

7. this와 super

- this : 자신의 객체를 참조한다.(각 클래스의 모든 객체는 묵시적으로 this를 가진다)
- super : 부모 객체를 참조한다.

◆ this와 super 사용법

	생성자를 호출할 때	메서드를 호출할 때	속성을 호출할 때
this	this(인수1, 인수2, ...)	this.메서드명()	this.속성명
super	super(인수1, 인수2, ...)	super.메서드명()	super.속성명

```

class A{
    String s = "pass25";
    public void foo1(){ System.out.println(this.s); }
    public void foo2(){ this.foo1(); }
}

class B extends A{
    public void goo1(){ super.foo2();}
    public void goo2(){ this.goo1(); }
}

class ThisSuper{
    public static void main(String args[]){
        B b = new B();
        b.goo2();
    }
}
    
```

[실행결과]

pass25

//탐구 - this

클래스에 있는 모든 속성은 같은 클래스에 있는 모든 메서드에서 사용할 수 있다.
속성과 메서드의 매개변수 이름이 같으면 어떻게 구별할 것인가?라는 문제가 발생된다.
자바에서는 키워드 this를 사용하여 이를 해결하도록 권고하고 있다.

```
public class Date
{
    private int year, month;        //속성(멤버변수)

    public void setDate(int year, int month)
    {
        this.year = year;          //속성과 매개변수 이름이 같으므로 this를 붙여 구별함
        this.month = month;
    }

    public static void main(String args[])
    {
        Date d = new Date();        //객체 d 생성
        d.setDate(2030, 12);
        System.out.println(d.year + "년 " + d.month + "월");
    }
}
```

[실행결과]

2030년 12월

- 키워드 this를 사용하면 명시적으로 속성과 매개변수를 구별할 수 있다.
- 이는 각 클래스에서 생성되는 모든 객체는 묵시적으로 this를 가지고 있기 때문이다.
- 즉, this를 사용하면 속성을 명시적으로 지정할 수 있게 된다.(this.year=year처럼)
- 만약, this를 사용하지 않으면 그냥 매개변수에 매개변수를 대입하는 것이 된다.(year=year)
- 다시 설명하면, this는 내부적으로 자신의 객체를 참조한다.
- 이는 this를 이용하여 클래스를 구성하고 있는 속성과 메서드에 명시적으로 접근할 수 있다.

// super()를 이용한 부모클래스 생성자 호출

메서드 super()는 자식클래스가 부모클래스의 생성자를 호출하기 위해 사용한다.
즉, 자식클래스 생성자가 실행되기 전에 부모클래스의 생성자가 실행되어야 함을 뜻한다.
super()는 자식 생성자에서 제일 먼저 기술(호출)되어야 한다.

```
class Car
{
    protected String carType;
    protected String color;
    protected int power;
    public Car(String carType, String color, int power){
        this.carType = carType;
        this.color = color;
        this.power = power;
    }
}
public class Rexten extends Car{
    private String owner;
    public Rexten(String owner){
        super("Rexten", "White", 320); //super()는 첫번째 줄에 기술되어야 한다.
        this.owner = owner;
    }
    public String toString(){
        return owner + " : " + carType + "." + color + "." + power;
    }
    public static void main(String args[]){
        Rexten my = new Rexten("한성");
        System.out.println(my.toString());
    }
}
```

[실행결과]

한성 : Rexten.White.320

4 <http://cafe.daum.net/pass365>(홍재연)

// super()는 생략 가능

먼저, 상속 관계에 있는 다음 두 개의 Java 프로그램 차이점을 잘 살펴보기 바란다.

㉠	<pre>class A { A(){ System.out.println("A"); } } class B extends A { B(){ System.out.println("B"); } //super()가 없다.(생략된 것) } public class C extends B { C(){ System.out.println("C"); } //super()가 없다.(생략된 것) public static void main(String args[]) { C object = new C(); //최하위클래스 객체 생성 } }</pre>
㉡	<pre>class A { A(){ System.out.println("A"); } } class B extends A { B(){ <u>super()</u>; System.out.println("B"); } //super()가 있다. } public class C extends B { C(){ <u>super()</u>; System.out.println("C"); } //super()가 있다. public static void main(String args[]) { C object = new C(); //최하위클래스 객체 생성 } }</pre>

- 먼저, 두 프로그램은 같은 것이다. 실행 결과는 같다. “A, B, C” 순으로 출력된다.
- 상속 관계에서 자식클래스의 객체가 생성될 때, 상위클래스 생성자 호출이 없으면 **기본 생성자**가 우선적으로 자동 호출(실행)된다. 해서, A() → B() → C() 순으로 호출된다.
- 자식클래스의 생성자에서 부모클래스의 생성자 호출이 없으면, **super();**가 생략된 것이다.
- 여기서 중요한 것은 메서드 super();는 원시코드에서 생략할 수 있다는 것이다.

