19. 부모타입의 참조형이 자식타입의 객체 참조

[예제 1] 부모타입의 참조형(a)이 자식 객체를 참조

```
class A                                  //부모클래스 A
{
  int get(){ return 100; }
}
class B extends A                        //자식클래스 B
{
  int get(){ return 200; }               //메서드 재정의(overriding)
}
public class Test
{
  public static void main(String args[])
  {
    A a = new B();                       //부모타입의 참조형이 자식 객체를 참조
    System.out.println(a.get());         //출력 200 - 자식에서 재정의된 메서드 호출
  }
}
```

- 자식클래스에서 재정의된 것이 있으면, 재정의된 것이 호출된다. - a.get();
- 메서드 재정의는 "메서드 이름, 매개변수 개수 및 자료형"이 모두 같아야 한다.

[예제 2] 부모타입의 참조형(a)이 자식 객체를 참조

```
class A                                  //부모클래스 A
{
  int get(){ return 100; }
}
class B extends A                        //자식클래스 B
{
  int get(int k){ return 200; }          //메서드 재정의(overriding) 아님 - 인수 k가 있으므로
}
public class Test
{
  public static void main(String args[])
  {
    A a = new B();                       //부모타입의 참조형이 자식 객체를 참조
    System.out.println(a.get());         //출력 100 - 부모에서 정의된 메서드 호출(재정의가 아니므로)
  }
}
```

[예제 3] 부모타입의 참조형(a)이 자식 객체를 참조

```
class A                                  //부모클래스 A
{
  int x = 100;
  int get(){ return x; }
}
class B extends A                        //자식클래스 B
{
  int x = 200;
  int get(){ return x; }                 //메서드 재정의(overriding)
}
public class Test
{
  public static void main(String args[])
  {
    A a = new B();                       //부모타입의 참조형이 자식 객체를 참조
    System.out.println(a.x);             //출력 100 - 속성은 부모에서 정의된 것 호출
    System.out.println(a.get());         //출력 200 - 메서드는 자식에서 재정의된 호출
  }
}
```

◈ **부모타입의 참조형이 자식 객체를 참조하는 경우**

- 부모타입의 참조형(a)으로는 부모클래스에 정의된 것만 호출할 수 있다.
- 만약, 부모클래스에서 정의되지 않은 속성, 메서드를 호출하면 오류가 발생된다.
- 속성은 무조건 부모클래스에서 정의된 것이 호출된다. (a.x=100)
- 단, 일반메서드는 자식클래스에서 재정의된 것이 있으면, 재정의된 것이 호출된다.

A.x	B.x
100	200

a.x = 100, a.get() = 200

◈ **추가로 한마디 더하면(OOP의 특징)**

- 부모타입의 참조형(a)으로 자식클래스에 정의된 속성을 직접 참조하는 것은 불가능하지만
- 자식클래스에서 재정의된 일반메서드를 통해서 간접 참조는 가능하다.(int x = 200;)

[예제 4] 부모타입의 참조형(a)이 자식 객체를 참조

```
class A                          //부모클래스 A
{
  int x = 1;
  static int y = 2;
  int f(){ return 3; }           //일반메서드
  static int g(){ return 4; }    //정적메서드
}
class B extends A                //자식클래스 B
{
  int x = 5;
  static int y = 6;
  int f(){ return 7; }           //일반메서드
  static int g(){ return 8; }    //정적메서드
}
public class Test
{
  public static void main(String args[])
  {
    A a = new B();               //부모타입의 참조형 a가 자식 객체를 참조
    System.out.println(a.x);     //출력 1 : 일반 멤버 속성값 1 출력
    System.out.println(a.y);     //출력 2 : 정적 멤버 속성값 2 출력
    System.out.println(a.f());   //출력 7 : 자식클래스에서 재정의된 일반메서드를 호출하여 7 출력
    System.out.println(a.g());   //출력 4 : 부모클래스에 정의된 정적메서드를 호출하여 4 출력
  }
}
```

- 부모타입의 참조형(a)으로는 부모클래스에 정의된 것만 호출할 수 있다.
- 속성은 무조건 부모클래스에서 정의된 것이 호출된다.
- 정적메서드는 무조건 부모클래스에서 정의된 것이 호출된다.
- 단, 멤버 중 **일반메서드만**이 자식클래스에서 재정의된 것이 있으면, 재정의된 것이 호출된다.

