컴퓨터 프로그래밍

Computer Programming

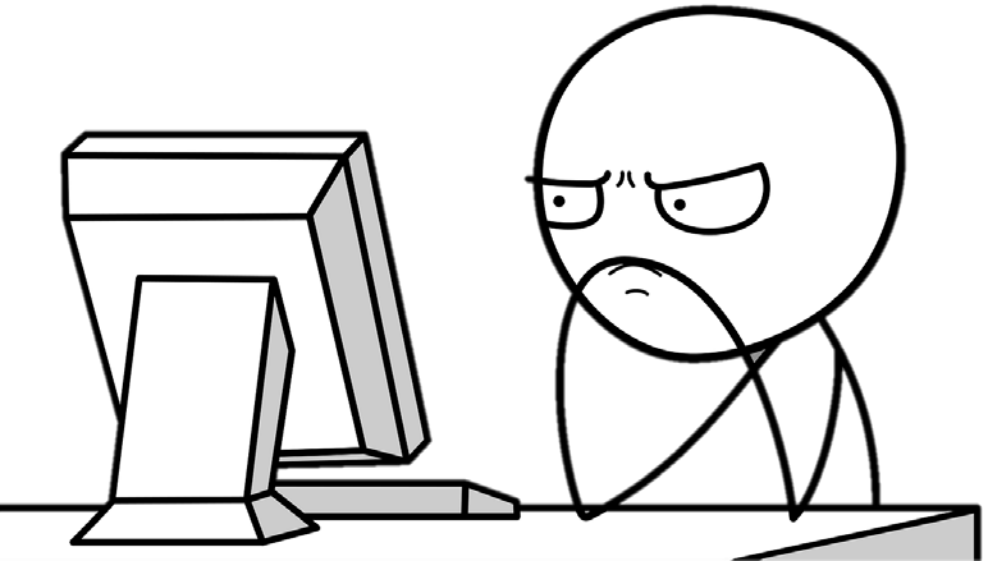
Groovy
programming
Java
Clojure
Erlang
OCaml
Assembly
C
LaTeX
PHP
Lua
Python
BASIC
MATLAB
Lisp
Processing
Haskell
Ruby
Visual
Pascal
Smalltalk
Ada
APL
Delphi
IBM
REXX
ABC
TypeScript
Programming
language
Elixir
JavaScript
Eiffel
PL/SQL
Dart
Scala
Forth
FoxPro
Bash
AppleScript
shell
Basic
ISO/IEC
Fortran
Squeak
EXEC
CIL
DataFlex
Microsoft
Max
CPL
Revolution Pipelines
Coral
Not
now
also PowerShell
Robot
Common
Informix-4GL
Strand
GDL
Chuck
Hartmann
Nemerle
GraphTalk
Constraint
Sawzall
CorVision
Action
Kojo
Arc
Shell
Fancy
Modelica
XOTcl
COBOL
Cesil
Autoit
Lisplaces
Q
CL
J++
Coldet
Pl
Scriptl
Simu
Pro
Cobol
Scriptl
Pro
Cobol
Scriptl
Simu
Pro
Cobol
Scriptl
Simu
Pro
Cobol
Scriptl
Simu

01
프로그래밍언어 개요

목차

1. ‘프로그램’이 무엇일까?
2. 언어의 계층과 번역
3. 왜 C 언어를 배워야 할까?
4. 프로그래밍의 자료 표현
5. 소프트웨어 개발
6. 다양한 ‘프로그래밍 언어'

1. ‘프로그램’이 무엇일까?



프로그램

- 스마트폰과 노트북, 혹은 데스크탑 컴퓨터에서 특정 작업을 위해 컴퓨터를 작동시키는 것
- 특정 목적의 작업을 수행하기 위한 관련 파일의 모임

생활에서의 프로그램

- 텔레비전의 방송에서도 예능 프로그램, 오디션 프로그램 등 프로그램이라는 용어를 자주 사용
- 연극이나 방송 따위의 진행 차례나 진행 목록
- 프로그램이란 용어의 공통적 의미
- 특정한 목적을 수행하기 위한 이미 정해놓은 순서적인 계획이나 절차를 의미

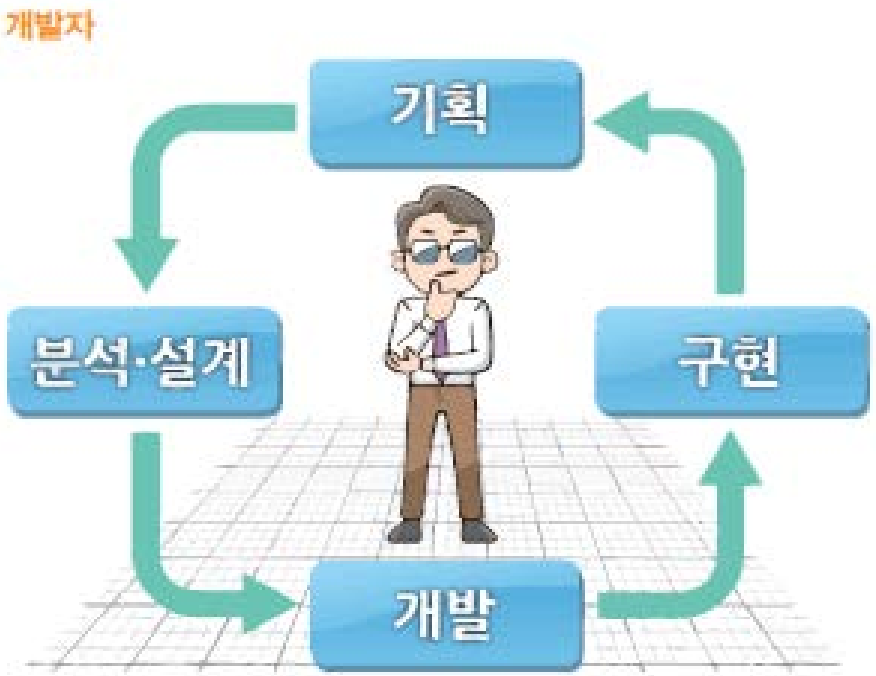
정보기술에서의 프로그램

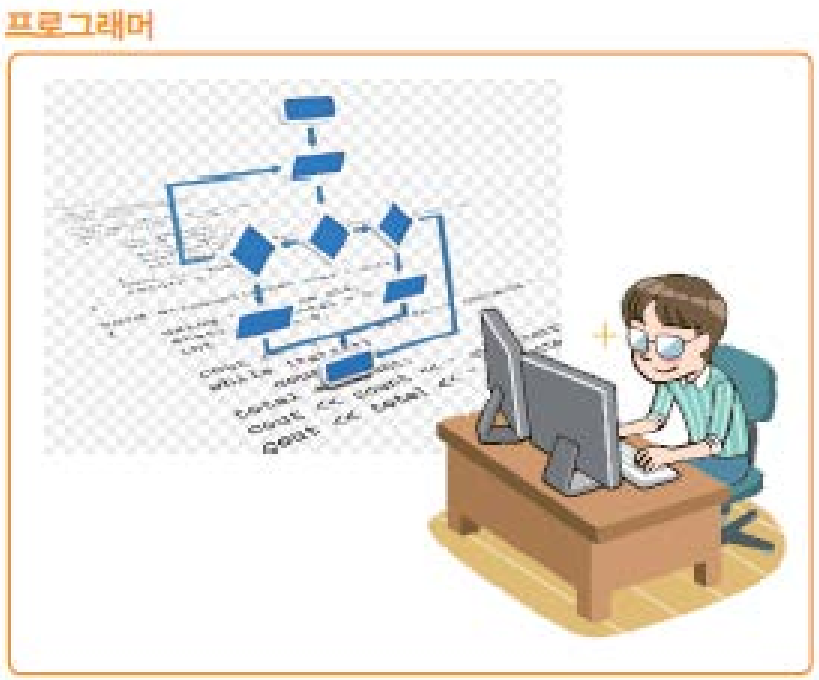
- 특정 작업을 수행하기 위하여 그 처리 방법과 순서를 기술한 명령어와 자료로 구성
- 컴퓨터에게 지시할 일련의 처리 작업 내용
- 사용자의 프로그램 조작에 따라 컴퓨터에게 적절한 명령을 지시하여 프로그램이 실행

1. 알람 시각 설정: 아침 7시 30에 알람을 울려라.
2. 소리 설정: 알람 음은 듣기 좋은 '물방울 소리'로 지정하자.
3. 스누즈 기능 설정: 처음에는 작은 소리였다가 점점 소리를 크게 하라.
4. 정지 조건 설정: 버튼을 누르면 알람 정지해라.

프로그래머와 개발자

- 프로그래머: 컴퓨터와 스마트폰 등의 정보기기에서 사용되는 프로그램을 만드는 사람
- 개발자: 소프트웨어 구축을 위한 기획에서부터 분석·설계와 개발, 구현에 이르는 모든 과정에 참여하는 사람



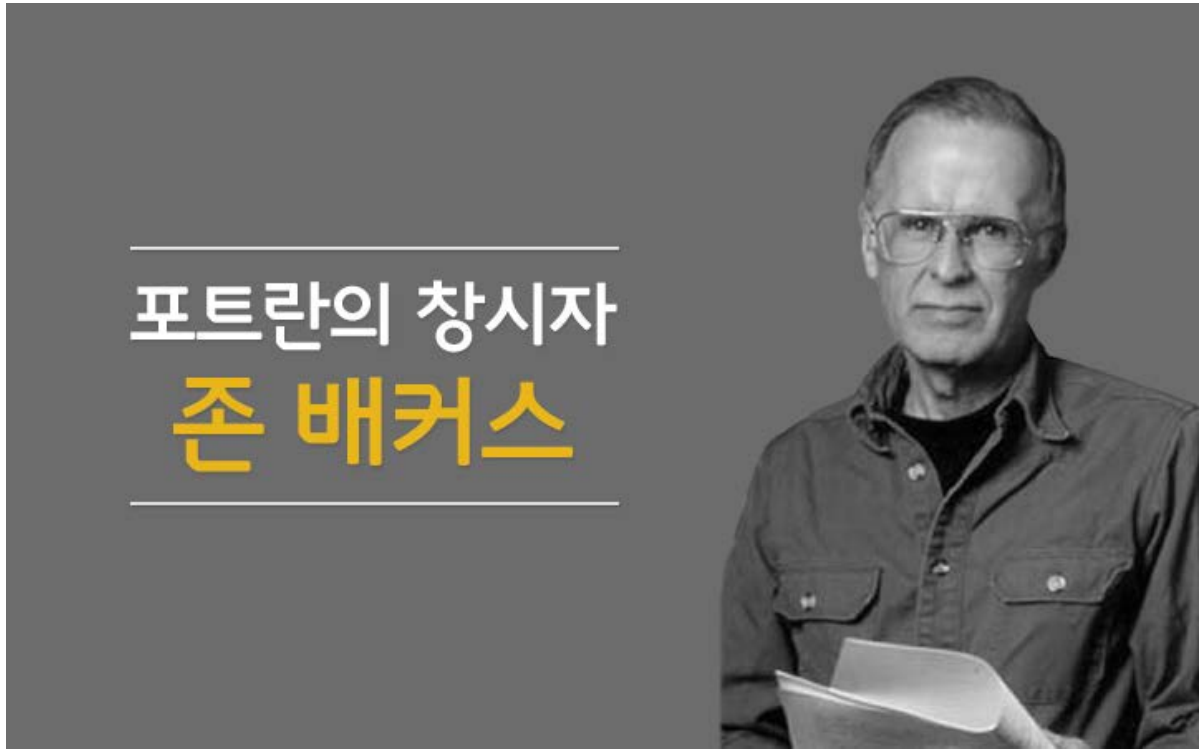


프로그래밍 언어

- 프로그램을 개발하기 위해 사용하는 언어
- 사람과 컴퓨터가 서로 의사 교환을 하기 위한 언어
- 사람이 컴퓨터에게 지시할 명령어를 기술하기 위하여 만들어진 언어
- 매우 다양한 언어가 개발

Note: 최초의 프로그래밍 언어는?

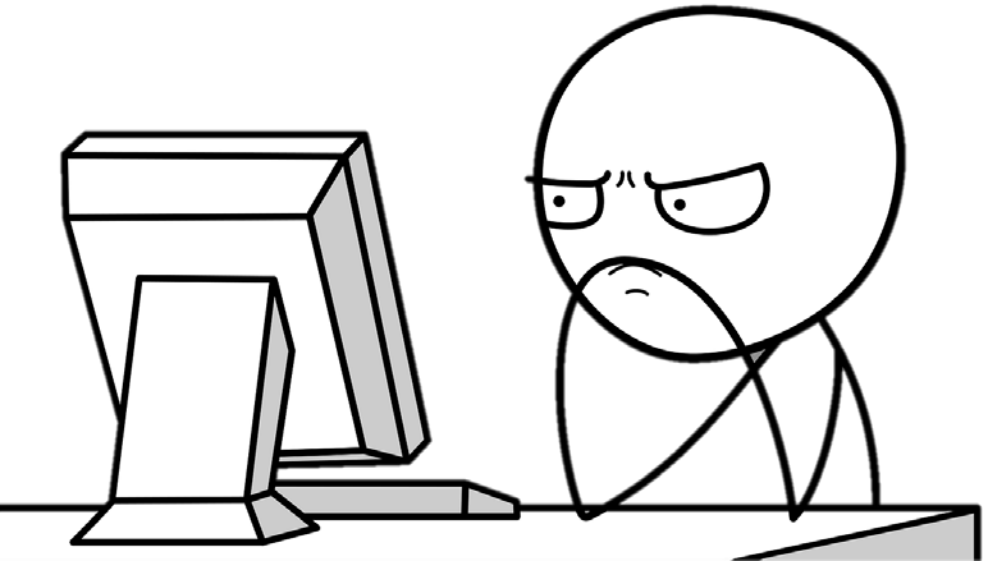
- 포트란FORTRAN
  - 최초의 대중화된 프로그래밍 언어는 1950년 중반 IBM에 근무하는 28세의 젊은 과학자인 존 배커스John Backus가 개발
  - 수식 변환기FORmulaTRANslator라는 의미의 약자
  - 공학과 과학 분야에서 계산 위주로 사용을 목적으로 개발된 프로그래밍 언어



Note: 최초의 프로그래머는 누구일까?

