Decorator

```
public interface DataSource {
	public void writeData(DataSource data);
	public DataSource readData();
}
public class FileDataSource implements DataSource {
	String name;
	public FileDataSource(String name) {
		this.name = name;
	}
	@Override
	public void writeData(DataSource data) {
			System.out.println("writing FileDataSource");
	}
	@Override
	public DataSource readData() {
		return null;
	}
}
public class DataSourceDecorator implements DataSource {
	protected DataSource dataSource;
	public DataSourceDecorator(DataSource dataSource) {
		this.dataSource = dataSource;
	}
	@Override
	public void writeData(DataSource data) {
		dataSource.writeData(data);
	}
	@Override
	public DataSource readData() {
		return dataSource.readData();
	}
}
public class CompressionDecorator extends DataSourceDecorator {
	public CompressionDecorator(DataSource dataSource) {
		super(dataSource);
	}
}
public class EncryptionDecorator extends DataSourceDecorator {
	public EncryptionDecorator(DataSource dataSource) {
		super(dataSource);
	}
}
```