정적 변수 ← <차이점의 '클래스 변수'와 동일한 것이다.>
=static variables(static fields)
=class variables
=클래스 변수
예제) static int a;

정적 메소드
=static methods
=class methods
=클래스 메소드
예제) static void move( ) {...}

정적 변수와 정적 메소드를 통틀어서 Static Members 라고 하기도 한다. 이렇게 정리하고 보니 우리가 흔히 쓰는 일반 변수와 일반 메소드도 여러가지 다른 용어로 많이 쓰이는데 말나온김에 비교할겸 적어보기로 하겠다.

일반 변수
=instance variables
=인스턴스 변수
=멤버 변수
=필드 변수
예제) int a;

일반 메소드
=instance methods
=인스턴스 메소드
=멤버 메소드
=필드 메소드
예제) void move( ) {...}

위의 단어들을 몰라서 애를 먹는걸 방지하고자 한번 적어봤다. 필자는 문맥의 통일성을 위해 분류별로 가장 상위에 있는 용어(정적변수,정적메소드,일반변수,일반메소드)만 쓰기로 하겠다. 아무래도 서로 계속 비교를 해가면서 설명을 해야될듯한 묘한 예감(?)이 들기에 오늘 강좌는 용어만 통일해서 설명해도 여러분이 강좌를 이해하는데 많은 시간과 에너지를 절약하게 되리라 본다.^^

정적변수와 정적메소드라고 하니 아주 거창한걸 기대(?)했을지 모르겠지만 정적변수와 정적메소드라 함은 위와 같이 일반변수와 일반메소드 앞에 static 키워드를 써놓은 것을 말한다. 정적변수의 경우 객체를 아무리 많이 만들어도 해당변수는 하나만 존재한다. 일반 변수의 경우 객체가 생성될때마다 새롭게 생성되어 쓰이는 정적변수는 이와는 다르게 처음에 한번만 생성되고 공유해서 사용한다. static이란 클래스가 로딩될때 메모리 공간을 할당하는데 이 결정된 공간이 변하지 않으므로 정적이라는 표현을 쓴다. 메모리에 로딩 즉 메모리 공간을 확보해야 해당멤버를 쓸수가 있는데 자바는 프로그램을 실행하면 클래스가 우선 메모리에 로딩되나 static은 이보다 먼저 메모리에 로딩되어진다. 일반변수는 객체가 생성될때마다 메모리 공간이 할당되나 static의 경우 클래스가 메모리에 로딩되기전이미 정적변수와 정적메소드를 위한 메모리 공간이 할당되므로 객체가 생성될때마다 메모리 공간이 할당되지 않는다.

이런 까닭에 static에 대한 장점을 크게 두가지로 나눌수 있다. 첫째로 static을 쓴 변수나 메소드는 객체생성없이 모든 객체가 아무런 제약없이 공유할수 있다. 물론 객체생성하고 써도 상관없다. 둘째로, static을 쓴 변수는 값을 공유하므로 연속적으로 그 값의 흐름을 이어갈수 있다는 것이다. 다시 말해서 일반변수는 객체생성시마다 그 값이 초기화되지만 정적변수는 객체생성을 하지 않으므로 다른 객체에서 계속적으로 이어서 그 값을 변화시킬수 있는 것이다. 이에 대한 부분은 예제를 보면 쉬울 것이다. 앞전에 배운 final이란 키워드를 함께 사용하면(static final double PI = 3.141592653589793;) 공유는 하면서 값은 고정시킬수 있는데 보통 상수가 이에 해당한다. 말나온김에 잠깐 상수에 대해서 언급하고 계속 진행하겠다. 자바에서 상수(Constant Value)를 쓸때 보통 아래처럼 static과 final을 함께 사용한다. 상수는 항상 변하지 않는 고유한 속성을 지닌 멤버니까 말이다.

(접근지정자) static final 자료형 상수명=상수값;
public static final double PI = 3.141592653589793;
static final double PI = 3.141592653589793;

핵심!!!

static 변수는 해당 클래스에서 생성된 객체들이 공유하고 있음.

상수의 경우 편리성을 위해 예제(PI)처럼 이름을 소문자가 아닌 대문자로 표기하는데 한단어가 아닌 여러단어로 이루어질 경우 전부 대문자이면 분간이 어려우므로 단어 사이마다 _(underscore)로 연결시키는게 관례다.

다시 돌아와서 보충설명을 이어서 해보도록 하겠다. 정적변수나 정적메소드를 쓰는 예제는 아래와 같다.

클래스명으로 바로 접근가능!!

```
클래스명.a=100;  //객체생성없이 클래스명으로 정적변수에 접근가능
객체명.a=100;  //물론 객체생성후 얻은 인자로도 정적변수에 접근가능

클래스명.move( );  //객체생성없이 클래스명으로 정적메소드에 접근가능
객체명.move( );  //물론 객체생성후 얻은 인자로도 정적메소드에 접근가능
```

⇒ 객체 생성 없이 '클래스 변수'와 '클래스 메서드' 사용 가능!!

정적메소드나 정적메소드를 호출하는 방법은 위와 같으며 같은 클래스내에서는 클래스명을 생략하고 써도 무방하다. 일반메소드 안에서는 객체생성없이 정적변수나 정적메소드를 호출할수 있으나 static을 쓰는 정적메소드 안에서는 객체생성없이 일반변수나 일반메소드를 호출할수가 없다. 이유는 이미 위에서 메모리에 관련해 설명한 것처럼 static멤버들이 먼저 로딩되므로 아직 만들어지지않은 객체의 변수나 메소드를 참조해서 쓸수는 없기 때문이다. 따라서 this라는 키워드도 정적메소드안에서는 사용이 불가능하며 메소드안에서 쓰이는 지역변수에도 사용할수 없다. 반대로 당연하지만 일반메소드안에서는 정적변수나 정적메소드를 쓸수가 있다. 이미 로딩된 static멤버들을 쓰지 못할 이유가 없기 때문이다.

↳ 정적 메소드는 객체 생성 없이 호출된다. 따라서 정적 메소드 내에서 객체를 생성한 후 사용가능한 인스턴스 변수를 호출할 수 없다.

우리가 저번 시간에 abstract 추상메소드에 대해서 공부한바 있다. 여기에는 static을 쓸수가 있을까? 당연히 불가능하다. static은 객체생성 없이도 호출이 가능한 키워드인데 내용이 없으면 안된다. 우리가 예전에 배운 초기화 블록(클래스안에 {...} 괄호만 있는 형태)을 기억하는가? static도 초기화 블록처럼 정적변수와 정적메소드를 초기화시킬수 있는 공간을 클래스안 어느 위치에서나 갯수에 구애받지않고 아래와 같이 만들수 있는데 이를 정적 초기화 블록(Static Initialization Blocks)이라고 부른다. 정적초기화블록은 클래스 로딩될때 자동으로 작동되며 초기화 블록에다가 아래처럼 static만 점가한 형태이다.

```
static
{
... //정적변수나 정적메소드만 쓸수 있다.
..... //초기화시키고 싶은 내용 아무거나 쓰면 된다.
System.out.println("static {...}은 쓰는데 갯수 제한없다");
}
```