☞ super();가 생략되어 있는 것을 모르면 프로그램 실행 결과를 묻는 문제에서 답은 틀릴 수밖에 없다. 평소에는 알고 있다가 시험 보면서 눈에 보이지 않으니깐! 그냥 풀게 된다.

기출문제 분석

1. Java 프로그램의 실행 결과로 옳은 것은? [2019년 계리]

<pre>public class B extends A{ int a = 20; public B() { System.out.print("다"); } public B(int x) { System.out.print("라"); } }</pre>	<pre>public class A { int a = 10; public A() { System.out.print("가"); } public A(int x) { System.out.print("나"); } public static void main(String[] a){ B b1 = new B(); A b2 = new B(1); System.out.print(b1.a + b2.a); } }</pre>
---	---

- ① 다라30 ② 다라40 ③ 가다가라30 ④ 가다가라40

☞ 자바 프로그램 - 상속

· 주어진 자바 프로그램은 다음처럼 기술할 수 있다.

```
class A {
    int a = 10;
    public A() { System.out.print("가"); }
    public A(int x) { System.out.print("나"); }
}
class B extends A {
    int a = 20;
    public B() { super(); System.out.print("다"); } //super()가 생략되어 있음
    public B(int x) { super(); System.out.print("라"); } //super()가 생략되어 있음
}
public class Test {
    public static void main(String[] a){
        B b1 = new B(); //객체 b1 생성, A()와 B()가 호출 - 출력 : 가다
        A b2 = new B(1); //객체 b2 생성, A()와 B(1)이 호출 - 출력 : 가라
        System.out.print(b1.a + b2.a); //b1.a + b2.a = 20 + 10 = 30 - 출력 : 30
    } //b2.a = 10인 이유 : 객체 b2가 부모타입의 참조형이므로
}
```

정답 : ③

2. 다음 Java 프로그램의 출력 값은? [2018년 컴일 국가 9급]

```
-----  
class Super {  
    Super() { System.out.print('A'); }  
    Super(char x) { System.out.print(x); }  
}  
class Sub extends Super {  
    Sub() { super(); System.out.print('B'); }  
    Sub(char x) { this(); System.out.print(x); }  
}  
public class Test {  
    public static void main(String[] args) {  
        Super s1 = new Super('C');  
        Super s2 = new Sub('D');  
    }  
}  
-----
```

- ① ABCD ② ACBD ③ CABD ④ CBAD

☞ Java 프로그램

- 먼저, 주어진 프로그램에서 Super()와 super()는 서로 다른 것이다.(당연히 알고 있는 것!)
- Super()는 사용자가 정의한 Super 클래스의 생성자이고
- super()는 상속 관계에서 부모클래스의 생성자를 호출하는 것이다.

```
class Super {  
    Super() { System.out.print('A'); }           //⑥ 문자 'A' 출력  
    Super(char x) { System.out.print(x); }      //② 전달받은 문자 'C' 출력  
}  
class Sub extends Super {  
    Sub() { super(); System.out.print('B'); }    //⑤ super();에 의해 부모 생성자 Super() 호출  
                                                //⑦ 문자 'B' 출력  
    Sub(char x) { this(); System.out.print(x); } //④ this();에 의해 자신 생성자 Sub() 호출  
                                                //⑧ 전달받은 문자 'D' 출력  
}  
public class Test {  
    public static void main(String[] args) {  
        Super s1 = new Super('C');             //① 생성자 Super(char x) 호출, 문자 'C' 전달  
        Super s2 = new Sub('D');              //③ 생성자 Sub(char x) 호출, 문자 'D' 전달  
    }  
} //출력 : CABD
```