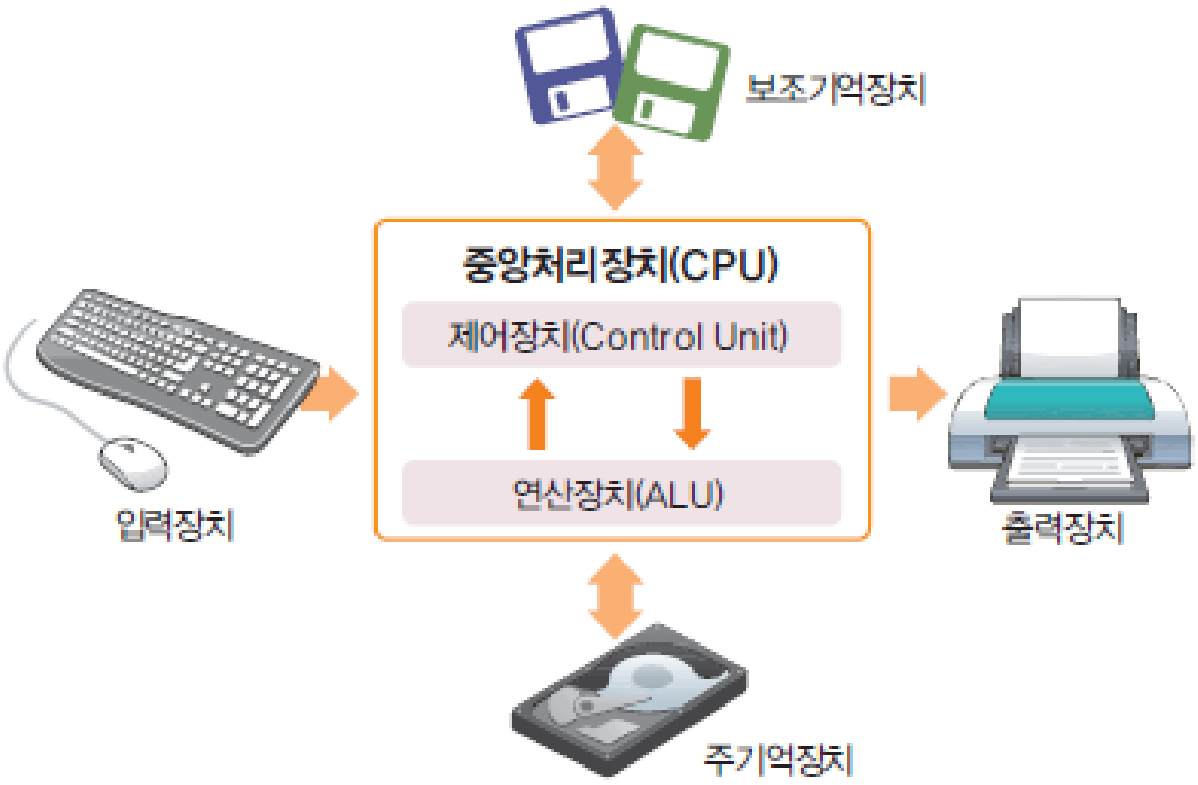
- 인류 최초의 프로그래머: 오거스타 에이다
  - 수학에 천재적 재능
  - 최초의 컴퓨터 창안자 찰스 배비지가 고안한 기계를 이해
  - 1833년에 배비지가 고안한 '분석 엔진Analytical Engine'에 계산과정을 기술하는 프로그램을 만들어 오늘날 일반적으로 사용하는 컴퓨터의 시조가 되는데 공헌
  - 오늘날 컴퓨터의 원형이 된 '분석 엔진'에 관한 책인 '배비지의 해석기관에 대한 분석Observations on Mr. Babbage's Analytical Engine을 출간
  - 현대 컴퓨터 프로그래밍 역사의 기원
  - 100년 뒤인 1950년대에 이르러, 이러한 에이다의 업적을 기려 그녀에게 '세계 최초의 프로그래머'라는 호칭이 주어짐
  - 1979년에 미국 국방성에서는 새로 개발한 프로그래밍 언어를 그녀의 이름을 따서 "ADA"라고 명명

2. 언어의 계층과 번역



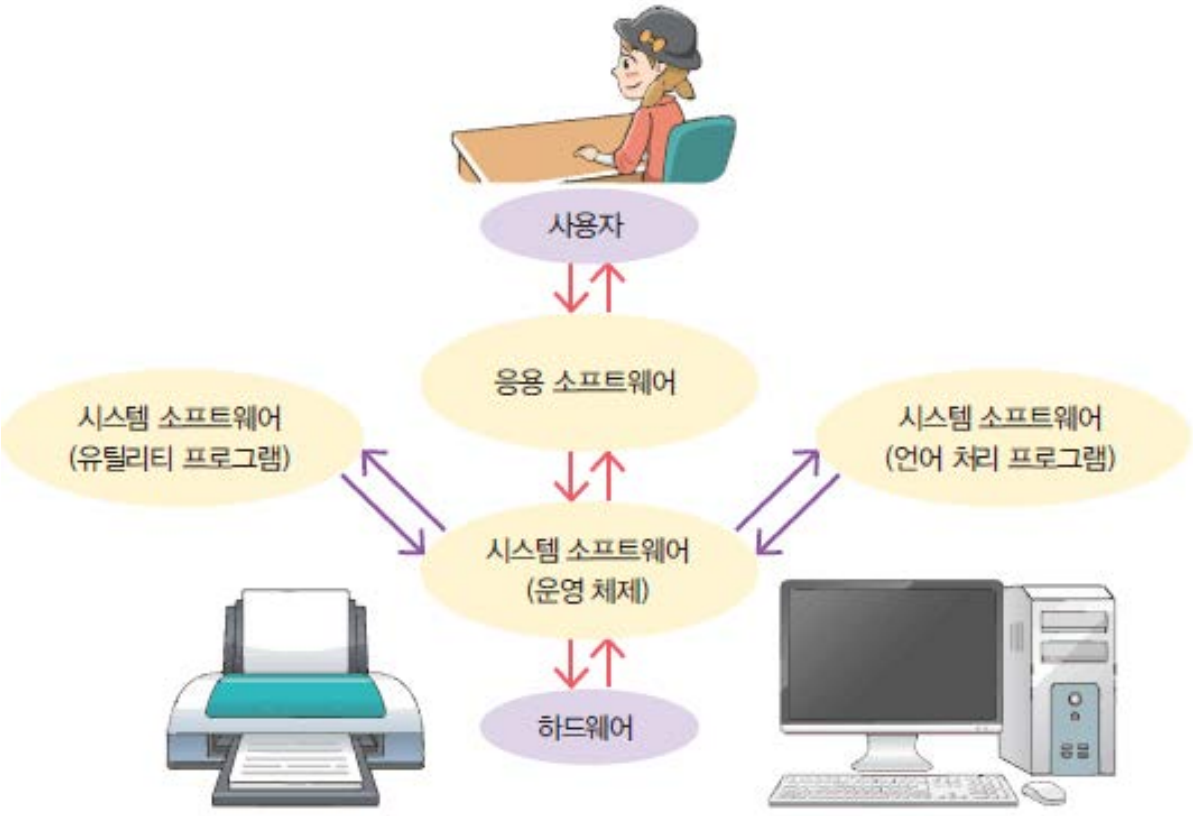
컴퓨터, 하드웨어와 중앙처리장치

- 영어 단어 compute와 er의 조합으로 '계산하는 기계', 즉 컴퓨터는 '전자적으로 계산을 수행하는 장치'
- 하드웨어hardware와 이 하드웨어를 작동하게 하는 소프트웨어software로 구성
- 중앙처리장치
  - 연산을 수행하는 연산장치ALU: Arithmetic Logic Unit와 연산을 제어하는 제어장치control unit로 구성
  - 중앙처리장치의 칩이 프로세서processor

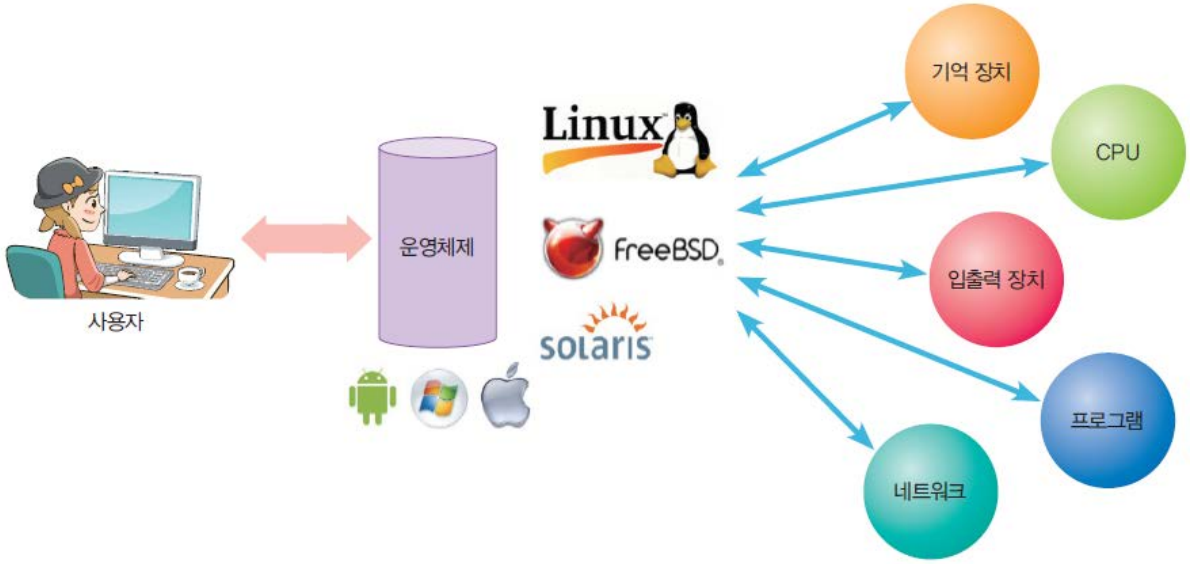


소프트웨어

- 컴퓨터가 특정 작업을 수행할 수 있도록 해주는 전자적인 명령어 집합: 컴퓨터의 하드웨어가 해야 할 작업 내용을 지시
- 소프트웨어는 크게 응용 소프트웨어와 시스템 소프트웨어로 나뉨
- 시스템 소프트웨어: 컴퓨터가 잘 작동하도록 도와주는 기본 소프트웨어
- 응용 소프트웨어: 문서 작성이나 인터넷 검색, 게임 하기, 동영상 보기 등과 같은 특정 업무에 활용되는 소프트웨어



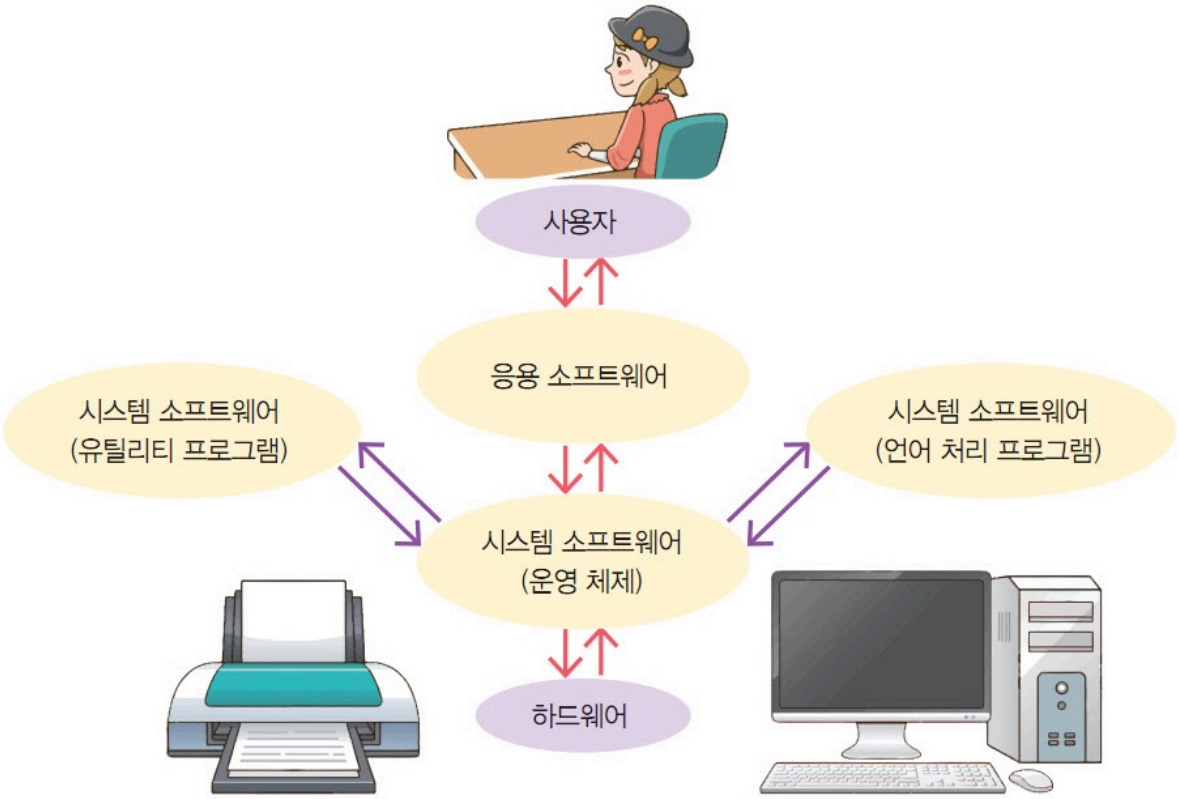
시스템 소프트웨어 – 운영체제

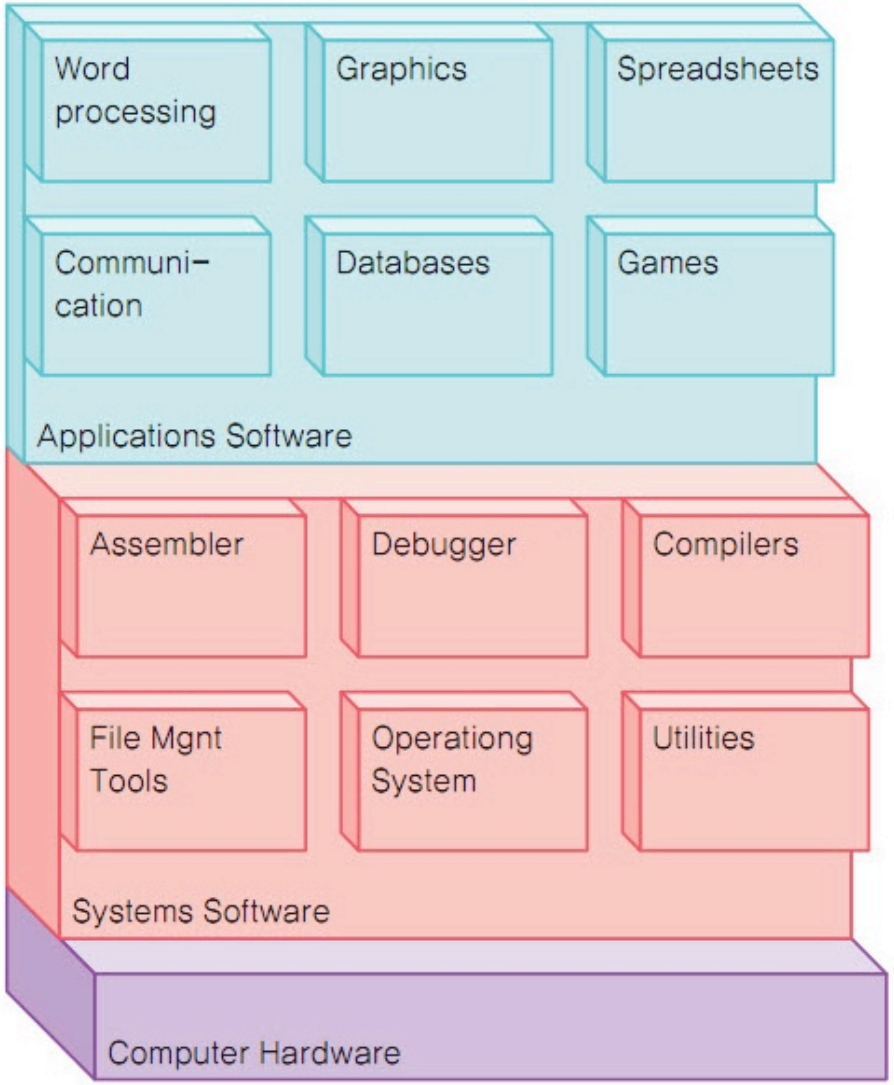


- 컴퓨터 하드웨어 장치의 전반적인 작동을 제어하고 조정하며, 사용자가 최대한 컴퓨터를 효율적으로 사용할 수 있도록 돕는 시스템 프로그램
- 하드웨어와 응용프로그램간의 인터페이스 역할을 하면서 CPU, 주기억장치, 입출력장치 등의 컴퓨터 자원과 함께 다양한 프로그램과 네트워크 등을 관리
- 유닉스Unix, 리눅스Linux, 윈도우즈Windows, 맥Mac OS X 등
- 스마트폰과 같은 스마트 기기를 위한 운영체제로는 iOS, 안드로이드Android, 파이어폭스Firefox, 윈도우폰Window Phone 등이 있음

