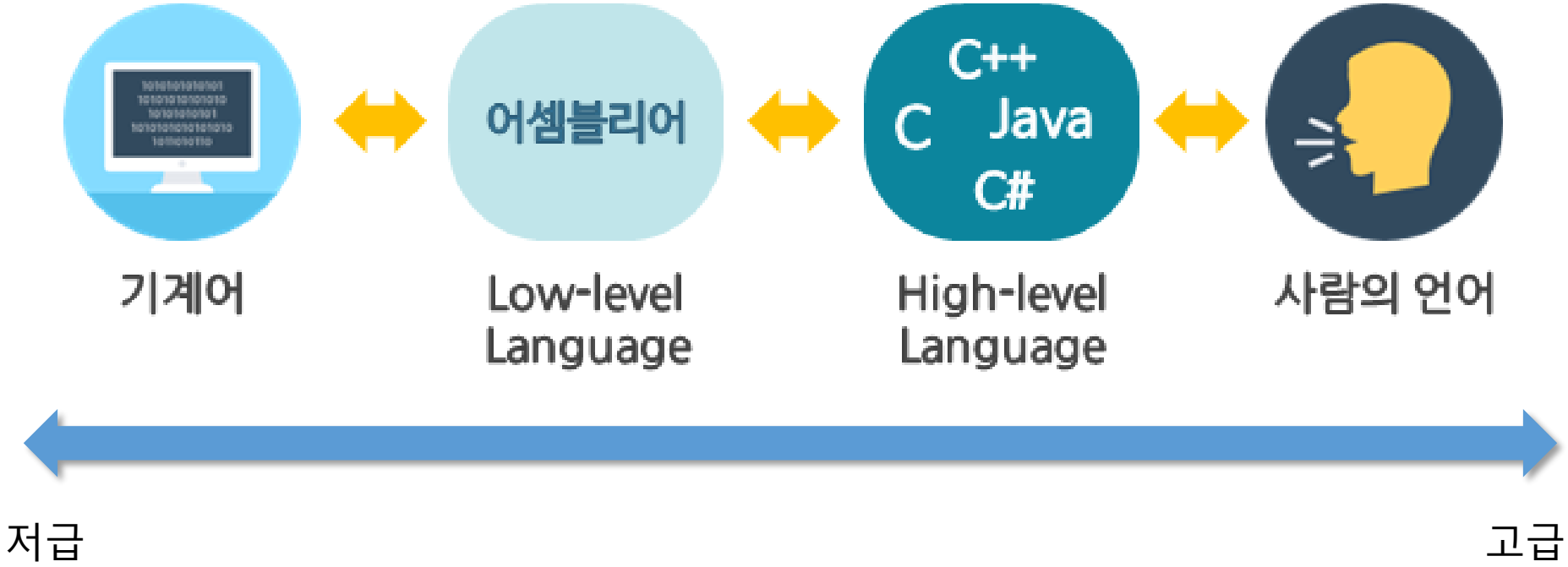
기계어와 어셈블리어

박시원

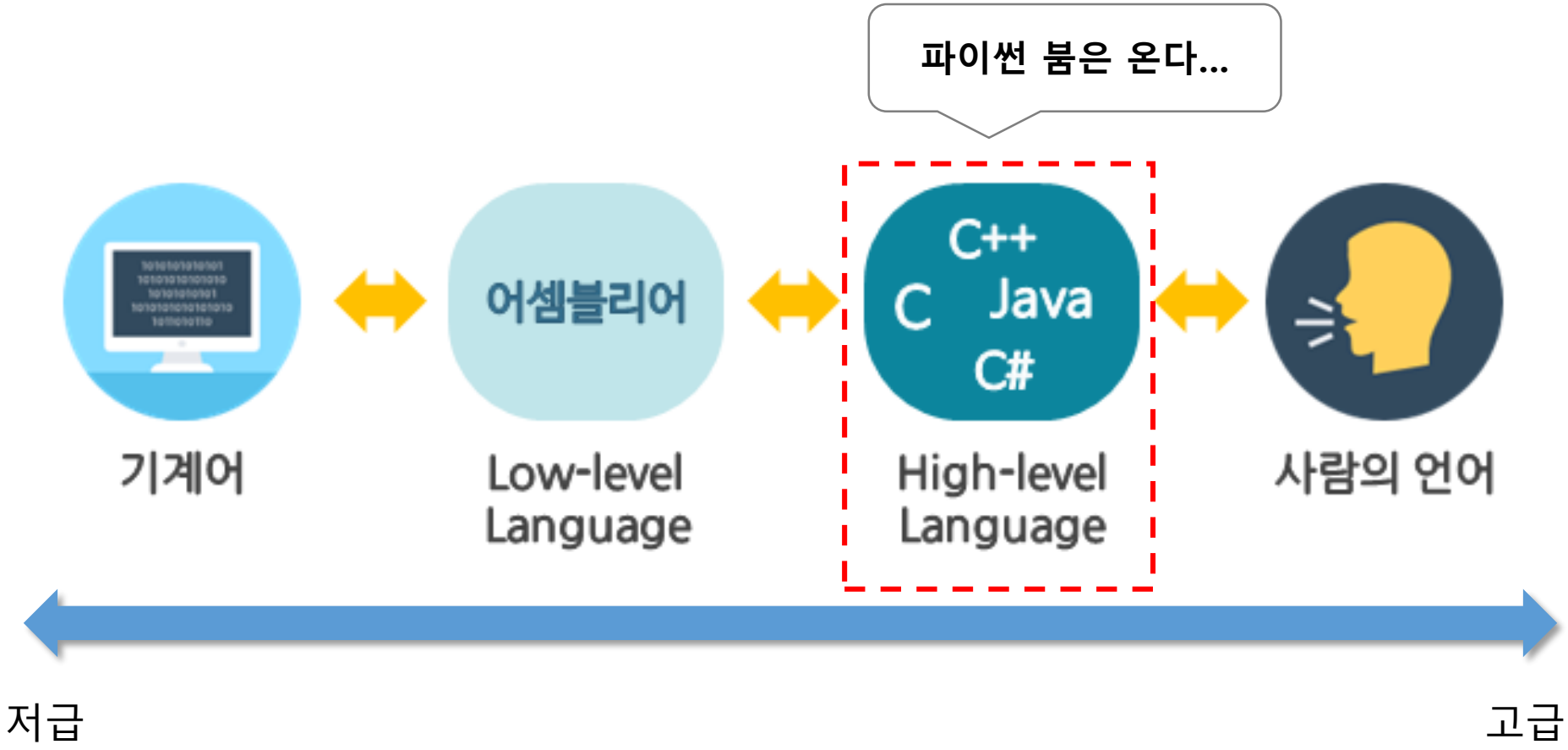
언어(Language)

컴퓨터 과학에서 언어는 크게 저급(Low-Level) 언어와 고급(High-Level) 언어로 나눌 수 있음
저급-고급이라고 해서 언어의 수준을 의미하는 것이 아니라, 얼마나 더 기계에 가까운 수준으로 레벨을 내렸느냐에 의미를 둠



