 여겨지고 있다.



The C Programming Language, Second Edition

by [Brian W. Kernighan](#) and [Dennis M. Ritchie](#)
 Prentice Hall, Inc., 1988.
 ISBN 0-13-110362-8 (paperback), 0-13-110370-9 (hardback).

- The book is readily available at large bookstores (especially university ones around beginning of term) but tends not to be stocked at the mall because it's treated like a textbook. Ordering it online is easy, for example at [Amazon.com](#), [Barnes & Noble](#), [Borders](#), [Blackwell's](#), or [Faber](#).

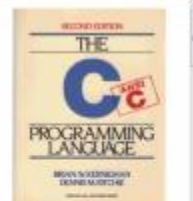
You can also look it up at the [Prentice-Hall](#) home page.

Kernighan, a relatively uncommon name, is a good search term to find it quickly at all of these places.

If you are curious, a [Gif-format graph](#) of its sales rank at Amazon.com for the last couple of years is available.

- Here is a list of [errors](#) in the published version; many of these are corrected in recent printings.
- The history of the language is traced in "The Development of the C Language", from HCPL II, 1989: [hardcopy](#), or printable [PostScript](#) or [PDF](#). This and other historical material, including early manuals and compilers, is available at [Dennis Ritchie's home page](#), while [Brian Kernighan's home page](#) collects pointers to his work on C and diverse other languages and systems.

The book has been translated into many languages, including



C 언어의 특징

- 절차지향 언어

- 함수 중심으로 구현되는 절차지향 언어(procedural language)
 - 적어도 하나 이상의 절차인 여러 함수 (function)로 구성되는 언어
 - 구조적 프로그래밍(structured programming)
 - 복잡한 문제를 잘 정의된 여러 개의 함수와 자료로 나누어 구성하고 해결

- 간결하고 효율적인 언어

- 다양한 연산과 이미 개발된 다양한 시스템 라이브러리(rich library)를 제공
 - 비트연산, 증감연산, 축약대입연산 등

- 이식성이 좋은 언어

- 다양한 CPU 와 플랫폼의 컴파일러를 지원



그림 1-20 C 언어의 주요 기능과 특징

C 언어를 배워야 하는 이유

- 많은 언어에 영향을 미친 가장 기본이 되는 프로그래밍 언어
 - 자바, C++, C#, 파이썬 등 여러 프로그래밍 언어에 많은 영향을 미침
 - 향후 언어 습득이 매우 쉬워 짐
- 현장에서 다양한 분야에 사용되는 범용적인 프로그래밍 언어
- 프로그래밍 지식과 프로그래밍 방법을 학습