시스템 소프트웨어 – 유틸리티 프로그램

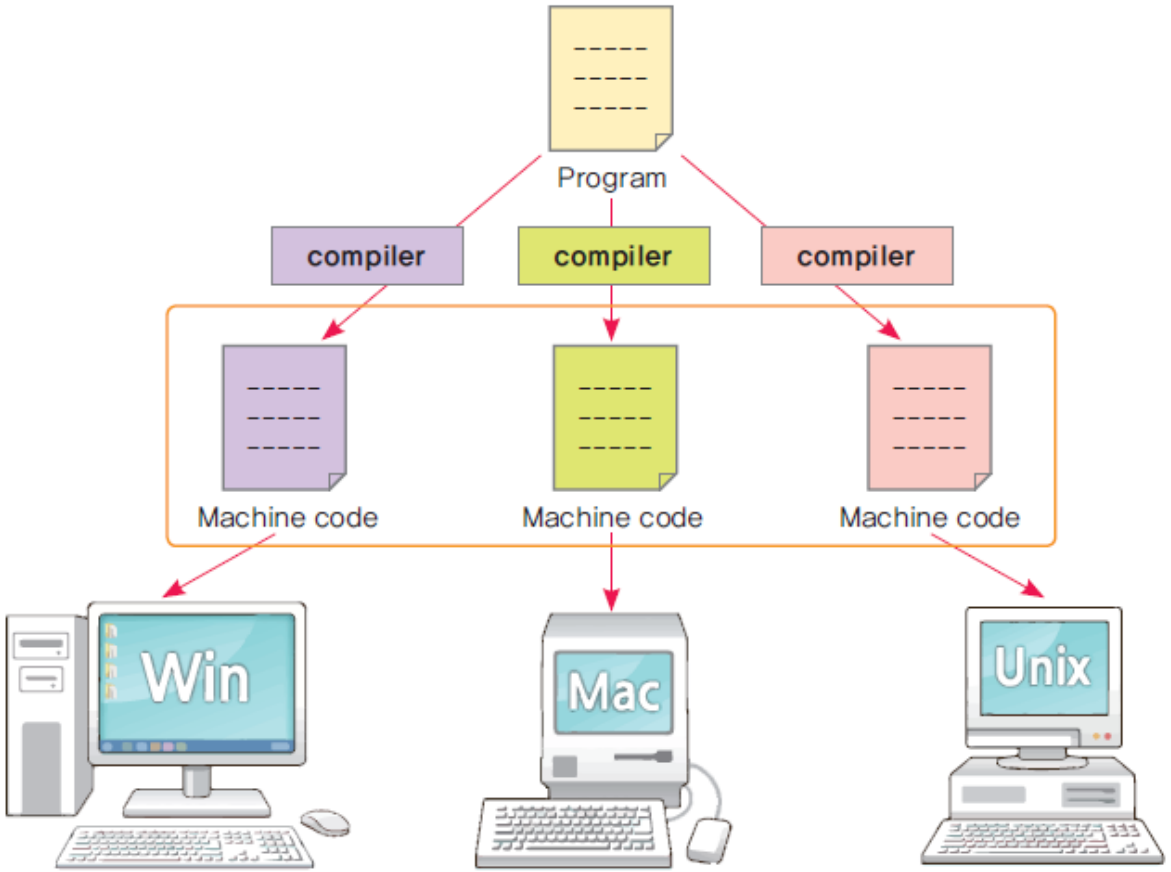
- 운영체제를 돕고 컴퓨터 시스템이 원활하게 작동하도록 도움을 줌





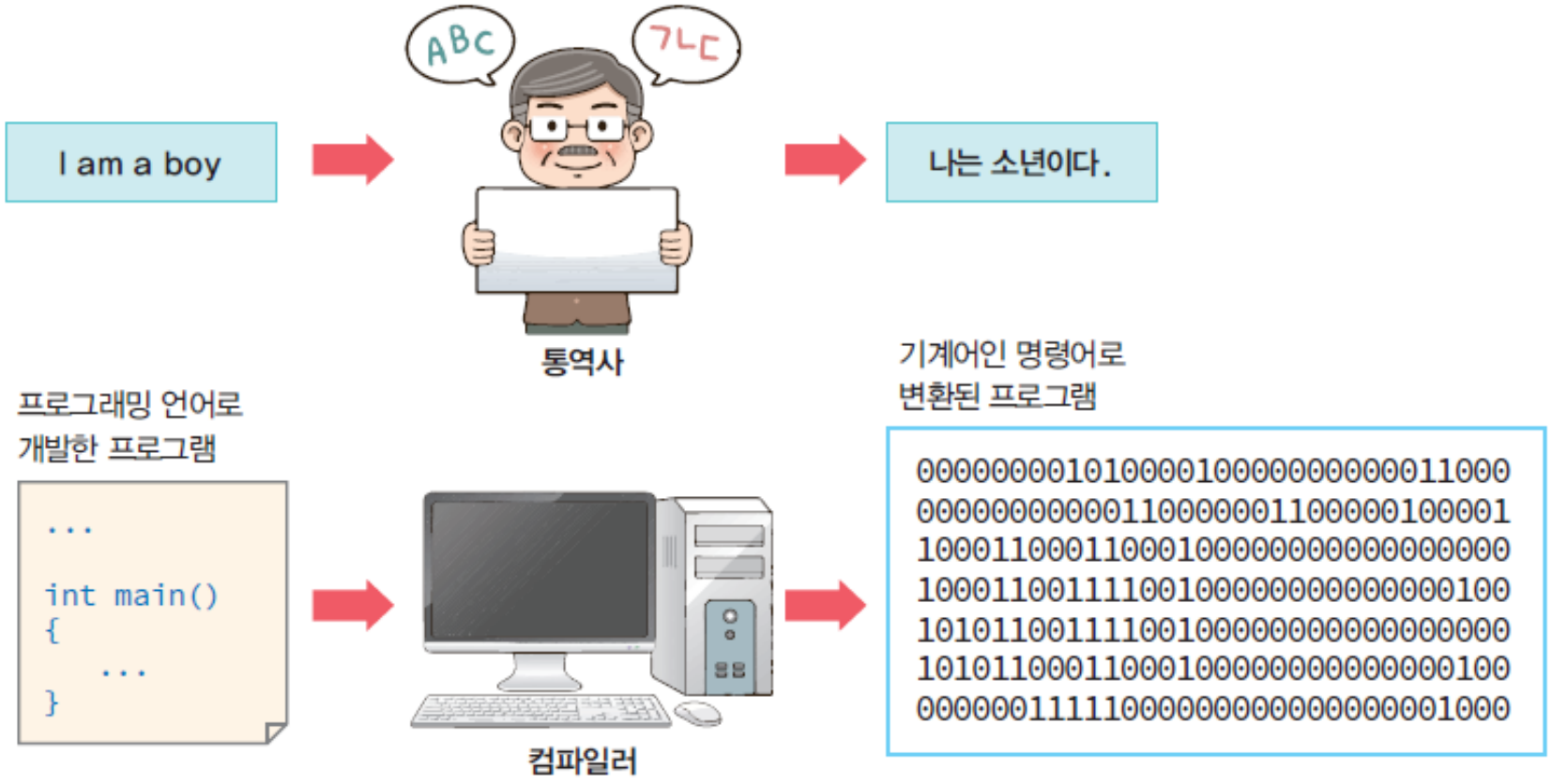
기계어와 컴파일러

- 컴퓨터는 기계어라는 것만을 인식
  - 즉 전기의 흐름을 표현하는 1과 흐르지 않음을 의미하는 0으로 표현되는 기계어machine language가 바로 컴퓨터가 유일하게 바로 인식하는 언어
- 통역사인 컴파일러compiler가 필요
  - 프로그래밍 언어를 사용하는 프로그래머와 기계어를 사용하는 컴퓨터가 서로 의사 교환을 하려면 통역사와 같은 번역기가 필요



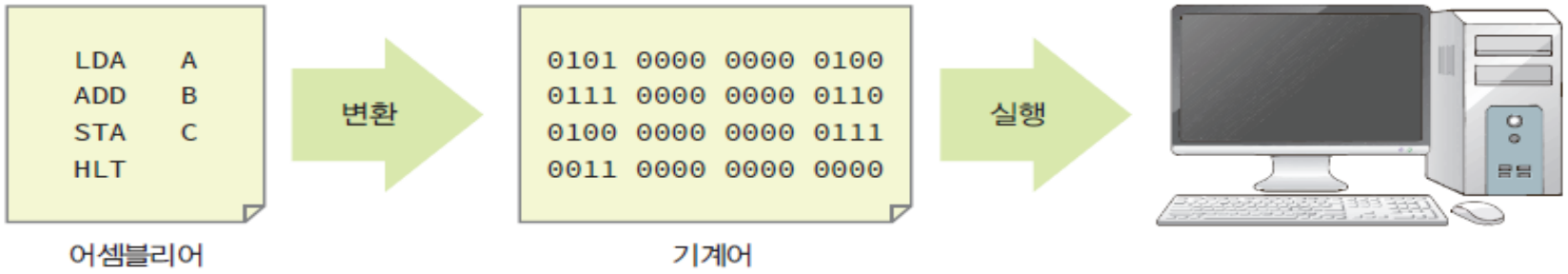
기계어

- 중앙처리장치CPU: Central Processing Unit에 종속dependent되는 언어
- 컴파일러compiler라는 변환기에 의해 프로그램이 기계어로 구성된 기계 코드로 변환되어 특정한 플랫폼에서 실행



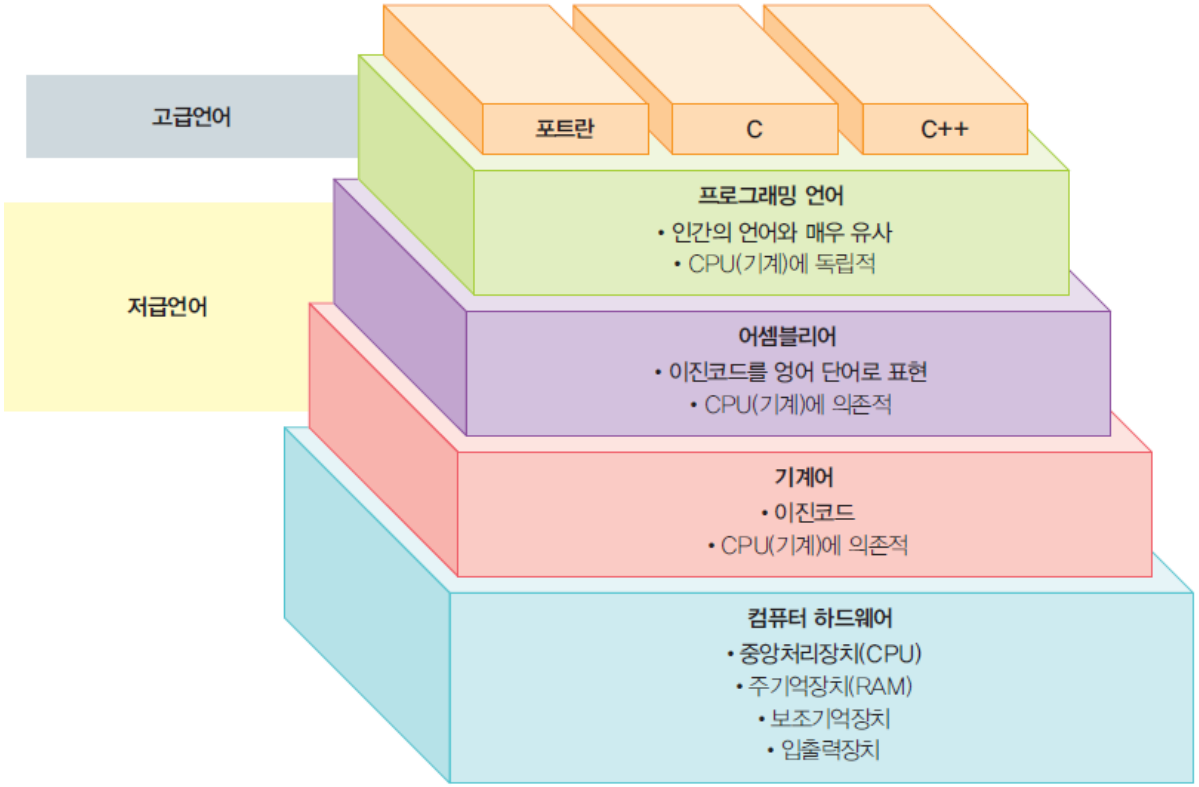
어셈블리어

- 어셈블리어assembly language
  - 기계어를 프로그래머programmer인 사람이 좀 더 이해하기 쉬운 기호 형태로 일대일 대응시킨 프로그래밍 언어
    - CPU마다 제각각 다름
  - 어셈블리 언어는 기계어보다는 프로그래밍이 훨씬 용이
    - 문장과 연산코드가 일대일 대응되기 때문에 기계어처럼 하드웨어 장치에 대한 강력한 통제 역시 가능하다는 장점
- 어셈블리 명령어 예
  - LDA(LoaD Address), ADD(ADD), STA(STore Address) 등
    - 명령어를 기호화한 것을 니모닉mnemonic이라 함
  - 연산식 C = A + B을 처리하는 프로그램을 기계어와 어셈블리어