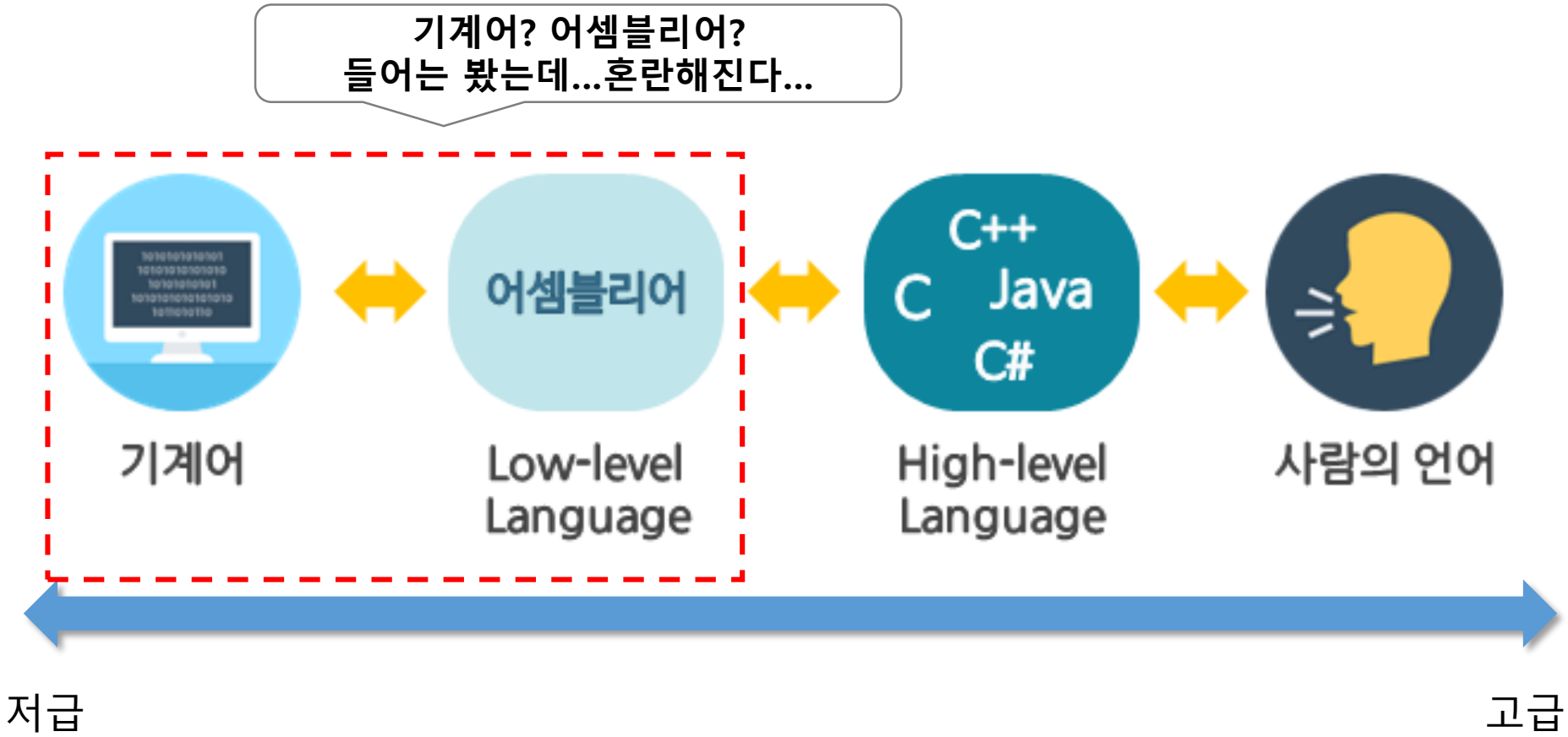
언어(Language)

우리가 흔히 잘 아는 Python, Java, C와 같은 프로그래밍 언어들을 컴퓨터 과학 세계에서는 고급 언어로 분류됨. 고급 언어로 갈수록 인간과 상호작용하기 쉬운 반면, 내려갈수록 인간이 이해하기 난해해짐