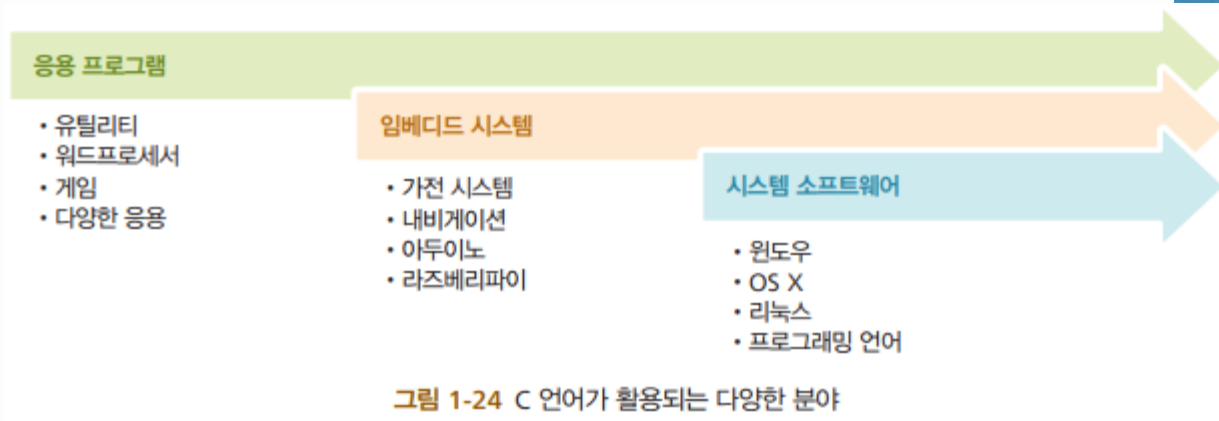


그림 1-25 프로그래머나 정보기술 개발자가 되기 위한 초석

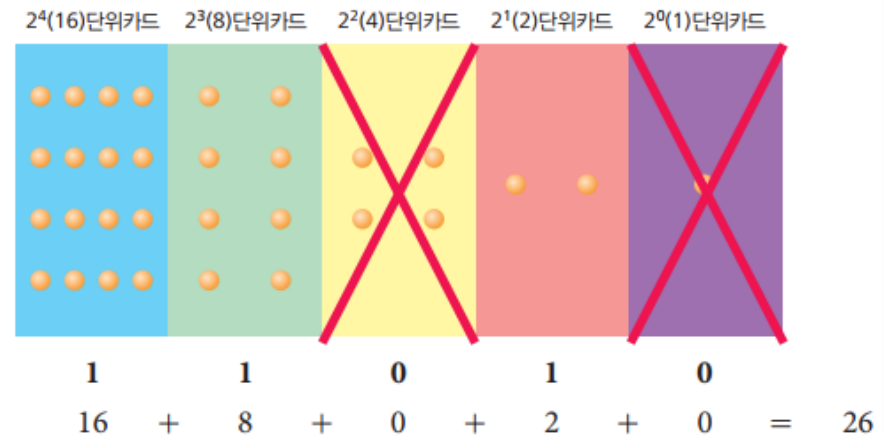
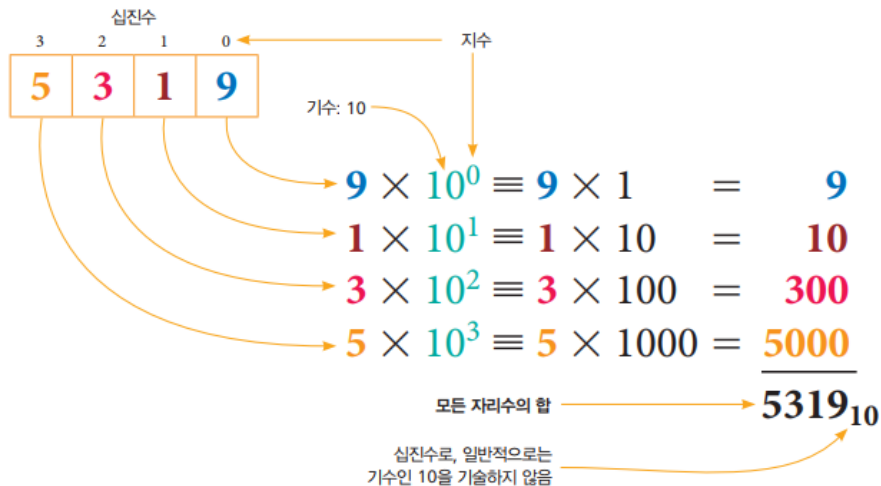
프로그래밍의 내부 표현, 0과 1

• 우리에게 친숙한 10진수

- 하나의 자릿수(digits)에 사용하는 숫자가 0, 1, 2, 3, 4, 5, 6, 7, 8, 9까지 열 개
- 기수(base)
 - 10

• 컴퓨터가 이해하는 2진수

- 수의 자릿수에 사용할 수 있는 숫자가 2개(0, 1) 이므로 2진수
 - 2진수의 기수는 2



비트와 바이트

- 비트(bit)

- Binary digit의 합성어
 - 가장 작은 기본 정보 단위(basic unit of information)
- 진수인 1과 0으로 비트의 표현이 가능
 - 전기의 흐름 상태가 온(on)과 오프(off) 두 가지

- 바이트(byte)