- 아무튼, 위의 프로그램 실행 결과를 잘 정리해야 한다. 외부적으로 눈에 보이지 않는다.
- 자바 시험에서 계속 출제되고 있다.

부모타입의 참조형이 자식 객체를 참조

//아버지는 돈이 1억 있고, 장남은 10억이고, 차남은 100억이고, 막내는 1000억인 경우
//현실적인 예를 들어 본 것입니다.

```
class 아버지
{
  int 돈 = 1억;                                  //아버지는 돈이 1억 있고
  int get() { return 돈; }
}
class 장남 extends 아버지
{
  int 돈 = 10억;                                 //장남은 돈이 10억 있고
  int get() { return 돈 + this.돈 + super.돈; }   //10억 + 10억 + 1억 = 21억
}
class 차남 extends 아버지
{
  int 돈 = 100억;                                //차남은 돈이 100억 있고
  int get() { return 돈 + this.돈 + super.돈; }   //100억 + 100억 + 1억 = 201억
}
class 막내 extends 아버지
{
  int 돈 = 1000억;                               //막내는 돈이 1000억 있음
  int get() { return 돈 + this.돈 + super.돈; }   //1000억 + 1000억 + 1억 = 2001억
}
class Test
{
  public static void main(String args[])
  {
    아버지 p1 = new 장남();       //부모타입의 참조형이 자식 객체를 참조
    아버지 p2 = new 차남();       //부모타입의 참조형이 자식 객체를 참조
    아버지 p3 = new 막내();       //부모타입의 참조형이 자식 객체를 참조
    System.out.println(p1.get());
    System.out.println(p2.get());
    System.out.println(p3.get());
  }
}
```

기출문제 분석

1. 다음 자바코드를 실행할 때 출력 결과는? [2006년 국가 7급]

```
-----------------------------------------
class A
{
  static int f(){ return 1; }
  int g(){ return 2; }
}
class B extends A
{
  static int f(){ return 4; }
  int g(){ return 8; }
}
class Test
{
  public static void main(String args[])
  {
    A a = new B();
    System.out.println(a.f() + a.g());
  }
}
-----------------------------------------
```

① 3　　② 5
③ 9　　④ 12

♤ **부모타입의 참조형이 자식 객체를 참조**

- **정적메서드**는 무조건 부모클래스에서 정의된 것이 호출된다.
- **일반메서드**는 자식클래스에서 재정의된 것이 있으면, 재정의된 것이 호출된다.

```
static int f(){ return 1; }    //정적메서드
int g(){ return 8; }           //일반메서드
```

∴ a.f() + a.g() = 1 + 8 = **9**

정답 : ③

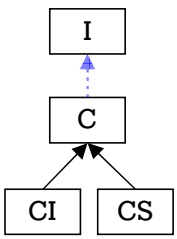
2. 다음에 주어진 Java 프로그램의 Example 클래스 내부의 내용 중에서 컴파일 오류를 일으키는 잘못된 코드는? [2022년 군무원 9급]

```
-----------------------------------
class C{}
class CS extends C{}
interface I{}
class CI extends C implements I{}
class Example
{
  static I i = new CI();
  static C ca = new CI();
  static CS cs = new C();
  static C cb = new CS();
}
-----------------------------------
```

① static I i = new CI();
② static C ca = new CI();
③ static CS cs = new C();
④ static C cb = new CS();