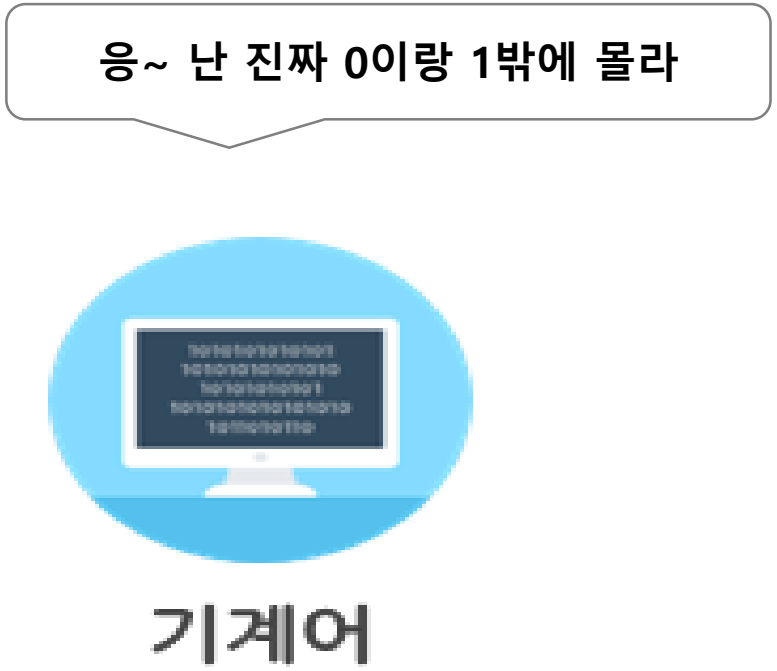
언어(Language)

Low-Level 언어는 기계어, 어셈블리어 단 둘 뿐이며, 인간이 이해하기 난해하여 배우기가 쉽지 않음
즉, 기계어로 사람이 프로그래밍하는 것은 좋은 선택이 아님



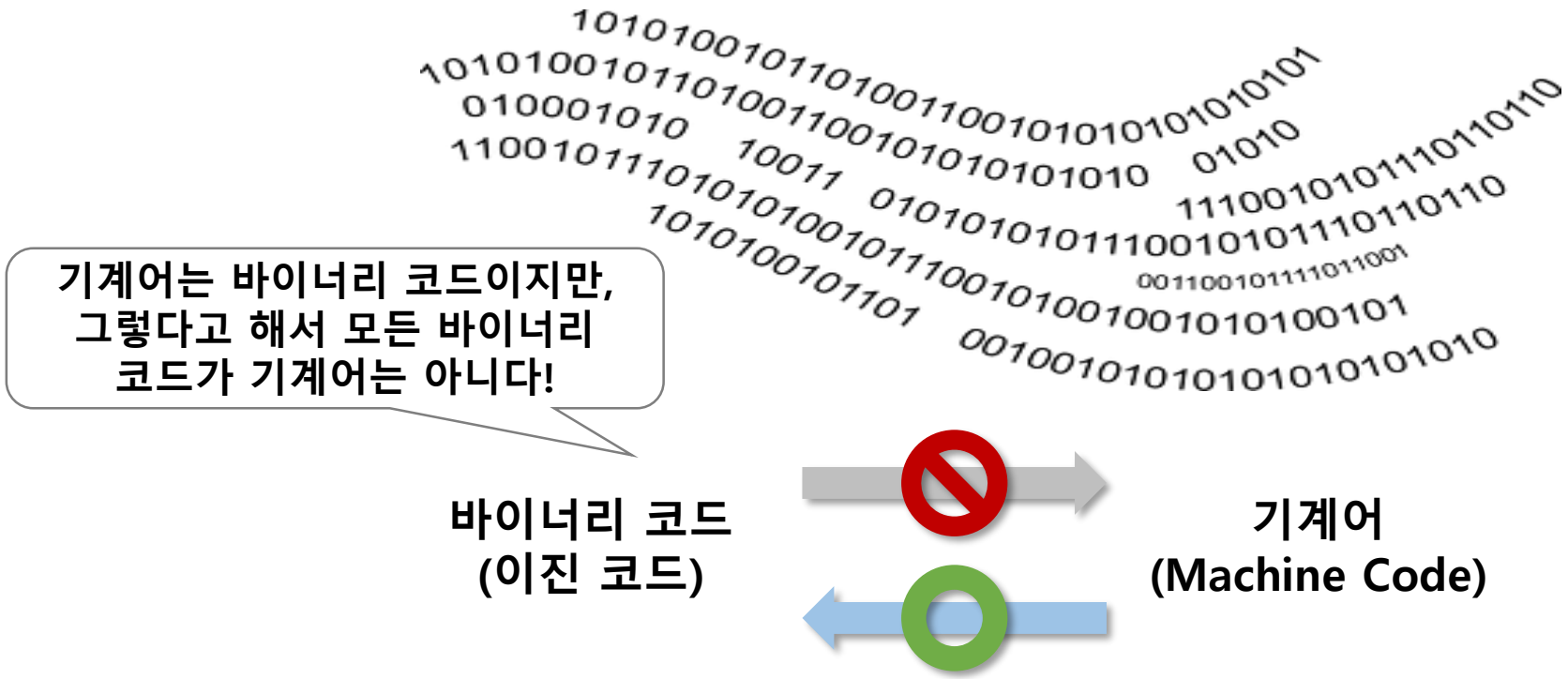
기계어(Machine Code)