- 비트가 연속적으로 8개 모인 정보 단위
 - 1바이트는 8개의 비트를 조합하므로 총 256가지의 정보 종류를 저장

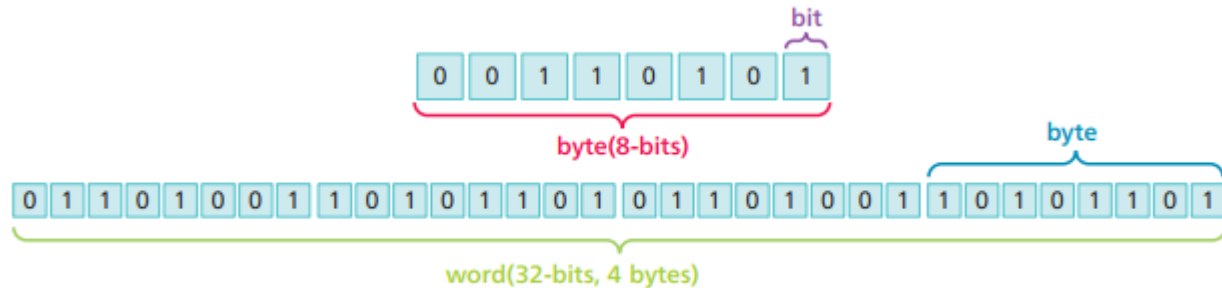


그림 1-31 비트, 바이트, 워드



TIP 저장 용량 단위

다음은 자주 이용하는 바이트의 단위로 파일이나 주기억장치, 저장장치의 크기를 표현하는 단위이다. 정확히 말하자면 바이트가 정보 용량의 단위이고 킬로, 메가, 기가, 테라 등은 그 크기를 표현한다. 즉 킬로(Kilo)는 2¹⁰을 의미하며 1024개를 나타낸다. 마찬가지로 메가(Mega)는 계량단위 앞에 사용하여 1024 x 1024인 백만을 의미한다. 마찬가지로 기가 바이트(Giga Byte)는 2³⁰을, 테라 바이트(Tera Byte)는 2⁴⁰을 의미한다. 페타 바이트(Peta Byte)는 2⁵⁰을, 엑사 바이트(Exa Byte)는 2⁶⁰을 의미한다.

표 1-1 저장 용량 단위

단위	약자	표현	바이트(byte)	근접
1 bit	1 b (lower case)	0 또는 1		
1 Nibble		4 bits	½ byte	
1 Byte	1B (upper case)	8 bits 또는 2 Nibbles 또는 2 ³ bits	1 byte	
1 Kilobyte	1 KB	2 ¹⁰ bytes	1,024 byte	1 thousand bytes
1 Megabyte	1 MB	(2 ¹⁰) ² bytes	1,048,576 byte	1 million bytes
1 Gigabyte	1 GB	(2 ¹⁰) ³ bytes	1,073,741,824 byte	1 trillion bytes
1 Terabyte	1 TB	(2 ¹⁰) ⁴ bytes	1,099,511,627,776 byte	1 quadrillion bytes
1 Petabyte	1 PB	(2 ¹⁰) ⁵ bytes	1,125,899,906,842,624 byte	1 quintillion bytes
1 Exabyte	1 EB	(2 ¹⁰) ⁶ bytes	1,152,921,504,606,846,976 byte	1 sextillion bytes
1 Zetabyte	1 ZB	(2 ¹⁰) ⁷ bytes	1,180,591,620,717,411,303,424 bytes	1 septillion bytes

논리 표현과 연산

- 논리 값

- 참(true)과 거짓(false)을 의미하는 두 가지 정보
- 하나의 비트 정보도 0과 1 이므로 이를 각각 거짓과 참으로 표현

- 부울 대수(Boolean Algebra)

- 영국의 수학자 조지 부울(George Boole)이 제창한 기호 논리학의 한 분야
- 0과 1 두 값 중 하나로 한정된 변수들의 상관 관계를 AND, OR, NOT 등의 여러 연산자를 이용하여 논리적으로 표현
- 기본적인 논리 연산
 - AND, OR 연산과 NOT 연산
 - AND 게이트, OR 게이트, NOT 게이트와 같은 논리 회로로 그림

