※ 클래스: 시스템을 구성하는 구성요소들을 클래스로 만든다.

어떤 클래스를 구축하느냐에 따라 용도가 달라진다.

미니 프로젝트 주제

※ 각 클래스에 필요한 데이터

ex) 은행 시스템: 고객, 상품, 계좌, DB연동 클래스 등

고객: 성명, 주민번호, 주소, 전화, 대출, 자산

계좌:

교육 시스템: 고객, 강사, 강의, 운영자 등의 클래스 필요

쇼핑몰 시스템: 상품, 고객, 결제, 성별, 나이, 결혼유무

※ 클래스 기능

ex) 계좌: 계좌이체하기, 내역조회하기, 입/출금하기

각 클래스가 하나의 기능만 하도록 만들어주어야 한다.

자바는 유니코드 방식이라 한글로 변수이름을 써도 되지만 모든 시스템이 다 영어로 구성되어 있어서 영어로 구성해야 한다.

※ 객체지향언어

1. 코드의 재사용 성이 높다.

새로운 코드를 작성할 때 기존의 코드를 이용하여 쉽게 작성할 수 있다.

2. 코드의 관리가 용이하다.

코드간의 관계를 이용해서 적은 노력으로 쉽게 코드를 변경할 수 있다.

3. 신뢰성이 높은 프로그래밍을 가능하게 한다.

제어자와 메서드를 이용해서 데이터를 보호하고 올바른 값을 유지하도록 하며, 코드의 중복을 제거하여 코드의 불일치로 인한 오 동작을 방지할 수 있다.

• 클래스의 정의: 클래스란 객체를 정의해 놓은 것이다.

• 클래스의 용도: 클래스는 객체를 생성하는데 사용된다.

• 객체의 정의: 실제로 존재하는 것, 사물 또는 개념; 어떤 시스템을 구성하는 구성 요소

• 객체의 용도: 객체가 가지고 있는 기능과 속성에 따라 다름

• 유형의 객체: 책상, 의자, 자동차, TV와 같은 사물

• 무형의 객체: 수학공식, 프로그램 에러와 같은 논리나 개념

그 시스템이 컴퓨터 안에서 돌아갈 수 있도록 하려면 무엇이 정의되어 있어야 되는지를 생각하고 요소를 구현해 놓은 것이 클래스이다.

※ 메서드: 특정 작업을 수행하는 일련의 문장들을 하나로 묶은 것이다.

\* 메서드 사용 이유:

1. 높은 재사용성(reusability)

2. 중복된 코드의 제거

3. 프로그램의 구조화

\* 메서드의 선언과 구현

int add (int x, inty){

int result = x + y;

return result; // 결과를 반환

}

int: 반환타입(출력)

add: 메서드 이름

(int x, inty): 매개변수선언(입력)

※ OOP 특성

캡슐화

상속성

다형성

※ 클래스 멤버변수 만든다.

**private** String name;

**private** **int** korean;

**private** **int** english;

**private** **int** math;

**private** **int** science;

※ set get 메소드 자동 생성하는 방법

멤버변수 밑에 커서 넣고 오른쪽 클릭 -> source -> Generate getters and setters 클릭 -> select all -> ok