CPU를 직접 제어하는 언어, CPU가 별다른 해석 없이(컴파일 없이) 읽을 수 있는 언어
우리가 흔히 컴퓨터는 0과 1밖에 모른다고 알고 있는데, 이 0과 1이 바로 기계어라고 할 수 있음
0과 1의 2개의 조합으로 이루어진 해당 언어 체계를 **바이너리(Binary) 코드(이진 코드)**라고 함



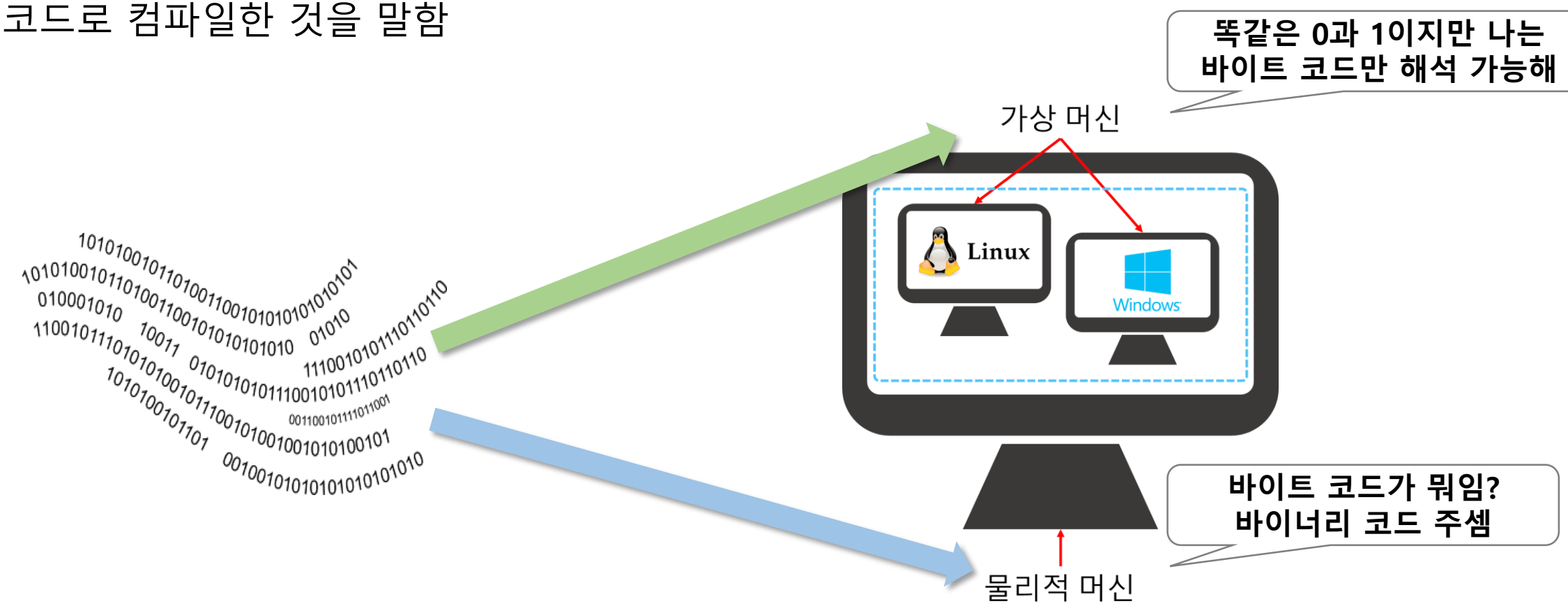
기계어(Machine Code)