♤ **인스턴스(객체) 생성**

- 부모 = 자식 ← 정상적으로 객체 생성
- 자식 = 부모 ← 객체 생성 불가 (오류 발생)




```
class C{}
class CS extends C{}
interface I{}
class CI extends C implements I{}
class Example
{
  static I i = new CI();        // 부모 I = 부모 C = 자식 CI
  static C ca = new CI();       // 부모 C = 자식 CI
  static CS cs = new C();       // 자식 CS = 부모 C ← 객체 생성 불가 (오류 발생)
  static C cb = new CS();       // 부모 C= 자식 CS
}
```

정답 : ③

3. 다음 Java 프로그램의 출력 결과는? [2019년 국가 9급]

```
class ClassP
{
  int func1(int a, int b) { return (a + b); }
  int func2(int a, int b) { return (a - b); }
  int func3(int a, int b) { return (a * b); }
}
public class ClassA extends ClassP
{
  int func1(int a, int b) { return (a % b); }
  double func2(double a, double b) { return (a * b); }
  int func3(int a, int b) { return (a / b); }
  public static void main(String[] args) {
    ClassP P = new ClassA();
    System.out.print(P.func1(5, 2) + ", " + P.func2(5, 2) + ", " + P.func3(5, 2));
  }
}
```

① 1, 3, 2　　② 1, 3, 2.5
③ 1, 10.0, 2.5　　④ 7, 3, 10

♤ 부모타입의 참조형(p)이 자식 객체를 참조, 자식에서 재정의된 메서드가 호출

- int func1(int a, int b) { return (a + b); }
- int func2(int a, int b) { return (a - b); }
- int func3(int a, int b) { return (a * b); }

↓재정의 여부

- int func1(int a, int b) { return (a % b); }　　//재정의
- double func2(double a, double b) { return (a * b); }　　//재정의 아님
- int func3(int a, int b) { return (a / b); }　　//재정의

- P.func1(5, 2) = 5 % 2 = **1**
- P.func2(5, 2) = 5 − 2 = **3** → 재정의가 아니므로 부모클래스에 정의된 메서드 호출
- P.func3(5, 2) = 5 / 2 = **2** → 정수와 정수 연산 결과는 정수

정답 : ①

4. 〈보기〉의 Java 프로그램의 실행 결과는?

```
------<보기>----------------------------------------
class A {
  public void f() { System.out.print("1"); }
  public static void g() { System.out.print("2"); }
}
class B extends A { public void f() { System.out.print("3"); } }
class C extends B { public static void g() { System.out.print("4"); } }
public class D {
  public static void main(String args[]) {
    A obj = new C();
    obj.f();
    obj.g();
  }
}
-------------------------------------------------
```

① 3 2　　② 3 4

③ 1 2　　④ 1 4

♤ 부모타입의 참조형(obj)이 자식 객체를 참조

```
class A {
  public void f() { System.out.print("1"); }                    //일반메서드
  public static void g() { System.out.print("2"); }              //정적메서드
}
class B extends A { public void f() { System.out.print("3"); } }      //일반메서드
class C extends B { public static void g() { System.out.print("4"); } }  //정적메서드
public class D {
  public static void main(String args[])
  {
    A obj = new C();
    obj.f();    //출력 : 자식클래스에서 재정의된 일반메서드를 호출하여 3을 출력
    obj.g();    //출력 : 부모클래스에 정의된 정적메서드를 호출하여 2를 출력
  }
}
```

- 부모타입의 참조형(obj)으로는 부모클래스에 정의된 것만 호출할 수 있다.
- 속성은 무조건 부모클래스에서 정의된 것이 호출된다.
- 정적메서드는 무조건 부모클래스에서 정의된 것이 호출된다.
- 단, 멤버 중 **일반메서드만**이 자식클래스에서 재정의된 것이 있으면, 재정의된 것이 호출된다.

정답 : ①

5. Java 프로그램의 실행 결과로 옳은 것은? [2019년 우정 9급]

```
public class B extends A{
  int a = 20;
  public B()
  {
    System.out.print("다");
  }
  public B(int x)
  {
    System.out.print("라");
  }
}
```

```
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);
  }
}
```

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