OSP-3rd Cycle

team#4

발표 : 송병우 201011338
박진성 201011334
근량 201013759

목차

1. Findbugs 대응
2. PMD 대응
3. Checkstyle 대응
4. Sonar 대응
5. OSP를 하면서 느낀점

1. FindBugs 대응

타입	1.1. 쓰지 않는 로컬변수
분석	로컬변수에 복잡한 로직이 들어가있는데 전혀 사용하지 않음
대응	해당 로컬변수를 삭제하여 수정

1. FindBugs 대응

타입	1.2. 랜덤에 대한 문제
분석	우리가 주로 사용한 랜덤 함수는 Math.random() 함수이었는데 이 함수에 대해서 성능상의 이슈를 지적
대응	해당 보고에서 추천하는 java.util.Random으로 변경하여 수정

1. FindBugs 대응

타입	1.3. Integer 램퍼 클래스의 생성자를 이용한 경우
분석	Primitive Type인 int를 사용하지 않고 Integer를 사용하려고 일부러 new Integer()를 통해 생성자를 호출 이 경우 3.5배정도의 속도가 느려진다는 보고
대응	로직 상으로 봤을 때 Primitive Type인 int를 사용하더라도 무방하다고 판단하여 int로 수정

1. FindBugs 대응

타입	1.4. Floating Point 타입에 대한 동등 연산을 한 경우
분석	Floating Point 타입 같은 경우 컴퓨터의 한계로 인해 반올림이 발생 하여 같은 값이더라도 0.000001 차이로 인해 다른값으로 연산할 버그가 존재할 수 있다.
대응	이에 해결방안으로 0.001값으로 동등성이 아닌 해당 값의 차를 비교하여 수정

1. FindBugs 대응

타입	1.5. .equals() 함수는 오버라이드 했는데 .hashCode()함수는 오버라이드 하지 않은 경우
분석	equals() 함수만 오버라이드 했을 경우, HashMap/HashTable과 같은 클래스에서 생각한 대로 동작하지 않을 가능성
대응	툴을 이용하여 .hashCode()도 만듬

2. PMD 대응

타입	2.1. 쓰지 않는 클래스를 임포트 한 경우
분석	대부분 작업하며 변경되면서 클래스를 임포트했다가 클래스를 사용하지 않게 된 경우가 많음
대응	이에 대해서는 전부 삭제하여 수정

2. PMD 대응

타입	2.2. 쓰지 않는 지역변수들
분석	대부분 작업하며 변경되면서 지역변수를 선언했다가 쓰지 않게 된 경우
대응	이에 대해서는 전부 삭제하여 수정

2. PMD 대응

타입	2.3. 쓰지 않는 Private 필드
분석	대부분 작업하면서 Private 필드를 선언했다가 쓰지 않게 된 경우
대응	이에 대해서는 전부 삭제하여 수정

2. PMD 대응

타입	2.4. 괄호가 너무 많이 쓰인 경우
분석	• 쓸데없이 괄호를 사용하여 가독성을 떨어트린 경우에 해당 • 대부분 수식에 대해서 우선순위가 명백히 성립함에도 괄호를 써서 가독성을 떨어트린 경우
대응	코드에 반영하여 명백한 우선순위가 있는 것들은 괄호를 삭제하여 수정

3. Checkstyle 대응

타입	3.1. Javadoc 주석이 없는 경우
분석	메소드와 클래스마다 Javadoc 주석이 있어야함
대응	미처 시간상 Javadoc 주석을 일일히 생성하지 못함

3. Checkstyle 대응

타입	3.2. 메소드 파라미터에 final이 안붙어 있는 경우
분석	Checkstyle에서는 파라미터의 변화가 없는 것에 대해 final을 붙일 것을 요구
대응	• 왜 붙여야 하는지 이유를 잘 모르겠음 • final을 붙일 경우 속도저하가 일어날 가능성이 있음 • 위의 2가지로 인해 수정하지 않음

3. Checkstyle 대응

타입	3.2. Whitespace를 적절하게 집어 넣지 않은 경우
분석	거의 대부분 수식에 대해서 whitespace를 넣지 않았다고 분석
대응	프로젝트 특징상 수학적 수식이 많은데 whitespace를 오히려 안넣었을 경우가 가독성이 더 좋았기 때문에 이 경고도 무시

3. Checkstyle 대응

타입	3.2. 숨겨진 필드
분석	전부 enum에서 쓰이는 생성자에 대해 숨겨진 필드라고 인식
대응	위의 분석사항으로 인해 enum에서는 private으로 생성자를 못 만듬. 따라서 대응 불가능

3. Checkstyle 대응

- 위 3가지로 인하여 우선순위가 낮다고 생각하여 **찾아보고 중요한 것만 수정**
- **대부분 무시**

4. Sonar 대응

타입	4.1. 매직 넘버로 상수를 사용한 경우
분석	• 로직에서 상수를 그대로 사용한 경우를 전부 분석함
대응	• 프로젝트 특징상 수학, 물리의 사용으로 인하여 수식이 많고, 시뮬레이터 프로그램이기 때문에 GUI상에 그려야할 경우가 많아서 어쩔수 없이 상수를 사용한 경우가 대부분이다. • 따라서, 일부의 분석된 이슈만 수정함.

4. Sonar 대응

타입	4.1. Naming Convention에 맞지 않는 네이밍을 사용한 경우
분석	• 우리가 자바에 익숙치가 않아서 자바에서 쓰이는 네이밍을 몰라 제대로 안 지킨 경우가 많기 때문에 많은 문제가 걸림 • 괄호의 사용, 꼬리에 오는 주석, CamelCase에 맞지 않는 네이밍 등이 많이 걸림
대응	• 왜 수정해야 하는지 이유 불문 • 현재 대부분은 CamelCase를 지키도록 수정 • 일부분 수학, 물리에서 쓰이는 수식에 대한 로직은 오히려 수학적 표기를 따름

5. OSP를 하면서 느낀점

- 코딩도 중요하지만 설계도 중요하다
- XP와 애자일의 탄생에 공감
- 문서작업이 오히려 너무 많아 수정을 기피