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1



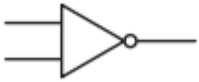
AND

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

OR

A	not B
0	1
1	0

NOT

Gate	Symbol	Operator
AND		$A \cdot B$
OR		$A + B$
NOT		\bar{A}

문자 표현 방식: 아스키코드와 유니코드

- **아스키(ASCII) 코드**

- 1967년에 표준으로 제정: American Standard Code for Information Interchange
 - **1986년에 마지막으로 개정**
- 아스키는 초창기에 7비트 인코딩(8비트 중 7비트만 데이터 비트로 사용)
 - **총 128개의 코드로 구성**
 - 33개의 출력 불가능한 제어문자들과 공백을 비롯한 95개의 출력 가능한 문자
 - 주로 그래픽에 관련된 문자와 선 그리기에 관련된 문자가 추가되어 256개로 확장

- **유니코드**

- 유니코드 협회(Unicode Consortium)가 제정하여 1991년 버전 1.0이 발표
 - **전 세계 모든 문자의 표현 위해 2바이트 즉, 16비트로 확장된 코드 체계**
 - 전 세계의 모든 문자를 일관되게 표현하고 다룰 수 있도록 설계된 산업 표준
- 1996년 65,536자의 코드 영역을 언어학적으로 분류
 - **한글 완성자 11,172자의 한글과 중국, 일본을 포함해 세계 유수의 언어 문자를 배열해 만든 유니코드**
- 국제표준화기구(ISO: International Organization for Standardization)
 - **계속 수정 보완**

표 1-3 아스키코드 표

10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	N	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	G	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	W	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	R	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

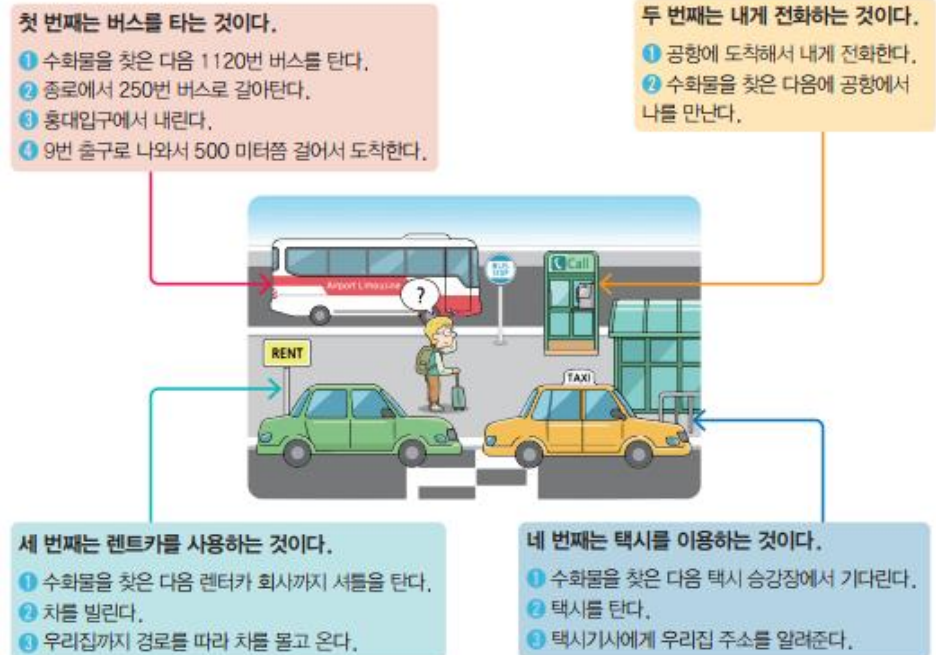
컴퓨팅 사고력과 알고리즘

• 컴퓨팅 사고력(CT: Computational Thinking)

- 일반인이 다양한 모든 분야의 문제해결에 적용될 수 있도록 컴퓨터의 프로그래밍을 구현하는 과정의 사고에서 도출된 절차와 역량
 - 1980년 미국 MIT 대학의 시무어 페퍼트(Seymour Papert) 교수가 처음 언급
 - 2006년 카네기 멜론 대학 자넷 wing(Jeannette Wing) 교수는 그녀의 논문
 - '컴퓨팅 사고력은 컴퓨터 과학자뿐만이 아니라 모든 사람 누구나 배워서 활용할 수 있는 보편적인 사고이자 기술이다.'라고 제시