여기서 주의할 점!
0과 1로 표현된 이진 코드이지만, 엄밀히 말해서 이 개념이 **숫자(Number) 데이터가 아님**
반대 급부의 개념으로 없음/있음, OFF/ON, 반응없음/있음 등과 같이 False/True의 개념이고,
인간이 해당 개념을 받아들이기 위해 직관적으로 숫자 0과 1로 표현한 것 뿐임
따라서 진짜 숫자 0과 1이 아니다

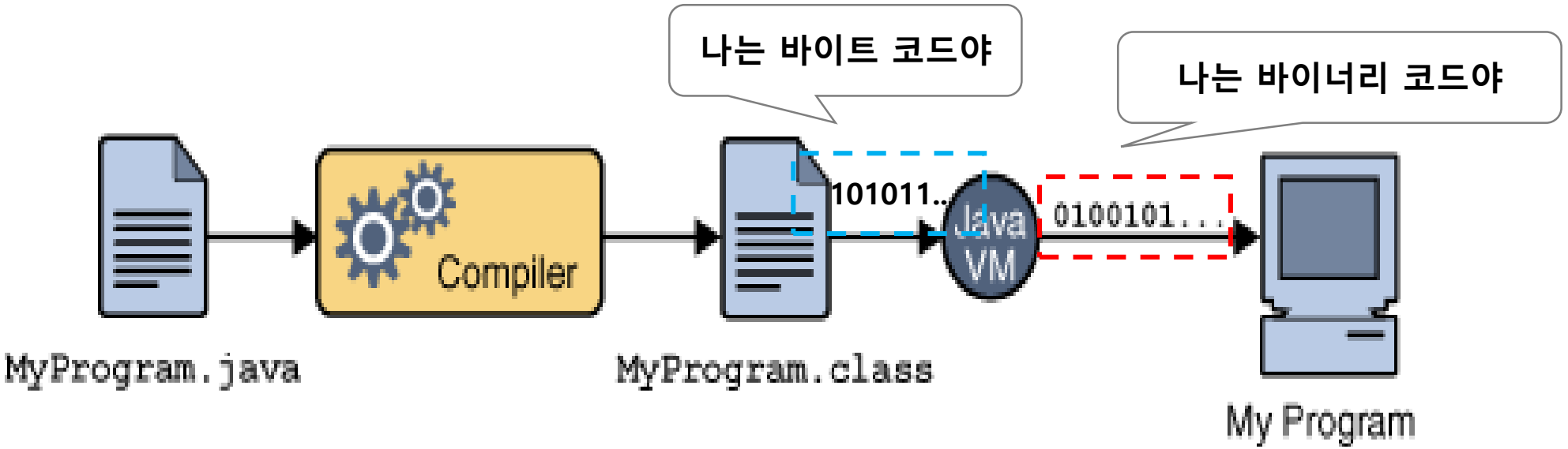


바이너리 코드? 바이트 코드?

