1. 계산기 토큰 시스템 구현

문제 설명

- 수식 계산기에서 사용할 토큰 시스템을 객체지향 설계 원칙에 따라 구현하세요.
- 토큰은 숫자와 연산자로 구분되며, 연산자는 종류에 따라 서로 다른 우선순위를 가집니다.

구현 요구사항

1. 인터페이스 및 클래스 계층구조

```
XToken (인터페이스)
├── XTokenNumber (클래스)
├── XTokenParenthesis(클래스)
└── XTokenOperator (추상 클래스)
    ├── XTokenHighOrderArithmeticOperator(클래스)
    ├── XTokenLowOrderArithmeticOperator(클래스)
    └── XTokenLogicOperator(클래스)
```

2. Token 인터페이스

```java
public interface XToken {
    /**
     * 토큰의 값을 반환합니다.
     * @return 토큰의 문자열 값
     */
    String getValue();

    /**
     * 토큰의 우선순위를 반환합니다.
     * @return 우선순위 (높을수록 먼저 처리)
     */
    int getPrecedence();
}
```

3. 클래스별 요구사항

3.1 XTokenNumber

- 정수값을 저장
- 우선순위는 0 (연산에 사용되지 않음)
- 유효성 검증: 정수 형식이 아닌 경우 예외 발생

3.2 XTokenOperator (추상 클래스)

- 모든 연산자의 공통 기능 구현
- 유효한 연산자인지 검증하는 추상 메서드 포함

3.3 XTokenLogicOperator

- 비교 연산자: `<`, `>`, `==`, `<=`, `>=`
- 우선순위: 1

3.4 XTokenLowOrderArithmeticOperator

- 후순위 산술연산자: `+`, `-`
- 우선순위: 2

3.5 XTokenHighOrderArithmeticOperator

- 선순위 산술연산자: `*`, `/`, `%`
- 우선순위: 3

3.6 XTokenParenthesis

- 괄호 : `(`, `)`
- 우선순위: 4

4. 예외 처리

- `IllegalArgumentException`: 잘못된 토큰 값
- null 입력에 대한 처리

구현 예시

```
Token num = new XTokenNumber("42");
System.out.println(num.getValue());      // "42"
System.out.println(num.getPrecedence()); // 0

Token plus = new XTokenLowOrderArithmeticOperator("+");
System.out.println(plus.getValue());      // "+"
System.out.println(plus.getPrecedence()); // 2

Token multiply = new XTokenHighOrderArithmeticOperator("*");
System.out.println(multiply.getPrecedence()); // 3
```

작성 파일

1. `Token.java` - 인터페이스
2. `XTokenNumber.java` - 숫자 토큰 클래스
3. `XTokenParenthesis.java` - 괄호 클래스
4. `XTokenOperator.java` - 연산자 추상 클래스
5. `XTokenLogicOperator.java` - 비교 연산자 클래스

6. `XTokenLowOrderArithmeticOperator.java` - 후순위 산술연산자 클래스
7. `XTokenHighOrderArithmeticOperator.java` - 선순위 산술연산자 클래스

참고사항

- 인스턴스 생성이 가능한 모든 클래스는 불변(immutable)으로 설계하세요
- 방어적 프로그래밍을 적용하세요
- 테스트 코드는 수정하지 마세요