• '알고리즘'

- 어떠한 문제를 해결하기 위한 절차나 방법
 - 명확히 정의된(well-defined) 유한 개의 규칙과 절차의 모임
 - 컴퓨터 프로그램
 - 특정한 업무를 수행하기 위한 정교한 알고리즘들의 집합



소프트웨어 개발 방법

• 소프트웨어 공학

– 공학적 원리에 의하여 소프트웨어를 개발하는 학문

- 소프트웨어 개발과정인 분석, 설계, 개발, 검증, 유지보수 등 개발수명 주기 전반에 걸친 계획·개발·검사·보수·관리, 방법론 등을 연구하는 분야

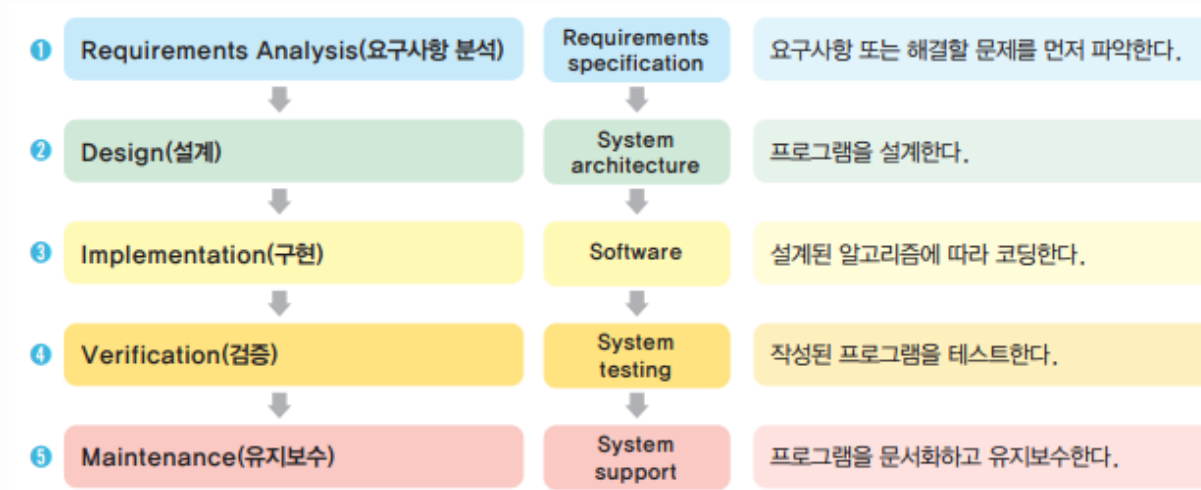


그림 1-39 소프트웨어 개발 과정

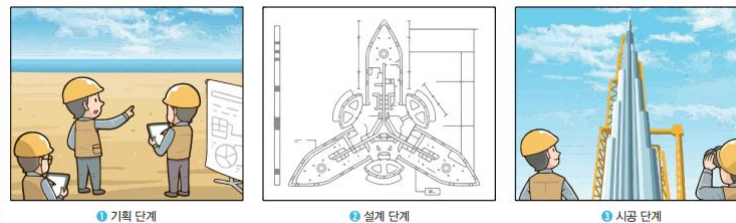


그림 1-38 부르즈 할리파의 건축 과정

알고리즘 기술 방법

- 자연언어
- 순서도, 흐름도
- 의사코드, 슈도코드(pseudo code)
 - 간결한 특정 언어로 코드를 흉내 내어 알고리즘을 써놓은 코드

```
binarySearch(A[1..N], numberToSearch)
  min<-0, max<-N, Mid<-0
  while(max != min)
    mid = (min + max)/2
    if(A[mid] == search)
      return mid
    else if(a[mid],search)
      min = mid
    else
      max = mid
  return -1
```