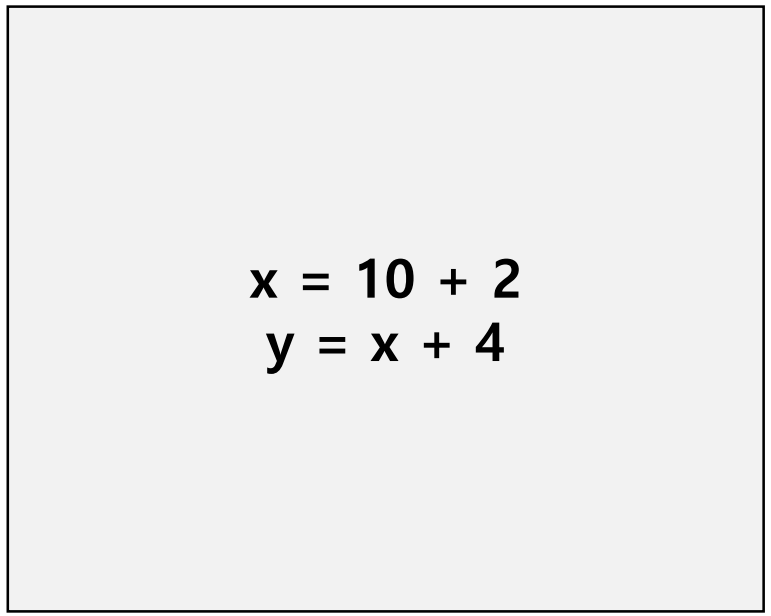
바이너리 코드는 뭐고, 바이트 코드는 또 뭐야?
CPU가 이해할 수 있는 언어인 기계어가 바이너리 코드라면, 바이트 코드는 VM(Virtual Machine, 가상 머신)이 이해할 수 있는 이진 코드 표현법
어떤 플랫폼에도 종속되지 않고 실행될 수 있는 가상 머신용 기계어 코드로 가상 머신이 이해할 수 있는 중간 코드로 컴파일한 것을 말함



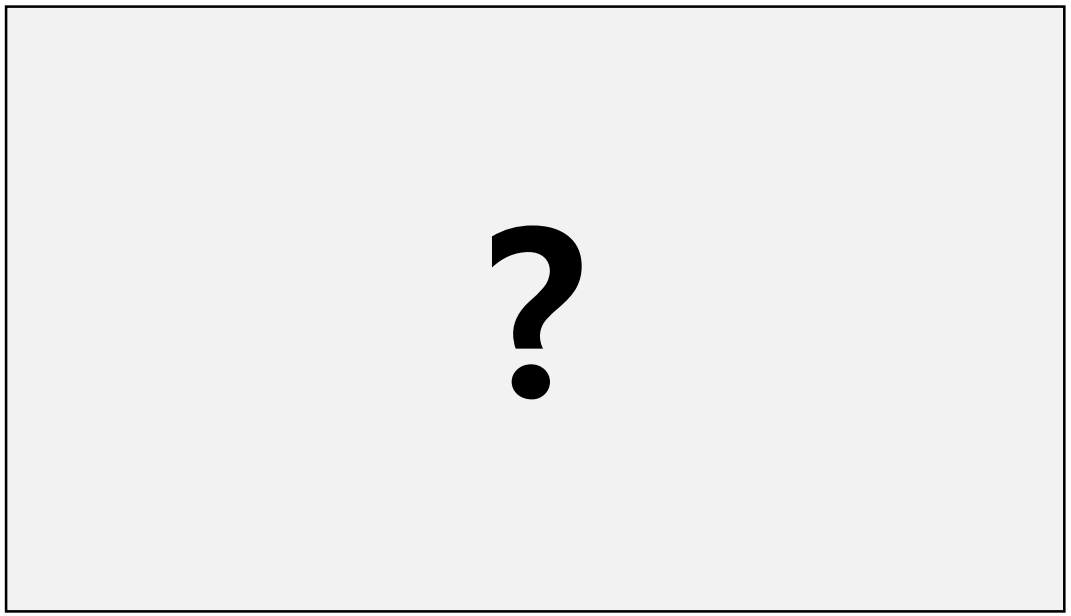
바이너리 코드? 바이트 코드?



