Casting(업캐스팅 & 다운캐스팅)

캐스팅이란?

> 변수가 원하는 정보를 다 갖고 있는 것

```
int a = 0.1; // (1) 에러 발생 X
int b = (int) true; // (2) 에러 발생 O, boolean은 int로 캐스트 불가
```

(1)은 0.1이 double형이지만, int로 될 정보 또한 가지고 있음

(2)는 true는 int형이 될 정보를 가지고 있지 않음

캐스팅이 필요한 이유는?

1. **다형성** : 오버라이딩된 함수를 분리해서 활용할 수 있다.
2. **상속** : 캐스팅을 통해 범용적인 프로그래밍이 가능하다.

형변환의 종류

1. **묵시적 형변환** : 캐스팅이 자동으로 발생 (업캐스팅)

```
Parent p = new Child(); // (Parent) new Child()할 필요가 없음
```

> Parent를 상속받은 Child는 Parent의 속성을 포함하고 있기 때문

2. **명시적 형변환** : 캐스팅할 내용을 적어줘야 하는 경우 (다운캐스팅)

```
Parent p = new Child();
Child c = (Child) p;
```

> 다운캐스팅은 업캐스팅이 발생한 이후에 작용한다.

예시 문제

```
class Parent {
    int age;

    Parent() {}

    Parent(int age) {
        this.age = age;
    }

    void printInfo() {
        System.out.println("Parent Call!!!!");
    }
}

class Child extends Parent {
    String name;

    Child() {}

    Child(int age, String name) {
        super(age);
        this.name = name;
    }

    @Override
    void printInfo() {
        System.out.println("Child Call!!!!");
    }

}

public class test {
    public static void main(String[] args) {
        Parent p = new Child();

        p.printInfo(); // 문제1 : 출력 결과는?
        Child c = (Child) new Parent(); //문제2 : 에러 종류는?
    }
}
```

문제1 : `Child Call!!!!`

> 자바에서는 오버라이딩된 함수를 동적 바인딩하기 때문에, Parent에 담겼어도 Child의 printInfo() 함수를 불러오게 된다.

문제2 : `Runtime Error`

> 컴파일 과정에서는 데이터형의 일치만 따진다. 프로그래머가 따로 (Child)로 형변환을 해줬기 때문에 컴파일러는 문법이 맞다고 생각해서 넘어간다. 하지만 런타임 과정에서 Child 클래스에 Parent 클래스를 넣을 수 없다는 것을 알게 되고, 런타임 에러가 나오게 되는것!