그림 1-40 의사 코드 예

시작

- 1 기차표 예매사이트에 접속한다.
 - 2 원하는 시간과 역에 정차하는 좌석이 있는가?
 - 3 있으면 5로 간다.
 - 4 없으면 다시 2로 가서 다른 좌석을 검색한다.
 - 5 결제한다.
 - 6 구매 완료한 기차표를 출력한다.
- 종료

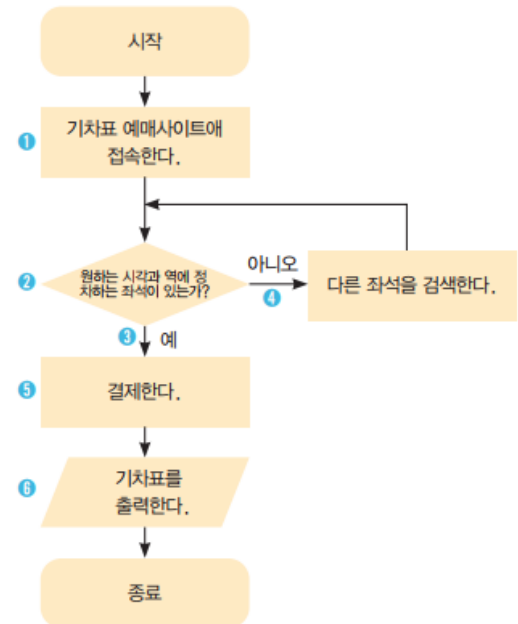


그림 1-43 '인터넷에서 기차표를 예매하고 출력'하는 과정의 자연어(한글) 기술과 순서도 표현

50 ~ 60년대에 개발된 프로그래밍 언어

- 포트란(FORTRAN)

- FORMula TRANslating system(수식 번역 시스템)의 약자
 - 과학과 공학 및 수학적 문제들을 해결하기 위해 고안된 프로그래밍 언어
 - 널리 보급된 최초의 고급 언어
- 1957년경, IBM의 존 배커스(John Backus) 개발한 프로그래밍 언어
 - 지금도 과학 계산 분야 등에서는 많이 사용
 - 가장 오래된 언어지만 언어 구조가 단순해 놀라운 생명력을 갖추

- 코볼(COBOL)

- 코볼(COMmon Business Oriented Language)의 약자
 - 협회 CODASYL이 1960년, 사무처리에 적합한 프로그래밍 언어로 개발
 - 포트란에 이어 두 번째로 개발된 고급 언어
- 사무처리 분야 적합
 - 파일이나 데이터베이스에서 데이터를 쉽게 읽고 쓰며
 - 양식을 가진 보고서를 쉽게 만들 수 있는 등 사무처리에 효율적

- 알골(ALGOL)

- 알고리즘(ALGORithm) 을 표현하기 위한 언어
 - ALGORithmic Language를 줄여서 만든 이름

70년대 이후 개발된 주요 프로그래밍 언어

• 파스칼

- 1971년 스위스 취리히 공과대학교의 니콜라우스 비르트 (Nicholas Wirth) 교수 개발
- 알고리즘 학습에 적합
 - 프랑스의 수학자인 파스칼(Pascal)의 이름에서 따온 언어

• C++

- AT&T의 얀 스트로스트룹(Bjarne Stroustrup)이 개발
 - 1972년에 개발된 C 언어는 1983년에 프로그램 언어 C++로 발전
 - C언어에 익숙한 프로그래머들이 C++언어를 쉽게 배울 수 있다는 장점
- 객체지향 프로그래밍(OOP: Object Oriented Programming)을 지원



NOTE: C++ 창시자 비야네 스트로스트룹

비야네 스트로스트룹(Bjarne Stroustrup)은 '최초의 객체지향 언어로 인정'되는 Simula67을 사용하여 프로그램을 개발하던 중 좀 더 편리하고 효율적인 프로그래밍 언어를 만드는 일을 시작하였고 1980년에 C 언어의 장점인 시스템 프로그래밍 능력에 'Simula67의 클래스 개념'을 도입하여 'C with Classes'라는 새로운 이름의 프로그래밍 언어를 설계했다. 이렇게 만들어진 'C with Classes'는 객체지향 프로그래밍에 필수 요소인 가상함수와 연산자 오버로딩 개념이 추가되어 1983년 C++ 1.0으로 소개되었다.



