**자바 언어의 이해**

**JAVA**

자바는 썬에서 개발한 **객체지향 언어**이다. | 뒤에서 자세하게 설명

자바는 ‘오크(OAK)’ 언어에서 비롯되었으며 친화적인 개발 언어로 방향을 전환하여 ‘자바’가 되었다

**자바 가상머신(JVM, Java Vurtical Machine)**

자바는 **‘바이트코드’**를 생성한다.

ㄴ 바이트코드(Bytecode) : 하드웨어에 중속적이지 않은 중간 파일로, 가상머신을 거쳐 해석 , 실행하는 구조이다.

가상머신으로 자바는 스마트폰, 컴퓨터, Mac, 리눅스 등 **서로 다른 기기나 운영체제에서 동일한 프로그램을 개발하고 수행**할 수 있다.

**자바 플랫폼**

**자바 플랫폼 :** 자바 프로그램이 실행되는 특정 환경

대표적인 자바 플랫폼 기술 : Java SE(Standard Edition), Java ME(Micro Edition), Java EE(Enterprise Edition)

**JDK(Java Development Kit): 자바 프로그램 개발에 필요한 구성 요소**

**JRE(Java Runtime Environment): 자바 프로그램 실행에 필요한 구성 요소**

**API(Application Programming Interface): 자바 응용 프로그램 개발에 사용하는 라이브러리**

**자바의 장점**

간결하면서도 강력한 객체지향 언어이다.

개방형 구조이다. (여러 기술과 융합이 가능하다)

JVM 때문에 운영체제나 하드웨어 동일하게 실행

**자바의 단점**

절차 지향 언어에 비해서 속도가 느리다

**자바 실행 과정**

자바 소스코드(.java) -> 컴파일러 -> 바이트코드(.class) -> JVM -> 운영체제 -> 실행

**배열**

배열: 인덱스와 인덱스로 참조 가능한 데이터로 된 인종의 자료구조

집합형 데이터 관리가 필요할 때 사용하며 원시 자료형 뿐만 아니라 객체 타입도 지원한다.

데이터가 할당되지 않는 인덱스 위치를 참조할 때 Null Pointer Exception 발생

‘

**NullPointerException**

참조 변수가 null 값을 가지고 있을 때, 객체의 필드나 메소드를 사용하려고 했을 때 발생

**객체지향과 자바**

**객체지향 프로그래밍 언어**

객체지향 : 현실 세계의 개체 모델을 바탕으로 프로그램을 구조화하고 개발하는 프로그래밍 기법

객체 : 물리적으로 존재하거나 추상적으로 생각할 수 있는 것 중에서 자신의 속성을 가지고 있고 다른것과 식별 가능한 것

클래스 : 속성이 같은 객체들을 대표할 수 있는 대상(필드와 메소드로 구성)

인스턴스 : 클래스에 의해서 만들어진 객체

메소드: 클래스 내에 구현된 함수

필드 : 클래스에 내에서 선언된 변수

**상속**

상속은 어떤 클래스에서 좀 더 내용이 구체적인 새로운 클래스가 필요할 떄 기존 클래스를 물려받아 새로운 부분만 추가하거나 수정하려고 만든 개념이다.

상속을 이용하면 슈퍼 클래스의 기본 구성 요소(필드, 메서드)를 물려받으며(상속), 자신만의 필드나 메서드를 추가하여 구체화하는 것이 가능하다

상속은 클래스를 선언할 때 extends 키워드를 사용하여 정의한다.

**객체지향 프로그래밍의 주요 특징**

캡슐화 : 생성한 객체를 어떤 메서드와 필드로 어떻게 일을 수행할지 외부에 숨기는 특성을 말한다.

상속 : 상속은 어떤 클래스에서 좀 더 내용이 구체적인 새로운 클래스가 필요할 떄 기존 클래스를 물려받아 새로운 부분만 추가하거나 수정하려고 만든 개념

다형성 : 클래스의 상속 관계를 이용하여 슈퍼 클래스가 같은 서버 클래스들이 동일한 요청을 다르게 처리할 수 있는 특징을 말한다. 실제 프로그램에서는 메서드를 오버라이딩하여 구현한다.