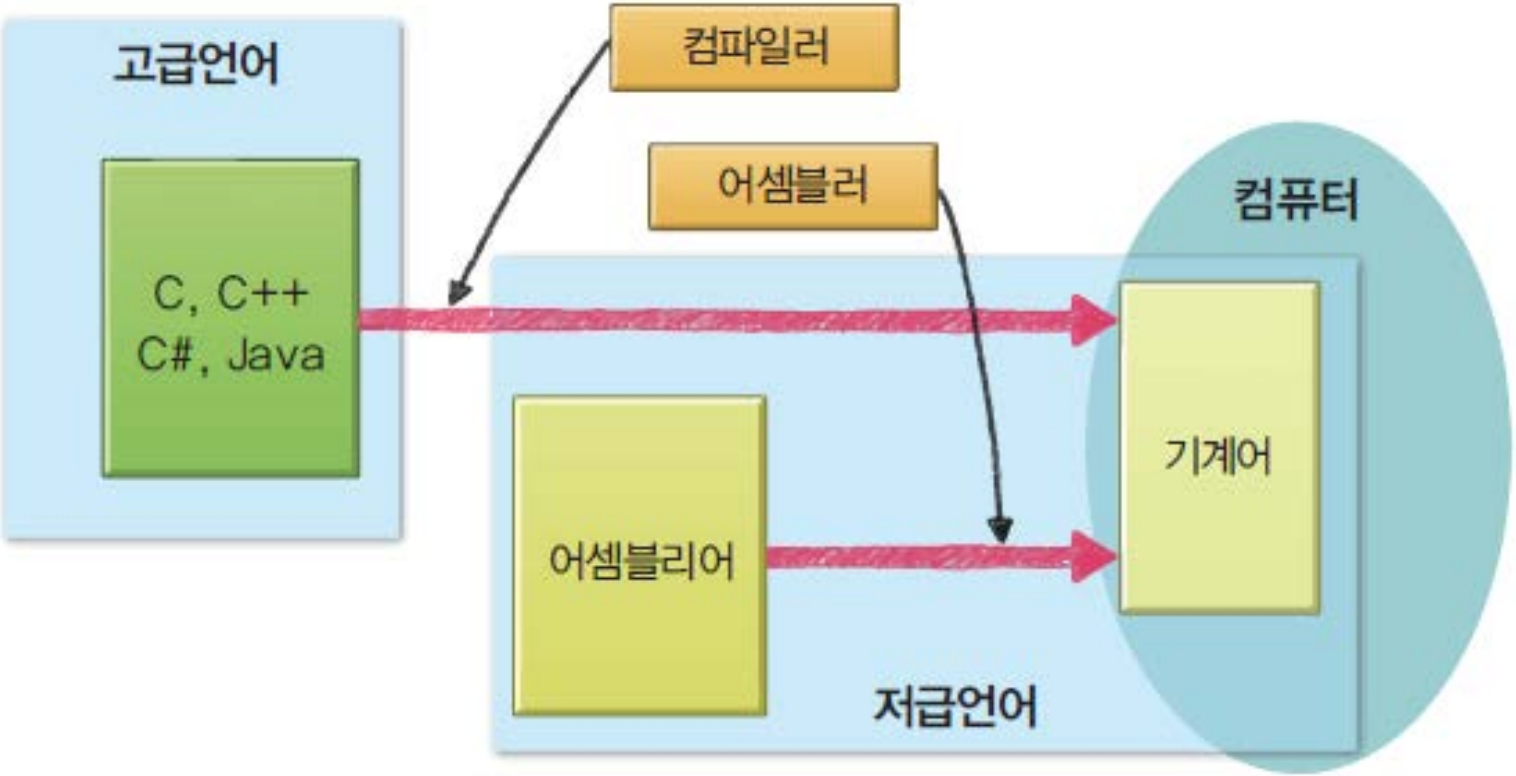
저급언어와 고급언어

- 저급언어
  - 컴퓨터의 중앙처리장치CPU에 적합하게 만든 기계어와 어셈블리 언어를 저급언어Low Level Language
  - 저급언어는 컴퓨터의 CPU에 따라 달라지며, 특정한 CPU를 기반으로 만들어진 언어
- 고급언어
  - 컴퓨터의 CPU에 의존하지 않고 우리 사람이 보다 쉽게 이해할 수 있도록 만들어진 언어를 고급언어High Level Language
  - 고급언어인 프로그래밍 언어로는 바로 우리가 배우고 있는 C 언어를 비롯하여 포트란, 파스칼, 베이직, C++, 자바, 파이썬 등 인간의 언어 만큼이나 매우 다양

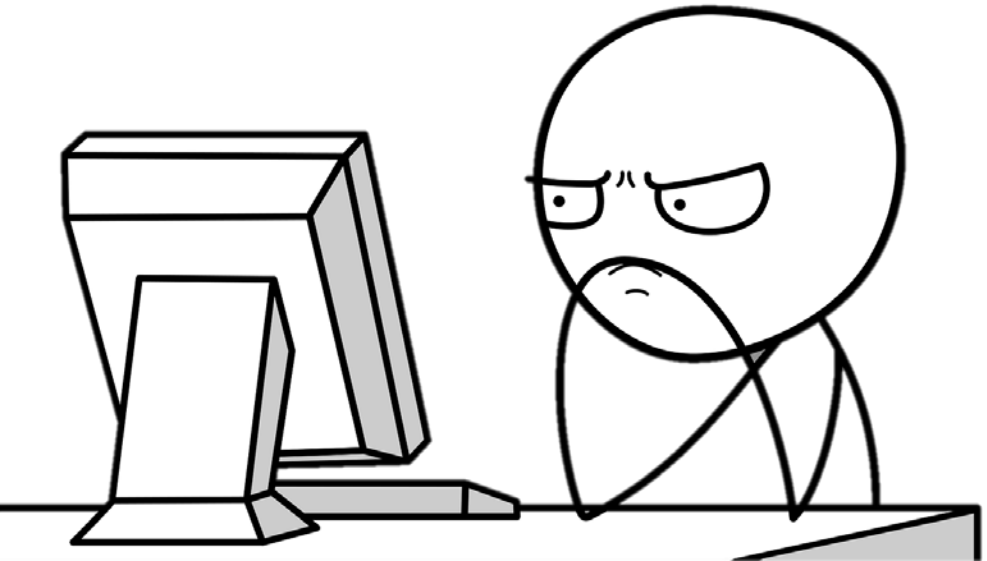


번역기

- 컴파일러compiler는 고급언어로 작성된 프로그램을 기계어 또는 목적코드object code로 바꾸어주는 프로그램
- 어셈블러assembler는 어셈블리어로 작성된 프로그램을 기계어로 바꾸어 주는 프로그램

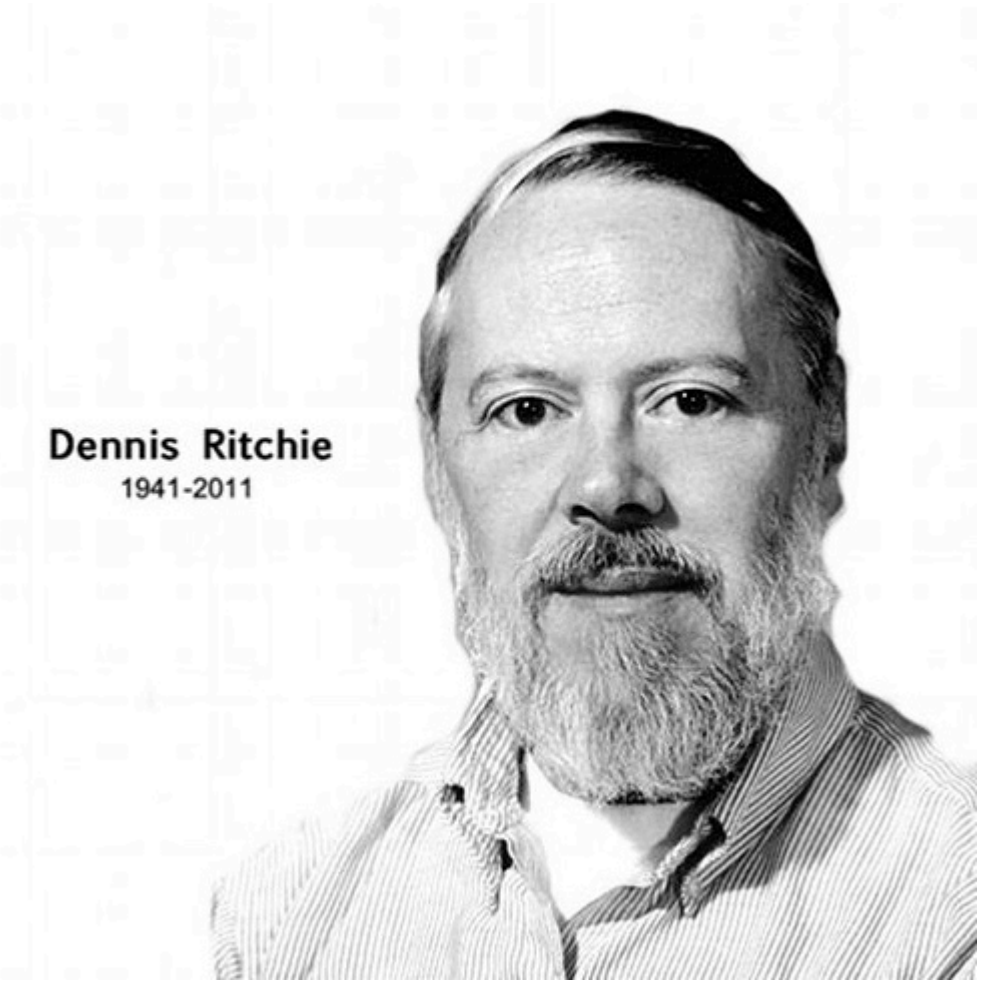


3. 왜 C 언어를 배워야 할까?



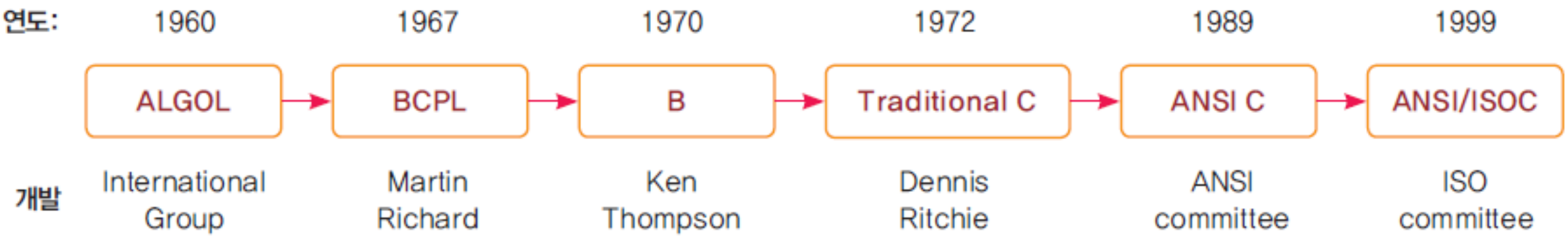
데니스 리치

- 1972년 미국전신전화국(AT&T)의 벨 연구소Bell Lab에 근무하던 데니스 리치는 시스템 PDP-11에서 운용되는 운영체제인 유닉스Unix를 개발하기 위해 C 언어를 개발
- 어셈블리 언어 정도의 속도를 내며, 좀 더 쉽고, 서로 다른 CPU에서도 작동되는 프로그래밍 언어가 필요



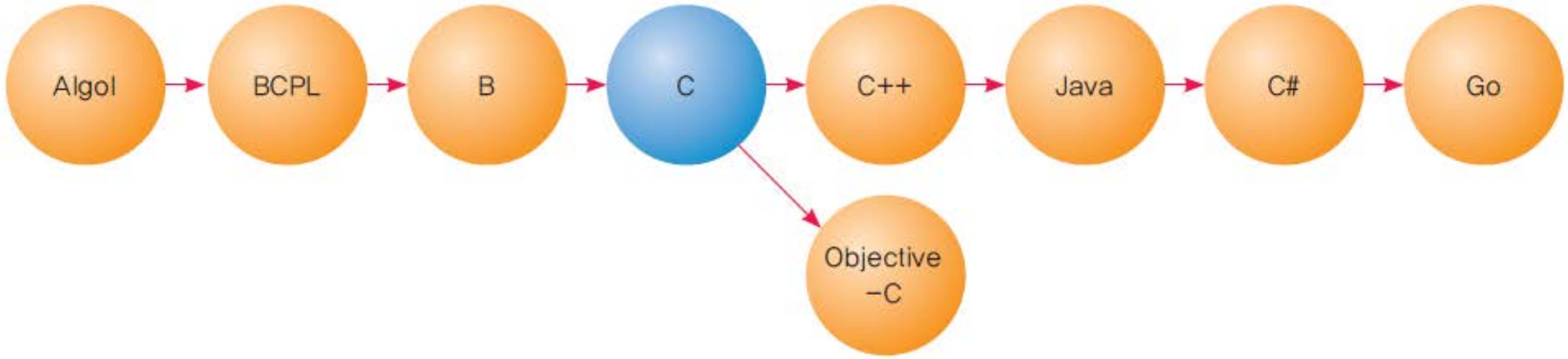
B 언어에서 발전

- 켄 톰슨이 1970년 개발한 B 언어에서 유래
- B언어 1970년 BCPL 프로그래밍 언어에 기반을 두고 개발된 언어
- BCPL Basic Combined Programming Language은 캠브릿지 대학의 마틴 리차드 Martin Richards가 1969년 개발한 프로그래밍 언어로 CPL Combined Programming Language 언어에서 발전
- 1960년 초에 개발된 CPL은 1960년에 개발된 Algol 60으로부터 많은 영향을 받은 언어



C 이후의 프로그래밍 언어 발전

- 1983년 얀 스트로스트럽Bjarne Stroustrup은 C 언어에 객체지향프로그래밍 개념을 확장시킨 프로그래밍 언어 C++를 개발
- 1995년에는 선 마이크로시스템즈Sun Microsystems 사는 C++ 언어를 발전시켜 인터넷에 적합한 언어인 자바Java를 발표



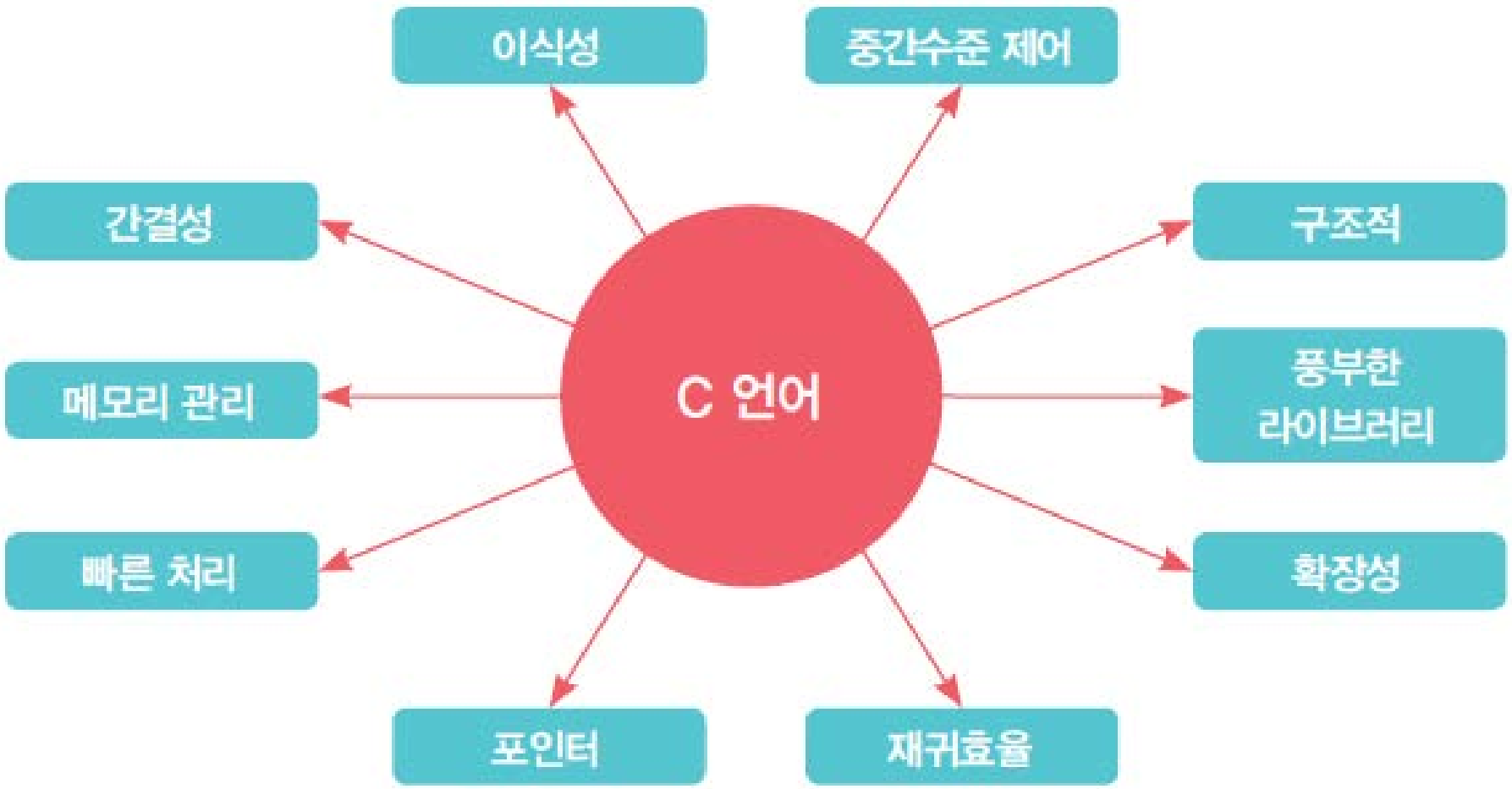
Note: C와 유닉스를 개발한 데니스 리치

- 유명한 현대 컴퓨터 분야의 선구자, 하버드 대학교에서 물리와 응용수학 전공
- 동료인 켄 톰슨과 함께 1983년에 '범용 운영체제 이론개발과 유닉스 운영체제의 구현에 대한 공로'로 컴퓨터 분야의 노벨상이라 불리는 튜링상Turing Awards을 수상, 1998년에 미국 과학기술상US National Medal of Technology을 수상