파이썬

- 1991년 네덜란드의 귀도 반 로섬(Guido van Rossum)이 개발
 - 객체지향 프로그래밍 언어
 - 프로그래밍 언어 파이썬이 대학의 컴퓨터기초 교육에 많이 활용
 - 비영리의 파이썬 소프트웨어 재단이 관리하는 개방형, 공동체 기반 개발 모델
- C 파이썬(cpython)
 - C언어로 구현된 버전이 사실상의 표준
- 베이직과 같은 인터프리터 언어
 - 간단한 문법구조를 가진 대화형 언어
 - 쉽고 빠르게 개발할 수 있어, 개발기간이 매우 단축되는 것이 장점

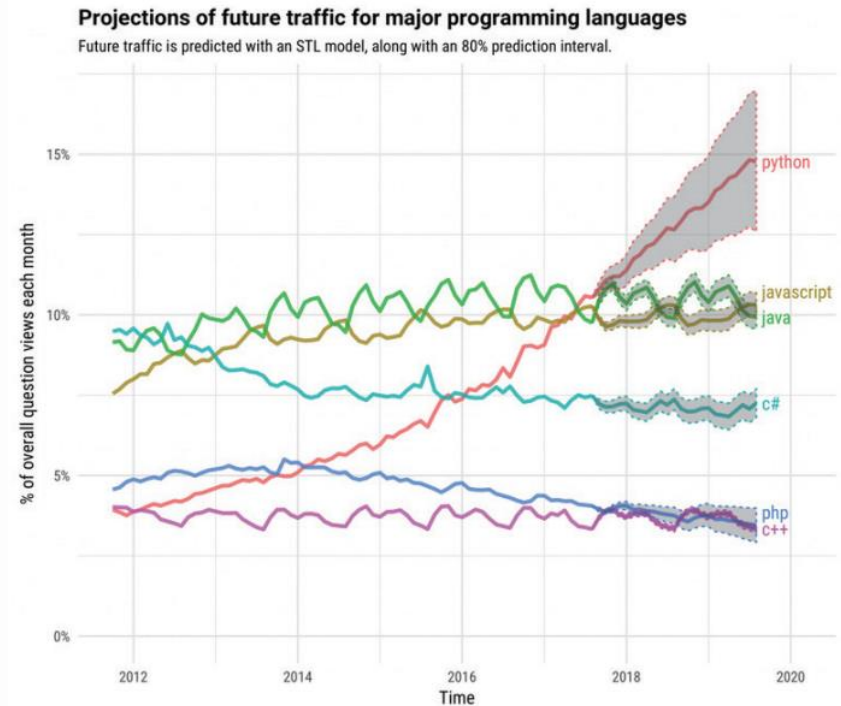


그림 1-44 프로그래밍 언어 예측(www.stacoverflow.com)

자바와 코트린

• 자바

- 1995년에 공식 발표
 - C++를 기반으로 한 객체지향 프로그래밍 언어
- 가전제품들을 제어하기 위해 고안한 언어
 - 선 마이크로시스템즈사의 그린 프로젝트(Green Project)
 - 1992년 양방향 TV를 만드는 제어 박스의 개발을 위해 개발
 - 오크(oak)라는 언어를 발전시켜 개발
 - 프로젝트의 책임자, 제임스 고슬링(James Gosling) 중심 개발

• 코트린

- 제트브레인스(JetBrains) 2011년에 공개
 - 자바 통합개발환경인 인텔리제이 아이디어(IntelliJ IDEA)를 개발사
- 자바가상기계(JVM)와 안드로이드(android) 환경 개발 언어
 - 객체지향 프로그래밍 언어
- 코딩하거나 읽기 쉽도록 매우 간결하고 간편
- 자바 언어와의 상호 운용, 100% 지원

감사합니다.