제네릭 타입의 상속과 구현

제네릭 타입의 상속과 구현

❖ 제네릭 타입을 부모 클래스로 사용할 경우

- 타입 파라미터는 자식 클래스에도 기술해야 !!!

```
public class ChildProduct<T, M> extends Product<T, M> { ... }
```

- 추가적인 타입 파라미터 가질 수 있음

```
public class ChildProduct<T, M, C> extends Product<T, M> { ... }
```

제네릭 타입의 상속과 구현

❖ Product.java

```
public class Product<T, M> {
    private T kind;
    private M model;

    public T getKind() { return this.kind; }
    public M getModel() { return this.model; }

    public void setKind(T kind) { this.kind = kind; }
    public void setModel(M model) { this.model = model; }
}

class Tv {}
```

제네릭 타입의 상속과 구현

❖ ChildProduct.java

```
public class ChildProduct<T, M, C> extends Product<T, M> {
	private C company;
	public C getCompany() { return this.company; }
	public void setCompany(C company) { this.company = company; }
}
```

제네릭 타입의 상속과 구현

❖ 제네릭 인터페이스를 구현할 경우

- 제네릭 인터페이스를 구현한 클래스도 제네릭 타입

제네릭 타입의 상속과 구현

❖ Storage.java

```
public interface Storage<T> {
    public void add(T item, int index);

    public T get(int index);
}
```

제네릭 타입의 상속과 구현

❖ Box.java

```
public class StorageImpl<T> implements Storage<T> {
    private T[] array;

    public StorageImpl(int capacity) {
        this.array = (T[]) (new Object[capacity]);
    }

    @Override
    public void add(T item, int index) {
        array[index] = item;
    }

    @Override
    public T get(int index) {
        return array[index];
    }
}
```

제네릭 타입의 상속과 구현

❖ ChildProductAndStorageExample.java

```
public class ChildProductAndStorageExample {
    public static void main(String[] args) {
        ChildProduct<Tv, String, String> product = new ChildProduct<>();
        product.setKind(new Tv());
        product.setModel("SmartTV");
        product.setCompany("Samsung");

        Storage<Tv> storage = new StorageImpl<Tv>(100);
        storage.add(new Tv(), 0);
        Tv tv = storage.get(0);
    }
}
```