Note: 튜링 머신과 봄베를 개발한 앨런 튜링

- 영국의 수학자이자 컴퓨터 과학자인 앨런 튜링Alan Turing, 1912~1954은 원칙적으로 해결이 불가능한 문제가 무한히 많다는 사실을 입증
- 컴퓨터 과학의 계산이론 분야와 알고리즘 분야, 그리고 인공지능 분야 등에서 많은 업적을 기려, 앨런 튜링을 컴퓨터의 아버지라 부름
- 튜링 머신
  - 명령어와 프로그램에 의해 작동
  - 모든 계산 가능한 문제들을 해결할 수 있는 가상의 기계 장치를 고안
  - 실제 기계가 아닌 개념적인 가상의 장치로, 후에 그의 이름을 따서 튜링 기계Turing Machine라 불림
- 튜링의 전기를 다룬 영화
  - 2015년에 2월에 국내에서 개봉한 영화 '이미테이션 게임'
  - 봄베BOMBE: 2차 세계 대전 동안 런던 근교의블레츨리 파크Bletchley Park에서 독일의 에니그마 기계Enigma Machine의 암호를 풀기 위해 만든 전자 계산기 시스템
  - 앨런 튜링의 팀이 개발한 봄베는 독일의 에니그마 암호 코드를 풀어 연합군이 2차 세계 대전을 승리하는 결정적 공헌

C 언어의 특징



C 언어의 특징

절차지향 언어	간편하고 효율적인 언어	이식성이 좋은 언어	다소 어렵다는 단점
• 함수 중심으로 구현되는 절차지향 언어procedural language • 시간의 흐름에 따라 정해진 절차를 실행한다는 의미 • 문제의 해결 순서와 절차의 표현과 해결이 쉽도록 설계된 프로그램 언어	• 크기도 작으며, 메모리도 적게 효율적으로 사용하여 실행 속도가 빠르다는 장점	• C로 작성된 소스는 별다른 수정 없이 다양한 운영제제의 여러 플랫폼에서 제공되는 컴파일러로 번역compile해 실행 • 다양한 CPU와 플랫폼의 컴파일러를 지원	• Don't worry…

C 언어를 배워야 하는 이유

많은 언어에 영향을 미친 가장 기본이 되는 프로그래밍 언어

- 많이 쓰이는 자바나 C#, Objective-C 등이 그 뿌리는 C
- C 언어를 알면 자바나 C#, Objective-C 뿐만 아니라 그 이후의 프로그래밍 언어들은 가지를 뻗어나가듯 습득이 매우 쉬워짐

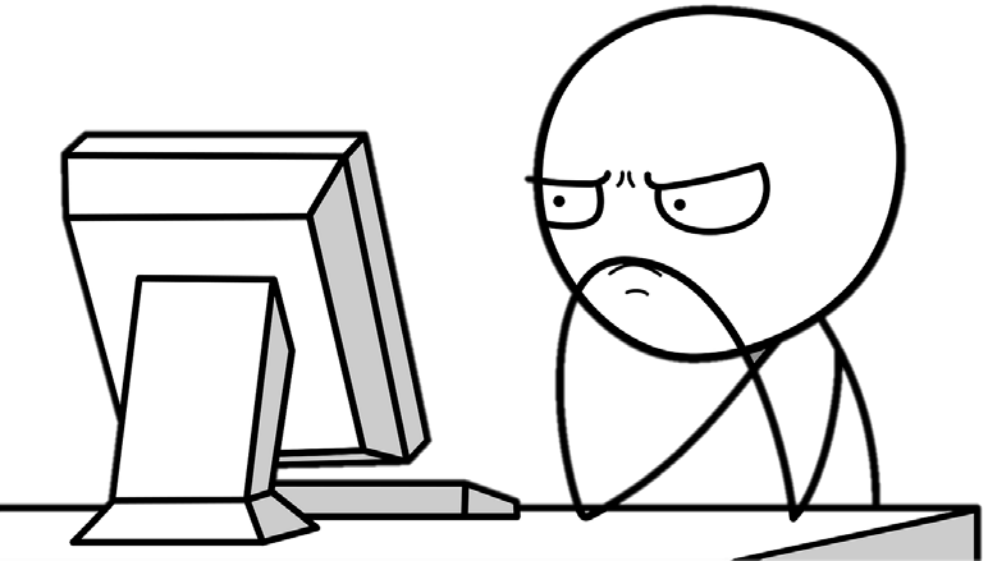
아직도 현장에서 다양한 분야에 사용되는 범용적인 프로그래밍 언어

- 임베디드시스템embedded system에서부터 응용프로그램application program, 운영체제와 같은 시스템 소프트웨어system software 개발 등 여러 분야에 널리 사용

프로그래밍 지식과 프로그래밍 방법을 학습

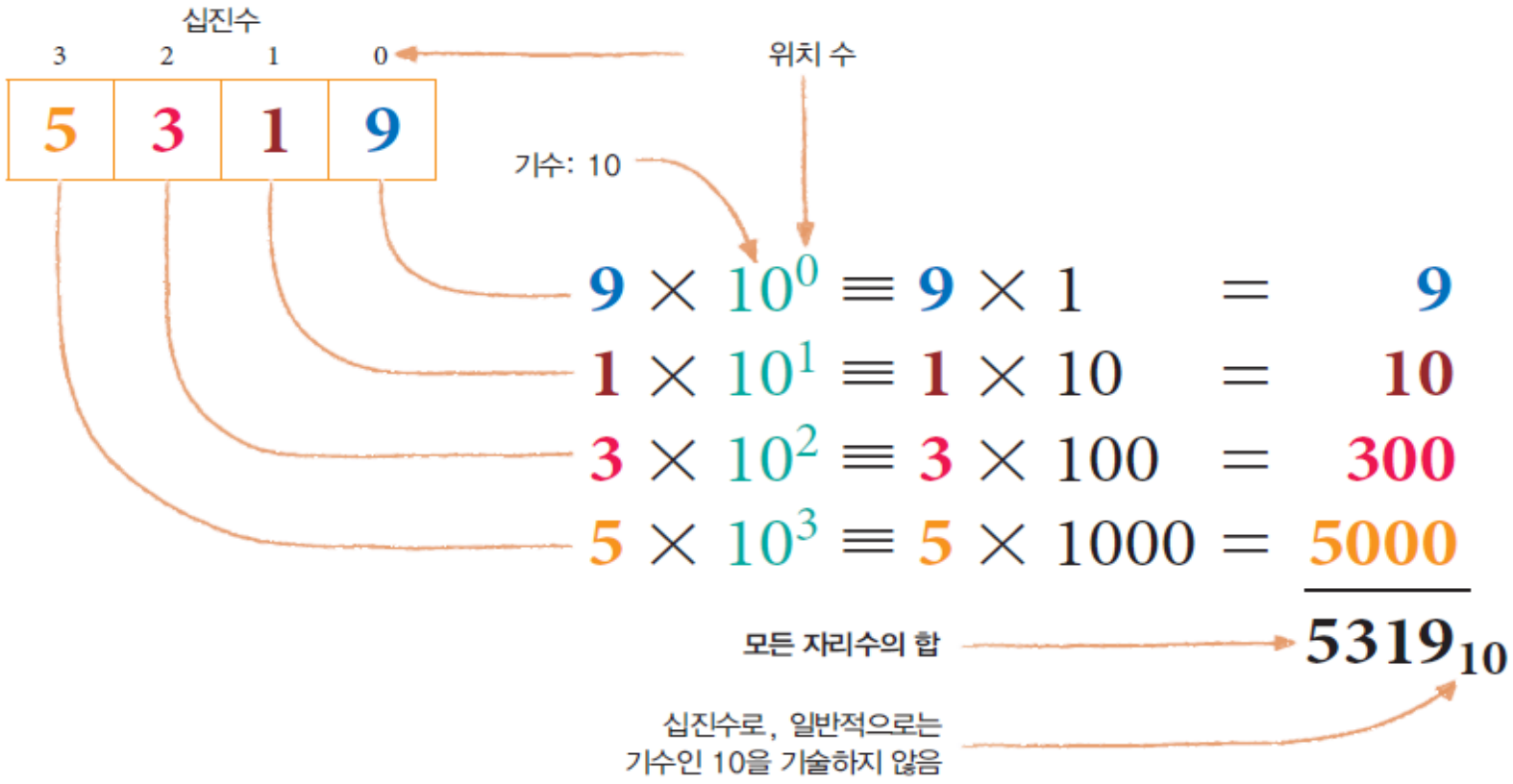
- 단순히 C 언어의 문법뿐 아니라 일반적인 프로그래밍 기초 지식을 학습
- C 언어의 학습은 여러분이 프로그래머나 정보기술 개발자가 되기 위한 초석을 다지는 기회

4. 프로그래밍의 자료 표현



우리에게 친숙한 십진수

- 십이라는 것을 기수base
  - 수에서 하나의 자릿수digits에 사용하는 숫자가 0, 1, 2, 3, 4, 5, 6, 7, 8, 9까지 열 개이므로 십진수
  - 십진수 5319: 1000(103)인 것이 5개, 100(102)인 것이 3개, 10(101)인 것이 1개, 마지막으로 1(100)인 것이 9개 모인 수

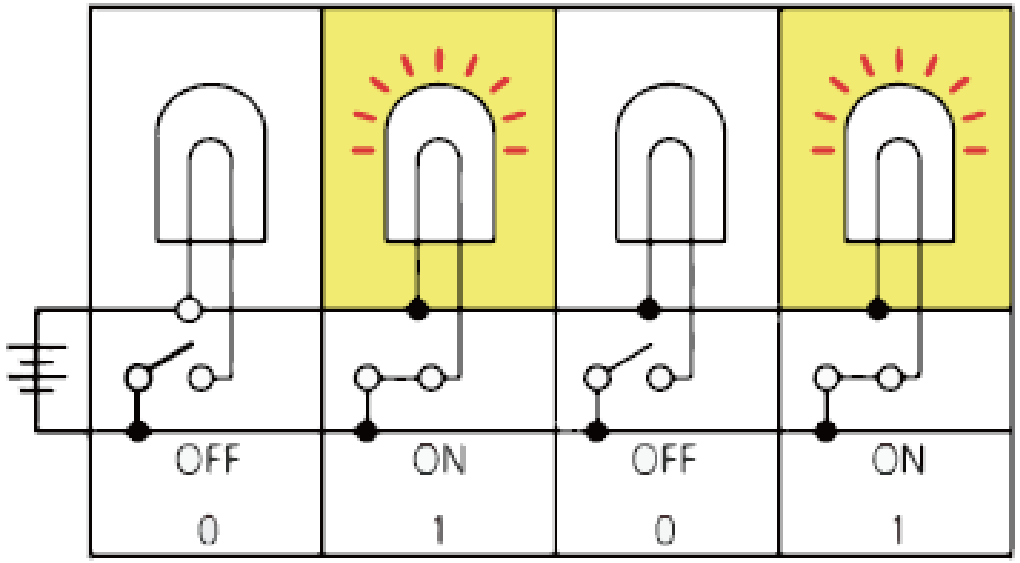


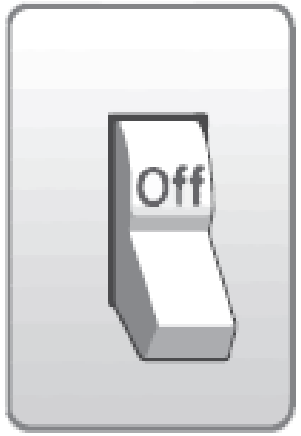
Question #01

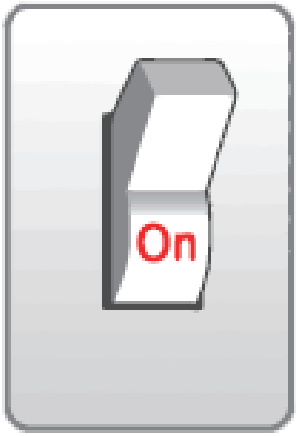
- 우리는 왜 십진수가 편할까?
- 십진수 외에 우리가 자주 사용하는 다른 진수는 무엇이 있을까?

0과 1로 표현되는 이진수 체계

- 전기가 흐를 경우 '참'을 의미하는 '1', 흐르지 않을 경우 '거짓'의 '0'으로 표현
- 컴퓨터는 논리의 조합이 간단하고 내부에 사용되는 소자의 특성상 편리하기 때문에 이진법을 사용하는 것이 가장 합리적이고 효율적인 방식

