**1. 저수준 언어와 고수준 언어**

* **저수준 언어(Low-Level)**
* 컴퓨터가 실질적으로 이해하는 언어
* 2진수(바이너리)로 이루어져 있음
* 이를 기계어(machine language)라고 함
* 기계어는 컴퓨터가 바로 이해하므로 별도의 변환 과정이 필요 없음
* **고수준 언어(High-Level)**
* 사람이 이해할 수 있는 언어
* 인간이 사용하는 언어로 이루어져 있음
* 컴퓨터가 이해하는 ‘기계어’로 변환하는 과정이 필요함
* **고수준 저수준은 상대적인 것**
* 컴퓨터가 이해하기 쉬운 언어에 가까울수록 ‘저수준’
* 인간이 이해하기에 쉬운 언어에 가까울수록 ‘고수준’
* **기계어(Machine Language)**
* 네이티브 코드(native code)라고도 함
* 2진수로 이루어져 있음
* 가장 낮은 수준(저수준)의 언어
* 모든 프로그래밍 언어는 기계어로 번역됨
* **어셈블리어(Assembly Language)**
* 기계어보다 사람이 이해하기 그나마 쉬운 언어
* 엄밀하게는 고수준 언어에 속함
* 기계어와 일대일 대응이 가능함 – 이 때문에 보통 저수준 언어로 취급됨
* **고수준 프로그래밍 언어(High-Level Programming Language)**
* 사람이 사용하는 언어(영어)로 구성되어 있음
* 그러나 기계어와 일대일 대응이 불가능함
* 메모리를 누가 관리하는가에 따라서 ‘매니지드’와 ‘언매니지드’ 언어로 나뉨

**2. 컴파일언어와 인터프리터 언어**

* **컴파일 언어(Compiled Language)**
* 소스 코드(고수준 언어 코드) → 컴파일러(complier) → 기계어 → 실행
* ‘컴파일’ 이라는 것은 소스 코드를 기계어로 만들어주는 행동을 의미
* 오프라인(offline) 컴파일러가 고수준 언어를 네이티브 코드로 컴파일
* 컴파일 언어의 동작 방식 - 실행 파일(exe 확장자)은 기계어 상태로 있음, 실행되면 기계어를 읽어서 실행
* 다양한 최적화가 가능  
  - 컴파일 1시간 – 컴파일러가 소스코드를 기계어로 컴파일 할 때 자동으로 최적화(단, 시간이 소요됨)  
  - 실행 0.1초 – 최적화된 실행 속도를 획득
* 플랫폼마다 다른 exe 파일을 컴파일해야 함  
  - CPU마다 이해할 수 있는 기계어가 다를 수 있음 → CPU(플랫폼) 각각의 exe 파일을 컴파일
* **JIT 언어(Just-In-Time Language)**
* 소스 코드(고수준 언어 코드) → 컴파일러 → 중간 언어 코드(Intermediate Language Code)  
  → (JIT 컴파일러 → 기계어) → 실행
* 오프라인 컴파일러가 기계어가 아닌 ‘중간 언어(IL)’로 컴파일하여 실행 파일을 생성
* 플랫폼 별로 가상머신을 설치해야 함
* 가상머신(JIT 컴파일러)가 중간언어(IL)–실행파일을 기계어로 바꾸어 실행함  
  - 즉, 실행 파일은 중간 언어(IL) 상태로 존재하게 됨
* 실행 파일은 곧바로 하드웨어에서 동작하는 것이 아니라  
  - 프로그램 사용 시, 중간 단계인 가상머신을 통해 기계어로 컴파일 된 후 실행됨  
  - Just-In-time: 사용될 때 컴파일 되어 실행된다는 의미
* JIT 언어들은 플랫폼(하드웨어)에 종속적이지 않고 VM에 종속적임
* **인터프리터 언어(Interpreter Language)**
* 소스 코드(고수준 언어 코드) → (인터프리터 → 기계어) → 실행
* 소스 코드 그대로를 실행할 수 있음
* 중간 단계가 없음  
  - 컴파일이나 가상머신 등의 단계를 거치지 않고 실행됨
* 인터프리터가 소스 코드를 기계어(혹은 기계어로 된 함수, 명령어)로 대응시켜 프로그램 실행
* 실행 중에 고수준 언어를 네이티브 코드로 바꿔서 실행함
* 실제로 실행하기 전까지 코드의 문제를 잡아낼 수 없음  
  - 오프라인 컴파일러가 없어서 불가능