기계어의 예시



수식으로 표현된
인간의 언어



기계어로 나타내면?

기계어의 예시

```
x = 10 + 2
y = x + 4
```

수식으로 표현된
인간의 언어

```
001001 11101 11101 1111111111111000
001000 00001 00000 0000000000001010
001000 00001 00001 0000000000000010
101011 11101 00001 0000000000000000
001000 00010 00001 0000000000000100
101011 11101 00010 0000000000000100
001001 11101 11101 0000000000001000
```

0과 1로 표현된 기계어

기계어의 예시

```
x = 10 + 2
y = x + 4
```

수식으로 표현된
인간의 언어

```
27 BD FF F8
20 20 00 0A
20 21 00 02
AF A1 00 00
20 41 00 04
AF A2 00 04
27 BD 00 08
```

0과 1로 표현된 기계어

기계어를 배우기 힘든 이유

기계어는 특정 언어가 아님
바이너리 코드(이진 코드)라는 언어의 체계는 갖춰져 있지만, 문법적으로 정해진 틀을 갖고 있는 언어가 아님
정확히 말해서, **CPU 제조사에서 CPU를 만들 때, 사용하는 명령어의 집합이 기계어**임
따라서 같은 회사의 CPU라도 경우에 따라 전혀 다른 기계어 명령을 가질 수 있으며,
마찬가지로 **같은 동작을 하는 명령어이지만 전혀 다른 0과 1의 나열로 표현될 수도 있음**!
물론 아주 기본적인 연산들은 호환되는 편임

10101 == 10001 ??
10101 != 10001 !!

가 A 0 1

한글 영어 기계어

어셈블리어(Assembly language)

기계어는 0과 1로 이루어졌기 때문에 간단한 프로그램 하나를 실행하더라도 코드가 길어질 뿐더러 사람이 읽기에도 큰 무리가 있음
이를 보완하기 위해 나온 언어가 어셈블리어임
어셈블리어는 어셈블러를 통해 기계어로 변환되며, 기계어와 1 대 1 대응됨
어셈블리어는 기계어와 고급 언어의 중간에 있다고 해서 중간 언어라고도 함

어셈블리어(Assembly language)

기계어와 1 대 1 대응되므로 어셈블리어 역시 기계어에 따라서 조금씩 달라지기도 함
대신 좀 더 언어 답게 문법, 데이터 체계도 갖추어져 있음

[그림 2-11] 어셈블리어의 데이터 타입

명령어 수행 방식

Opcode	Operand1	Operand2
ADD	EAX	EBX

```
mov     eax, DWORD PTR [ebp-8]
add     esp, 12
add     eax, DWORD PTR [ebp-4]
```

어셈블리어(Assembly language) vs 고급 언어

고급 언어	어셈블리어
▪ 프로세서 종류에 상관 없이 실행 가능하다 프로세서에 대한 사전 지식이 필요 없다 ▪ 프로그램 코드 양이 적고 디버깅이 용이하다 ▪ 컴파일러의 성능에 따라 프로그램의 성능이 좋다	▪ 동일한 종류의 프로세서만 실행할 수 있다 ▪ 프로세서에 대한 사전 지식이 필요하다 ▪ 메모리, I/O장치, 레지스터 등의 구성요소들을 직접 다룰 수 있다 ▪ 컴퓨터에서 실행하는 과정 등을 이해하기 쉽다 ▪ 컴퓨터의 성능에 최적인 프로그램을 작성할 수 있다