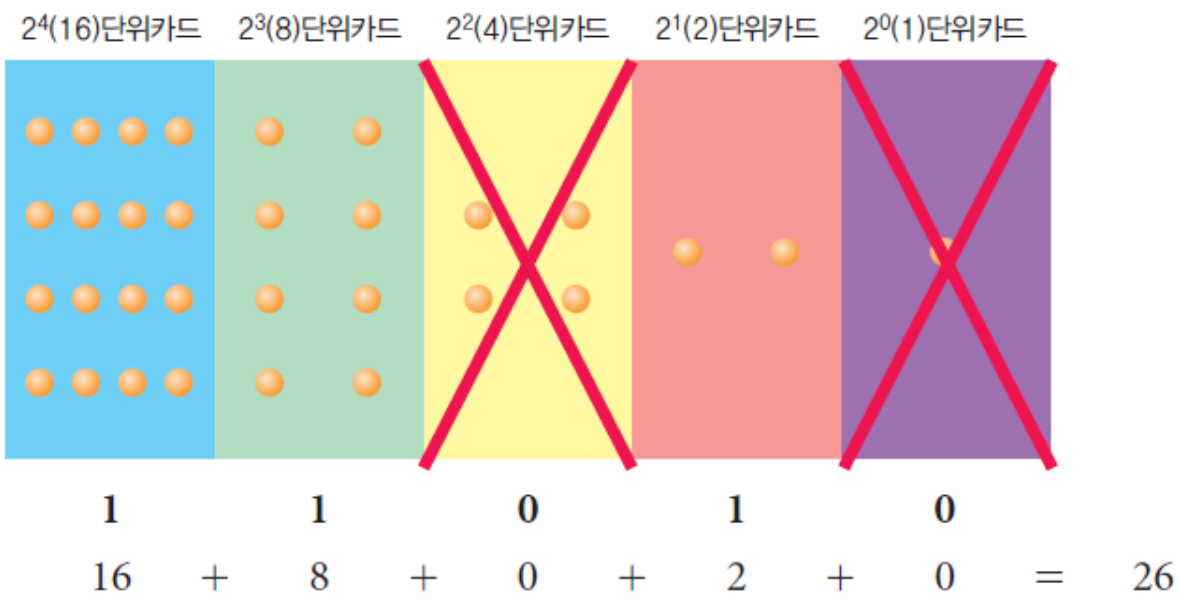


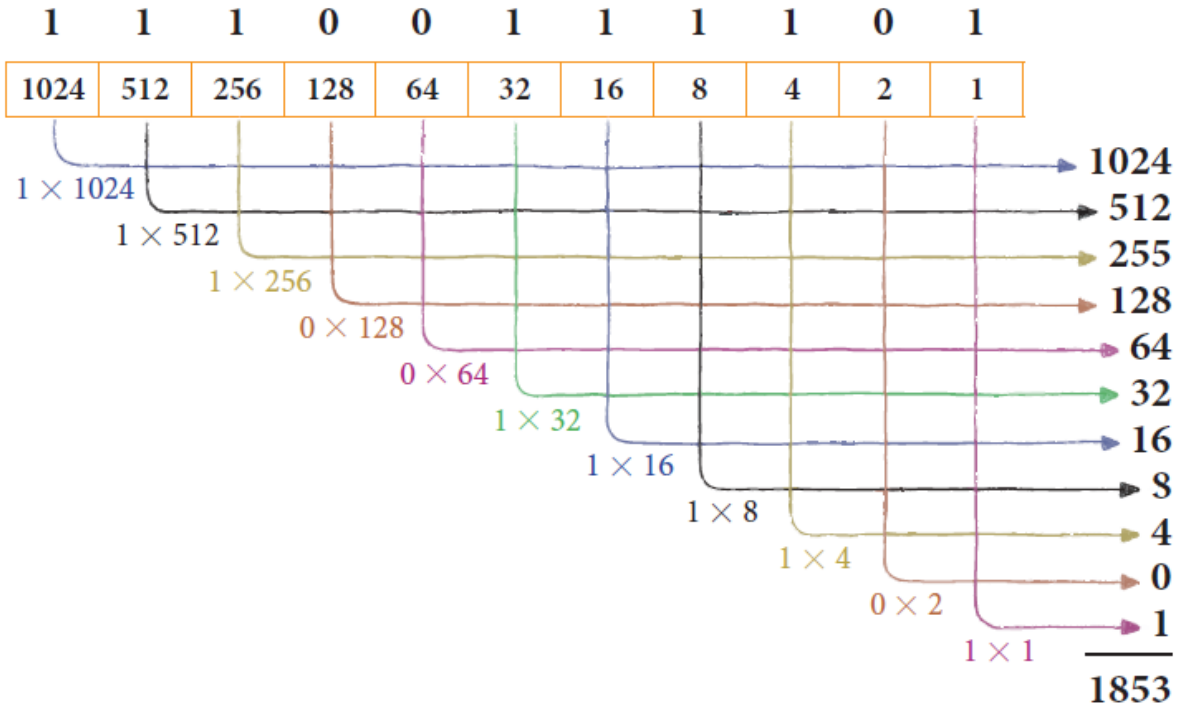




이진수의 이해

- 수의 자릿수에 사용할 수 있는 숫자가 0과 1, 2개 이므로 이진수
- 11010은 1, 2, 4, 8, 16 자릿수의 카드로 표현
- 이진수 11100111101은 1853



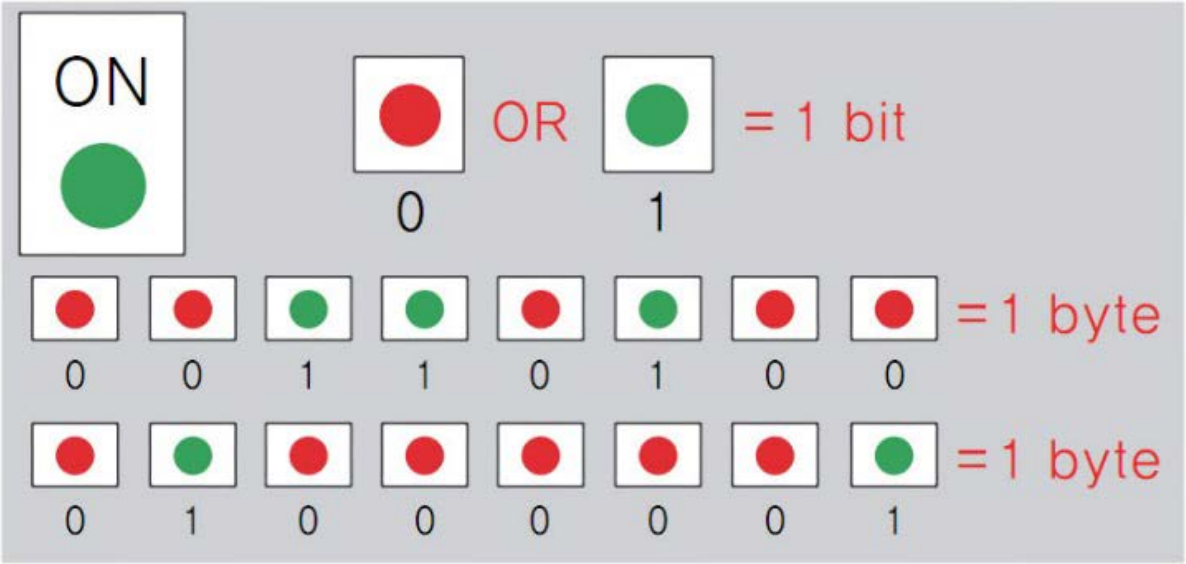


십간 십이지

- 우리나라와 중국 등 동양에서 갑(甲), 을(乙), 병(丙), 정(丁), 무(戊), 기(己), 경(庚), 신(辛), 임(壬), 계(癸)의 십간은 십진수
- 쥐, 소, 호랑이, 토끼 등 십이지는 자(子), 축(丑), 인(寅), 묘(卯), 진(辰), 사(巳), 오(午), 미(未), 신(申), 유(酉), 술(戌), 해(亥)로 12진수
- 태어나서 만으로 60세 생일이 되는 해는 자신이 태어난 해와 같은 간지가 되는 해가 됨, 그 해를 갑자를 돌아온다고 하여 환갑(還甲) 또는 회갑(回甲)이라고 함
- 십간 십이지를 조합한 간지는 60진법을 의미

비트

- 가장 작은 기본 정보 단위
- 즉 전기의 흐름 상태인 온on과 오프off를 표현하는 단위가 비트로 1과 0인 이진수로 표현이 가능
- 비트bit는 BInary digiT의 합성어



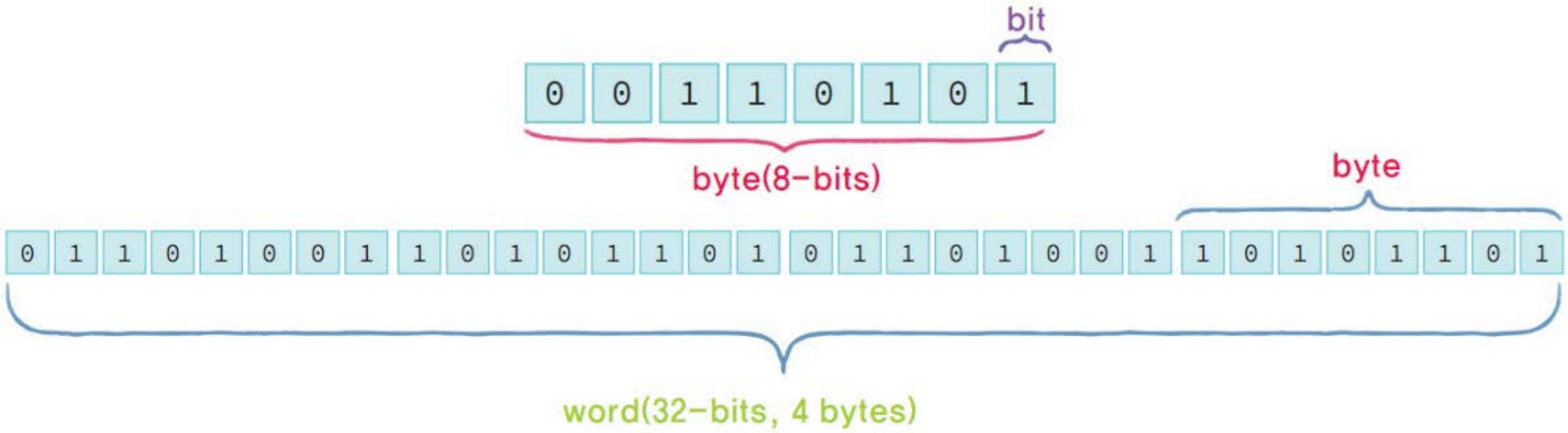
바이트

- 바이트가 정보 용량의 단위, 킬로, 메가, 기가, 테라 등은 그 크기를 표현
- 킬로Kilo는 2^{10}을 의미하며 1024

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 milion 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 Zeatbyte	1 ZB	$(2^{10})^7$ bytes	1,180,591,620,717,411,303,424 bytes	1 septillion bytes

바이트와 워드

- 8개의 비트가 모인 정보 단위를 바이트byte
- 비트는 총 28=256가지의 정보 종류를 저장
- 일반적으로 바이트가 4개, 8개 모이면 워드word
- 시스템마다 그 크기는 다를 수 있음



논리 표현과 연산

- 논리값
  - 0과 1을 각각 거짓과 참으로 표현 가능
  - 부울 대수Boolean Algebra
    - 영국의 수학자 조지 부울George Boole이 제창한 기호 논리학으로 컴퓨터가 정보를 처리하는 방식에 대하여 이론적인 배경을 제공
- 논리 연산
  - 기본적으로 AND, OR 연산과 NOT 연산 제공

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

A	B	A nand B
0	0	1
0	1	1
1	0	1
1	1	0

NAND

A	B	A nor B
0	0	1
0	1	0
1	0	0
1	1	0

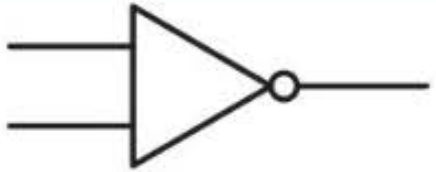
NOR

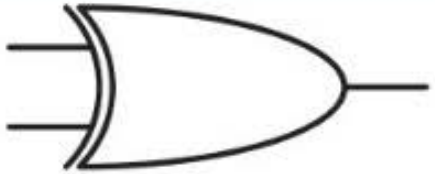
A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

XOR

디지털 회로 구성

- 게이트 등을 조합

Gate	Symbol	Operator
and		$A \cdot B$
or		$A + B$
not		$\overline{A}$

Gate	Symbol	Operator
nand		$\overline{A \cdot B}$
nor		$\overline{A + B}$
xor		$A \oplus B$

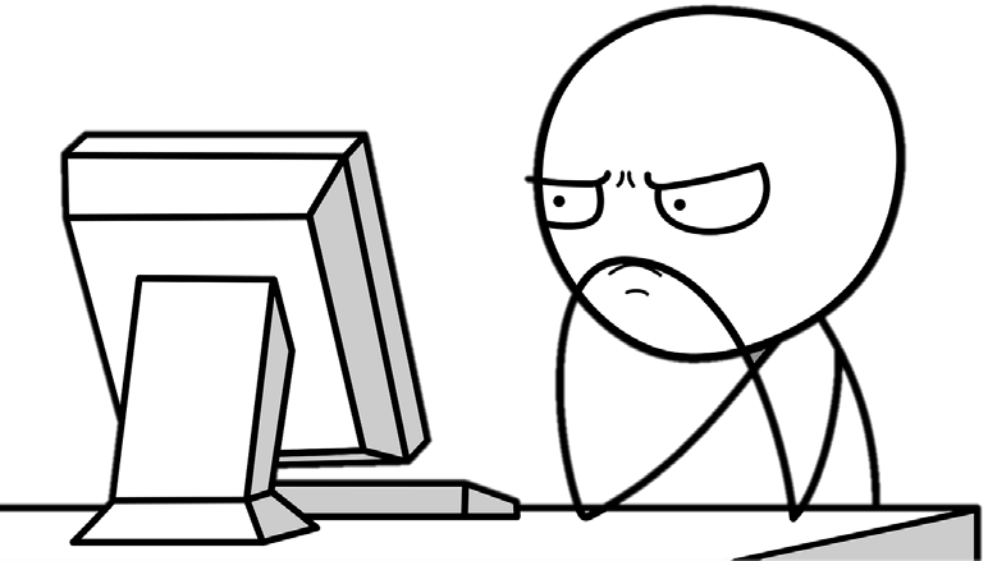
아스키코드

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

유니코드

- 유니코드는 전 세계의 모든 문자를 컴퓨터에서 일관되게 표현하고 다룰 수 있도록 설계된 산업 표준
- 유니코드 협회Unicode Consortium가 제정하여 1991년 버전 1.0이 발표
- 동양권의 컴퓨터 관련 시장을 쉽게 접근하기 위해서도 미국 등의 유수의 SW, HW업체에게 문자 코드 문제는 가장 시급하게 해결되어야 할 걸림돌
- 전 세계의 문자를 모두 표현하기 위해 2바이트 즉, 16비트로 확장된 코드 체계가 유니코드
- 1996년 65,536자의 코드영역을 언어학적으로 분류
- 한글 완성자 11,172자의 한글과 중국, 일본을 포함해 세계 유수의 언어 문자를 배열해 만든 유니코드는 국제표준화기구ISO: International Organization for Standardization에 상정 확정
- 현재 계속 수정, 보완

5. 소프트웨어 개발



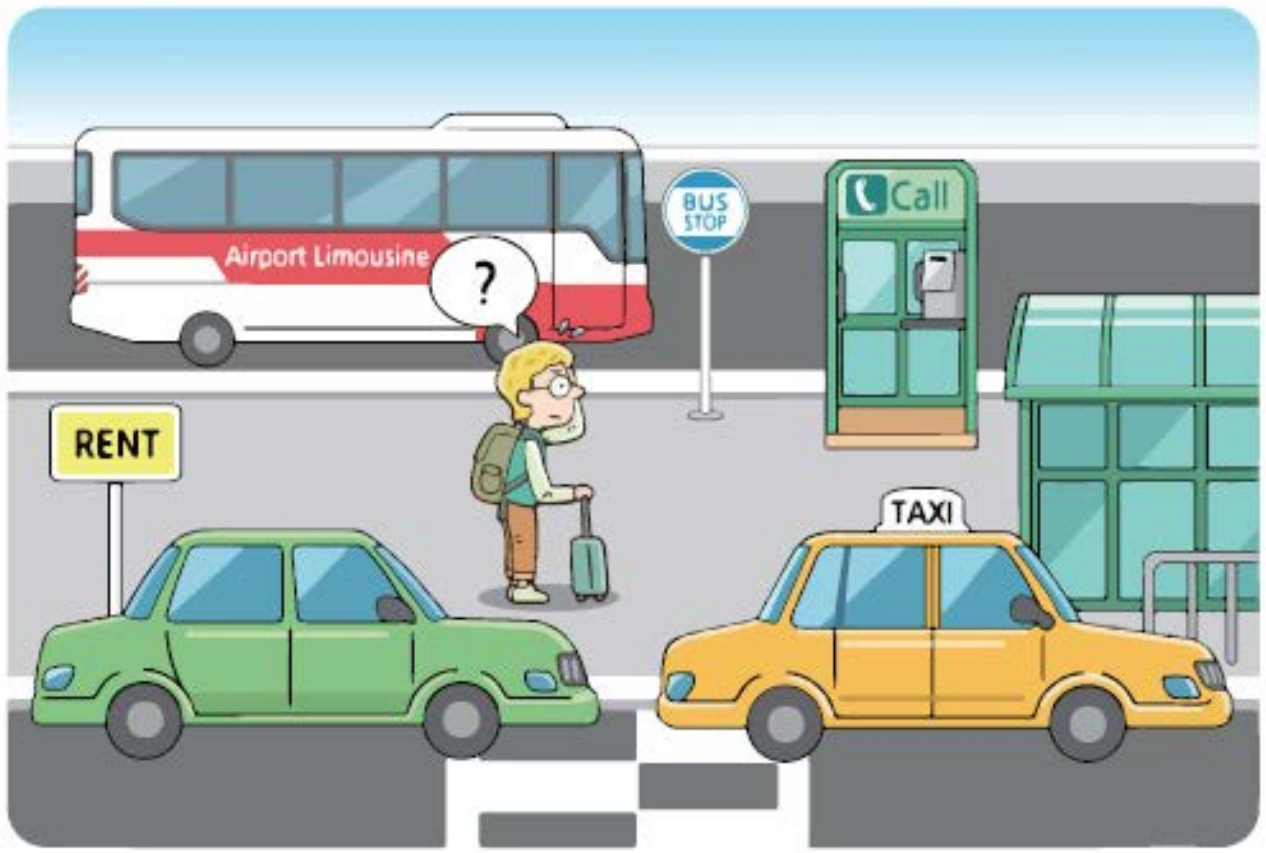
소프트웨어 교육의 중요성

- 소프트웨어 교육은 프로그래밍 언어 습득 자체보다 사고를 절차화하는 과정을 통해 논리력과 문제를 해결하는 능력 등 디지털 시대에 필요한 창의적 사고를 할 수 있는 기본 소양을 증진
- 세계는 지금 혁신과 성장, 가치창출이 중심이 되고 개인·기업·국가의 경쟁력을 좌우하는 '소프트웨어 중심 사회'SOS, Software Oriented Society로 급속히 발전
  - 2017년에는 소프트웨어 교육 과목을 초등학교 정규 교과로 편성
  - 2018년에는 중고등학교 정규 교과로 편성 예정

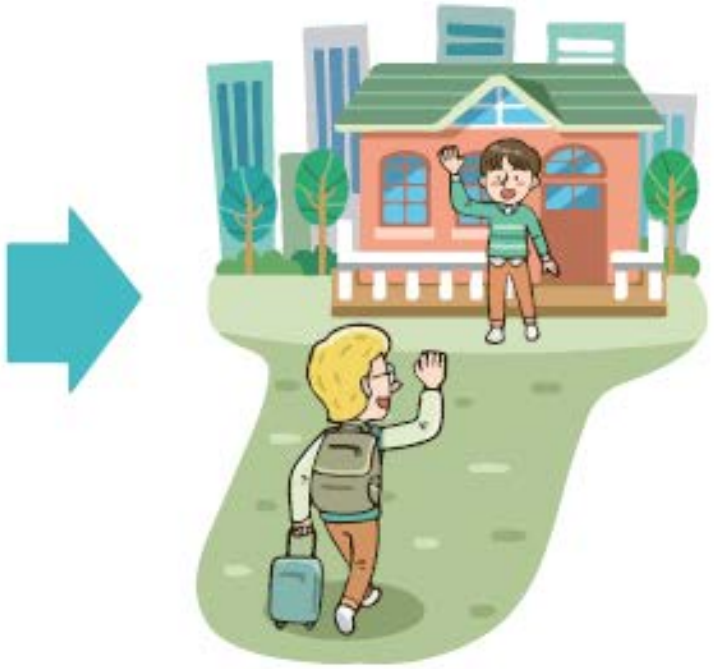
알고리즘

- 어떠한 문제를 해결하기 위한 절차나 방법으로 명확히 정의된well-defined 유한 개의 규칙과 절차의 모임
- 프로그램에서 문제를 해결하기 위한 여러 명령을 단계적으로 알려주어야 하는데, 이것이 바로 '알고리즘'
- 컴퓨터 프로그램: 특정한 업무를 수행하기 위한 정교한 알고리즘들의 집합

알고리즘 예제



출발지에서 목적지까지 대중교통으로 가는
방법을 찾는 명령 모임의 알고리즘



알고리즘 예제

첫 번째는 버스를 타는 것이다.

1. 수화물을 찾은 다음 1120번 버스를 탄다.
2. 종로에서 250번 버스로 갈아탄다.
3. 홍대입구에서 내린다.
4. 9번 출구로 나와서 500 미터쯤 걸어서 도착한다.

두 번째는 내게 전화하는 것이다.

1. 공항에 도착해서 내게 전화한다.
2. 수화물을 찾은 다음에 공항에서 나를 만난다.

세 번째는 렌트카를 사용하는 것이다.

1. 렌터카 회사까지 셔틀을 타고 간다.
2. 차를 빌린다.
3. 우리 집까지 경로를 따라 차를 몰고 온다.

네 번째는 택시를 이용하는 것이다.

1. 택시 승강장에서 택시를 기다린다
2. 택시를 탄다.
3. 택시기사에게 우리집 주소를 알려준다.

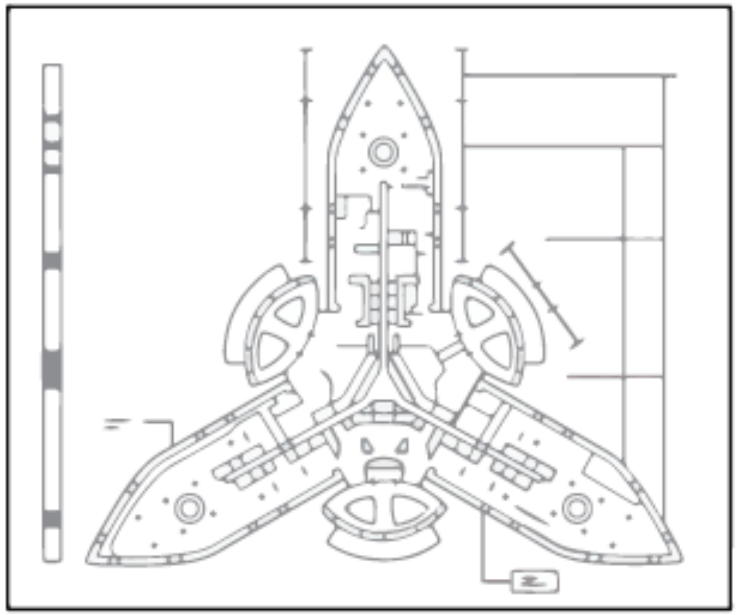
소프트웨어 공학software engineering

- 소프트웨어 공학이란 공학적 원리에 의하여 소프트웨어를 개발하는 학문
- 소프트웨어 개발과정인 분석, 설계, 개발, 검증, 유지보수 등 개발수명주기 전반에 걸친 계획·개발·검사·보수·관리, 방법론 등을 연구하는 분야

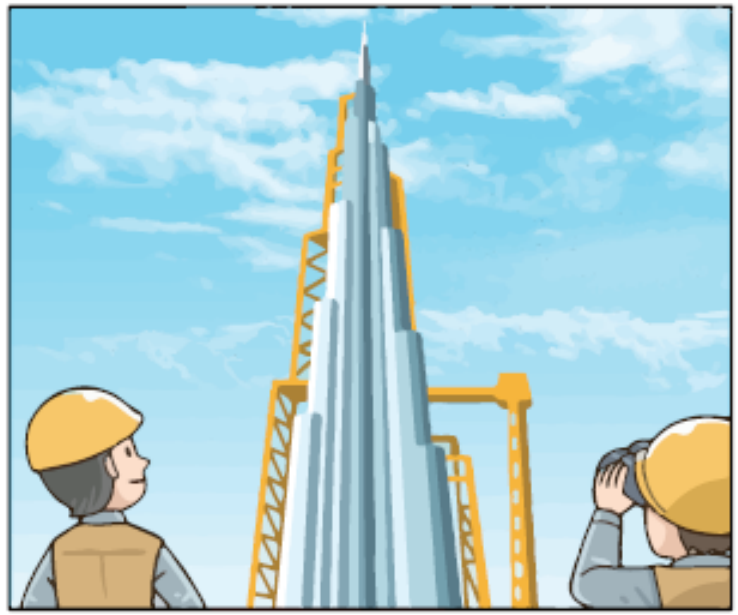
건물 건축과정

부르즈 할리파(Burj Khalifa)

❶ 기획 단계



❷ 설계 단계



❸ 시공 단계

소프트웨어 공학 단계

❶	Requirements Analysis(요구사항 분석)	Requirements specification	요구사항 또는 해결할 문제를 먼저 파악한다.
❷	Design(설계)	System architecture	프로그램을 설계한다.
❸	Implementatin(구현)	Software	설계된 알고리즘에 따라 코딩한다.
❹	Verification(검증)	System testing	작성된 프로그램을 테스트한다.
❺	Maintenance(유지보수)	System support	프로그램을 문서화하고 유지 보수한다.

알고리즘 기술 방법: 의사코드pseudo code

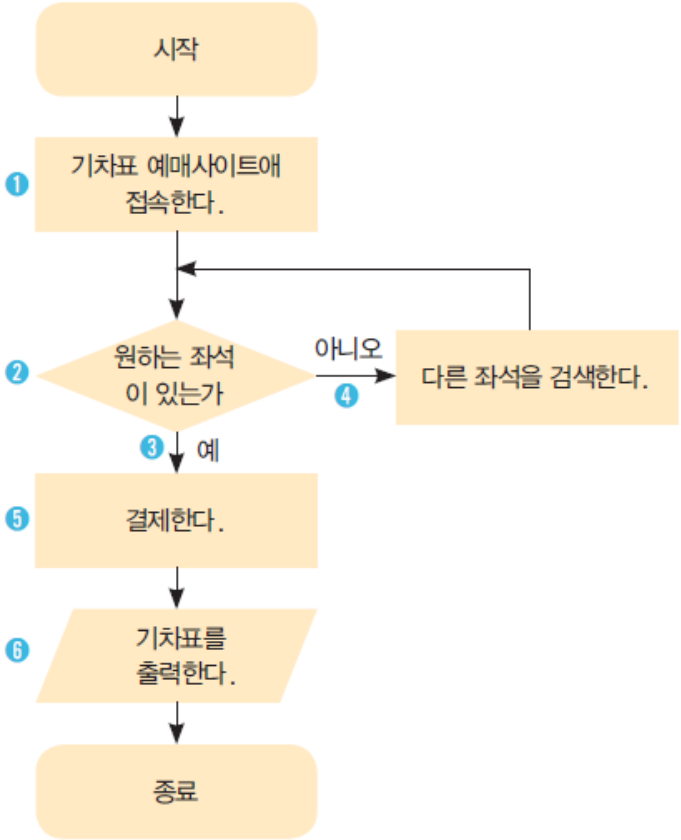
- 의사코드는 슈도코드라고도 하며
- 특정 프로그래밍 언어의 문법을 따르지 않고 간결한 특정 언어로 코드를 흉내 내어 알고리즘을 써놓은 코드
- 의사코드는 다양한 스타일의 언어가 존재

알고리즘 기술 방법: 흐름도flow chart

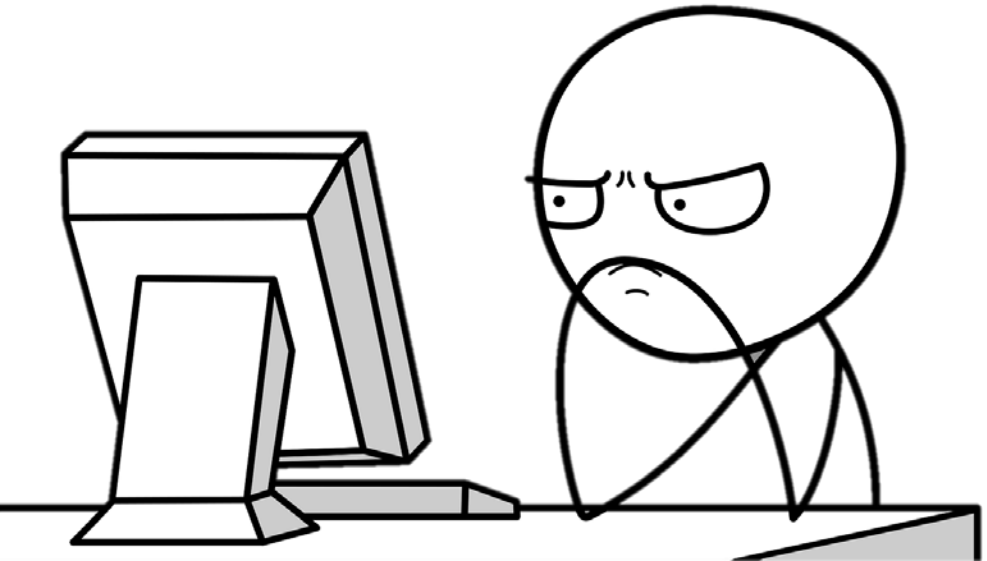
- 표준화된 기호 및 도형으로 도식화하여 데이터의 흐름과 수행되는 연산들의 순서를 표현하는 방법

기호	기능	기호	기능
	터미널 순서도의 시작과 끝을 표시		**처리** 각종 연산, 데이터 이동 등을 처리
	판단 여러 가지 경로 중 하나의 경로 선택을 표시		**입 · 출력** 데이터의 입력 및 출력 표시
	흐름선 처리간의 연결 기능을 표시		**연결자** 흐름이 다른 곳과 연결되는 입출구를 나타냄
	서류 서류를 매체로하는 입출력 표시		**준비** 기억장소, 초기값 등 작업의 준비 과정을 나타냄
	수동입력 콘솔에 의한 입력		**천공카드** 천공카드의 입출력

시작
❶ 기차표 예매사이트에 접속한다.
❷ 원하는 시각과 역에 정차하는 좌석이 있는가?
❸ 있으면 4로 간다.
❹ 없으면 다시 2로 가서 다른 좌석을 검색한다.
❺ 결제한다.
❻ 구매 완료한 기차표를 출력한다.
종료



6. 다양한 '프로그래밍 언어'



포트란FORTRAN

- 과학과 공학 및 수학적 문제들을 해결하기 위해 고안된 프로그래밍 언어
- 널리 보급된 최초의 고급 언어
- FORmula TRANslating system(수식 번역 시스템)의 약자
- 수학에서 사용하는 +, - , *, / 와 같은 사칙연산 기호와 sin,cos,tan의 삼각함수를 비롯하여 log, abs 등 다양한 수학함수들을 프로그램 내에서 그대로 사용
- 어셈블리 언어에 익숙해져 있던 1957년경, 포트란은 IBM에서 존 배커스(John Backus) 등의 전문가가 개발한 프로그래밍 언어
- FORTRAN은 가장 오래된 언어지만 언어 구조가 단순해 놀라운 생명력을 갖추고 있어 지금도 과학 계산 분야 등에서는 많이 사용

코볼COBOL

- 학계 위원과 허니웰, GE, 뷰로우, RCA, IBM등의 업체가 참여한 협회 CODASYL이 1960년, 사무처리에 적합한 프로그래밍 언어로 개발한 것이 코볼COmmon Business Oriented Language
- 포트란에 이어 두 번째로 개발된 고급 언어
- 대량의 데이터를 효율적으로 입력, 출력 및 처리
- 코볼 컴파일러만 있으면 어떠한 컴퓨터 기종이라도 코볼 프로그램을 작성하여 실행 가능
- 코볼은 사무처리에 목적이 있으므로 다른 프로그래밍 언어에 비하여 파일이나 데이터베이스에서 데이터를 쉽게 읽고 쓰며, 또한 양식을 가진 보고서를 쉽게 만들 수 있는 등 사무처리에 효율적
- FORTRAN이 수식과 비슷한 반면, 코볼은 일상 영어회화와 비슷한 구어체 문장 형태를 갖고 있으므로 쉽게 이해할 수 있도록 프로그램 작성이 가능

알골ALGOL

- 알고리즘ALGOrithm을 표현하기 위한 언어로 ALGOrithmic Language를 줄여서 만든 이름
- 포트란이 미국을 중심으로 사용했다면 알골은 유럽을 중심으로 과학기술 계산용 프로그래밍 언어로 사용
- 알고리즘의 연구 개발에 적합한 언어로 개발
- 절차적 언어로서 구조화 프로그래밍에 적합하고, 최초로 재귀호출이 가능한 프로그래밍 언어
- 알골은 파스칼, C 언어 등 이후 언어의 발전에 큰 영향을 주었으나, IBM이 주로 포트란을 사용하여 더 이상 대중화에 성공하지 못함

베이직BASIC

- 1964년에 미국 다트머스Dartmouth 대학의 켐니John Kemeny 교수와 커쯔Thomas Kurtz 교수가 개발
- 베이직BASIC은 'Beginner's All-purpose Symbolic Instruction Code'(초보자의 다목적용이고 부호를 사용하는 명령어 코드)의 약어
- 초보자도 쉽게 배울 수 있도록 만들어진 대화형 프로그래밍 언어
- 대화형의 영어 단어를 바탕으로 약 200여 개의 명령어들로 구성된 가장 쉬운 대화형 프로그래밍 언어
- 문장의 종류가 많지 않고 문법이 간단하며, 배우고 쓰기가 간단하고 쉬움
- 컴파일 없이 바로 작동되나 인터프리터를 거쳐야 하므로 실행 속도가 느리다는 단점

베이직의 발전

- 1980년대에 개인용 컴퓨터의 출현과 함께 베이직은 기본 개발 언어로 탑재되어 범용적인 언어로 널리 사용
- 마이크로소프트는 이 베이직을 기본으로 비주얼 베이직Visual Basic이라는 프로그램 언어를 개발
- 비주얼 베이직은 표준 베이직에 객체지향 특성과 그래픽 사용자 인터페이스를 추가한 프로그램 언어이자 통합개발환경
- 웹 프로그래밍 기술에서 많이 사용하는 언어 중 하나인 ASPActive Server Page도 VBScript를 사용하는데, VBScript도 베이직 문법을 그대로 사용

파스칼Pascal

- 파스칼은 프랑스의 수학자인 파스칼(Pascal)의 이름에서 따온 언어
- 프로그램을 작성하는 방법인 알고리즘 학습에 적합하도록 1971년 스위스 취리히 공과대학교의 니클라우스 비르트Nicholas Wirth 교수에 의해 개발된 프로그래밍 언어
- 파스칼은 알골Algol을 모체로 해서 초보자들이 프로그램의 문법적 에러를 줄일 수 있도록 매우 엄격한 문법을 가진 프로그래밍 언어
- 구조적인 프로그래밍structured programming이 가능하도록 begin~end의 블록 구조가 설계
- 1980년에서 1990년대까지 대부분의 대학에서 프로그래밍 언어의 교과과정으로 파스칼을 채택
- 애플 사는 1980년 초, 파스칼 문법에 객체지향 기능을 추가시킨 오브젝트 파스칼Object Pascal 개발
- 1980년대에는 볼랜드 사에서 파스칼을 발전시켜 터보 파스칼Turbo Pascal이라는 제품으로 상용화
- 볼랜드 사는 1990년 중반에 마이크로소프트 사의 비주얼 베이직과 유사한 오브젝트 파스칼 언어를 기반으로 하는 델파이Delphi를 출시

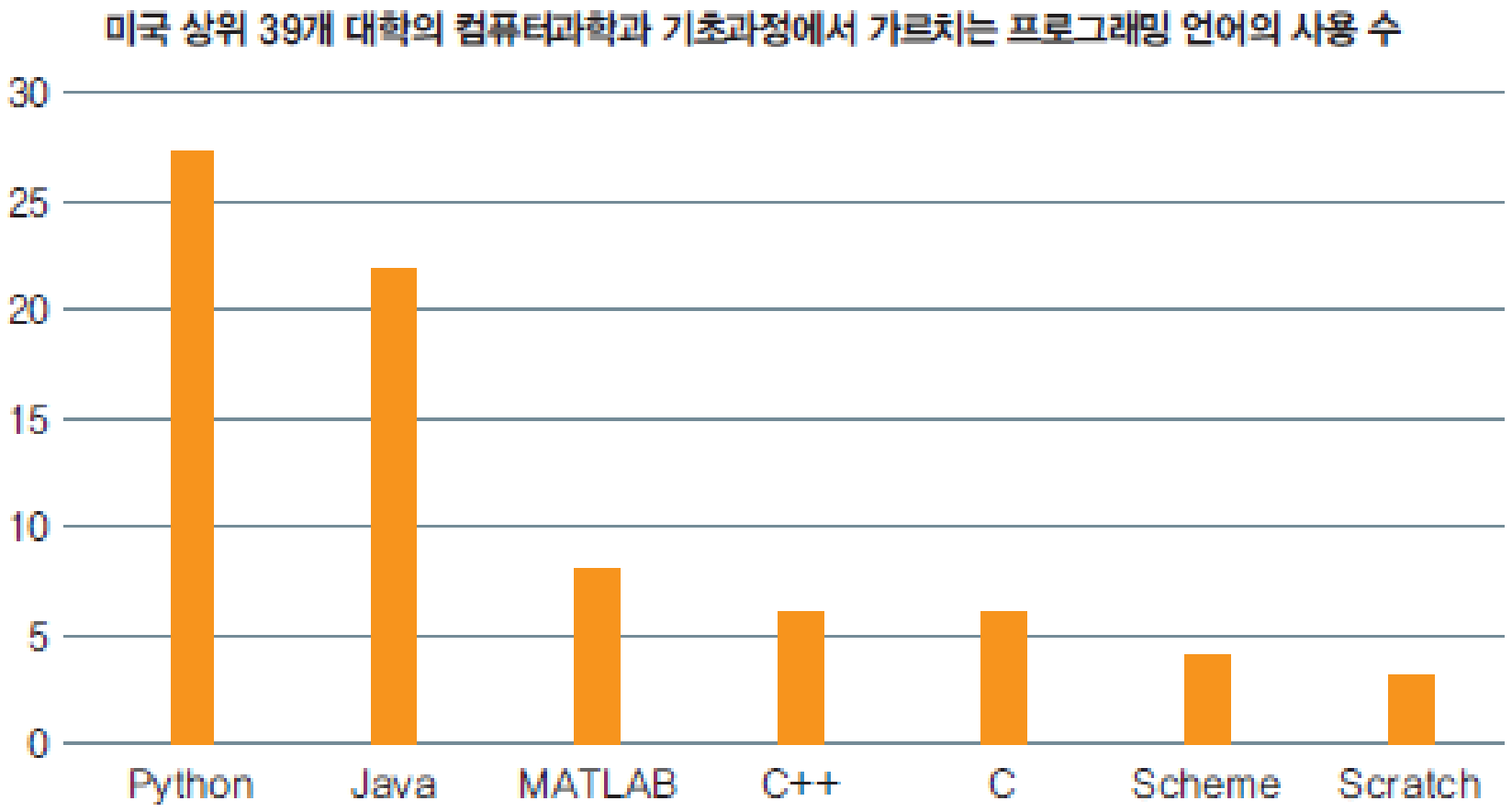
C++

- 1972년에 개발된 C 언어는 1983년에 프로그램 언어 C++로 발전
- C++는 객체지향 프로그래밍OOP: Object Oriented Programming을 지원하기 위해 C 언어가 가지는 장점을 그대로 계승하면서 객체의 상속성inheritance 등의 개념을 추가한 효과적인 언어
- AT&T의 얀 스트로스트럽Bjarne Stroustrup 이 개발
- C++언어가 C언어와 유사한 문법을 사용함으로써 C언어에 익숙한 프로그래머들이 C++언어를 쉽게 배울 수 있다는 장점

파이썬Python

- 현재 미국의 대학에서 컴퓨터 기초 과목으로 가장 많이 가르치는 프로그래밍 언어 중 하나
- 1991년 네덜란드의 귀도 반 로섬Guido van Rossum이 개발한 객체지향 프로그래밍 언어
- 비영리의 파이썬 소프트웨어 재단이 관리하는 개방형, 공동체 기반 개발 모델
- C#으로 구현된 닷넷프레임워크 위에서 동작하는 IronPython, Java로 구현되어 JVM위에서 돌아가는 Jython, 파이썬 자체로 구현된 PyPy 등 다양한 언어로 만들어진 버전 등 다양
- C언어로 구현된 C 파이썬cpython 구현이 사실상의 표준
- 프로그래밍 언어 파이썬이 대학의 컴퓨터기초 교육에 많이 활용
- 파이썬이 무료이며, 간단하면서 효과적으로 객체지향을 적용할 수 있는 강력한 프로그래밍 언어
- 베이직과 같은 인터프리터 언어로 간단한 문법구조를 가진 대화형 언어
- 쉽고 빠르게 개발할 수 있어, 개발기간이 매우 단축되는 것이 장점

미국 상위 39개 대학 기초과정 프로그래밍 언어

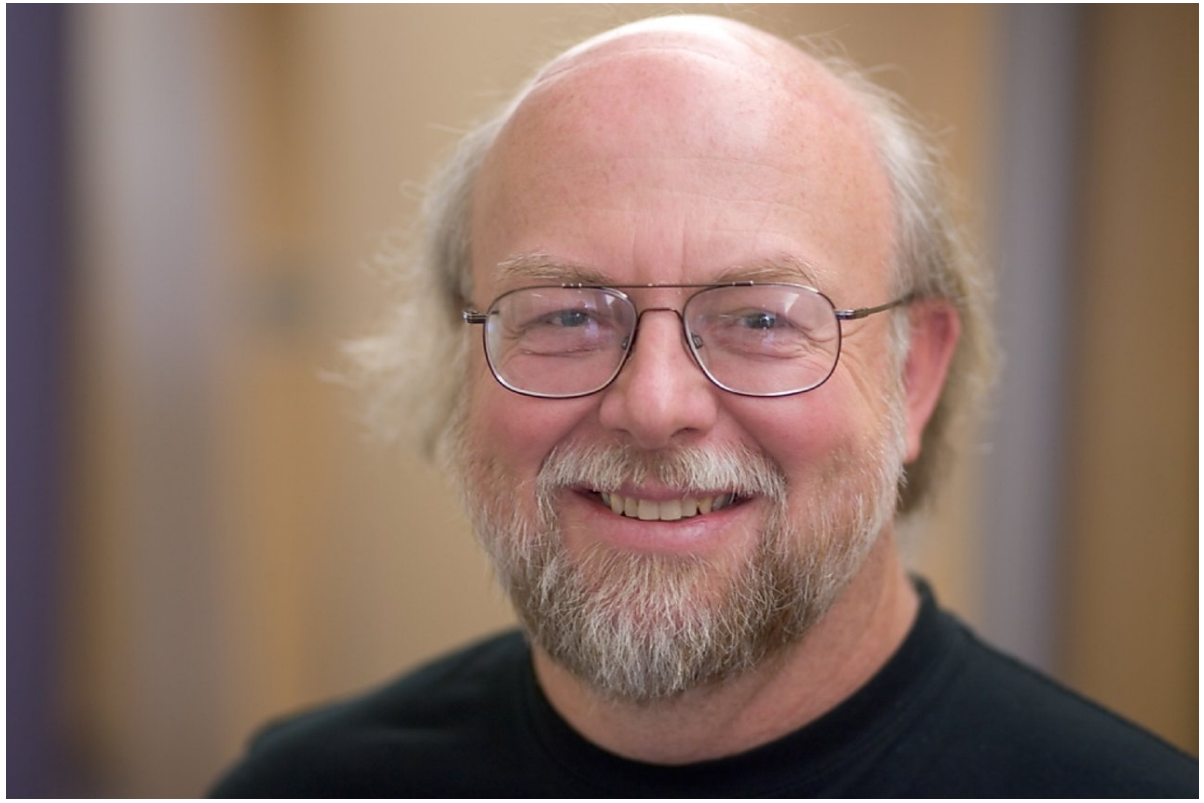


자바JAVA

- 1992년 미국의 SUN 사에서 가전제품들을 제어하기 위해 고안한 언어에서 부터 시작
- 1995년 5월에 SunWorld 95에서 공식 발표되었으며 C++를 기반으로 한 객체지향 프로그래밍 언어
- 월드와이드웹World Wide Web 이용에도 적합하도록 자바를 발전
- 범용적인 프로그래밍 언어 자바의 개발도구인 JDKJava Development Kit를 발표
- 선 마이크로시스템즈 사는 오라클Oracle 사에 합병되어 현재 자바는 오라클 기술
- 자바개발환경인 JDK는 현재까지 계속 발표

자바개발 배경

- 1990년 양방향 TV를 만드는 제어 박스의 개발을 위한 그린 프로젝트Green Project를 시작
  - 초기에는 객체지향 언어로 광범위하게 이용되고 있는 C++ 언어를 이용
  - 다양한 하드웨어를 지원하는 분산 네트워크 시스템 개발에 부족함을 느낀 개발팀은 C++ 언어를 기반으로 오크(Oak, 떡갈나무)라는 언어를 직접 개발
- 제임스 고슬링James Gosling은 이 오크라는 언어를 발전시켜 자바라는 범용적인 프로그래밍 언어를 개발



스크래치Scratch

- 2007년 MIT 대학의 미디어랩Media Lab에서 개발한 비주얼 프로그래밍visual programming 개발 도구
- 브라우저에서 직접 개발하는 환경으로 커뮤니티 기반 웹 인터페이스로 구성
- 일반인과 청소년 또는 프로그래밍 초보자를 학생들을 대상으로 컴퓨터 프로그래밍의 개념을 이해할 수 있도록 도와주는 교육용 프로그래밍 언어educational programming language
- 다양한 이미지나 사운드를 제공하여 쉽게 사용 가능
- 직관적으로 누구나 쉽게 이해할 수 있는 블록을 끼워 맞춰 프로그램을 작성

앨리스Alice

- 카네기 멜론 대학교에서 '마지막 강의'로 알려진 랜디 포시Randy Pausch 교수가 주도하여 개발한 프로그래밍 교육 도구
- 쉽게 삼차원 인터랙티브 그래픽 콘텐츠를 제작하면서 객체지향 프로그래밍 기법을 학습할 수 있는 교육용 소프트웨어
- 앨리스는 컴퓨터 전공학과의 저학년 학생들이 프로그래밍을 어렵게 생각하고 포기하지 않도록 프로그래밍 개념과 실습을 학습할 수 있도록 지원하는 프로그래밍 소프트웨어
- 특히 프로그래밍 언어 자바, C++, C#과 같은 객체지향 프로그래밍 기법을 학습하는데 유용한 개발 도구
- 다양한 캐릭터나 객체를 끌어다 놓는Drag & Drop 방식으로 프로그래밍
- 비주얼 프로그래밍 기법은 프로그래밍언어 문법학습에 대한 학습자들의 부담을 덜어 주고 쉽게 프로그래밍이 가능
- 클래스와 객체로 콘텐츠 제작에 사용할 수 있는 다양한 캐릭터를 제공하고, 사용법이 쉬우므로 일반인도 쉽게 삼차원 애니메이션이나 게임을 제작 가능

I
PROGRAMMING