# 쾌적한
# Clone Checker

OOPT 3rd Cycle – Static Analysis

Software Modeling & Analysis

유준범 교수님


Team. T1

201111388 조연호

201211374 이창오

201211379 장종훈

201314196 양동혁

# 목차

Contents

# System Test Respond Report _ Specification, 1조

| | |
|---|---|
| 위치 | Stage 2030 - Activity 2131. Define Essential Use Cases |
| 문제 | Analyze function 함수의 개수와 이름을 분석하였다면, 그 분석 결과를 저장하는 과정이 필요한데, 문서상에선 명시되지 않았다.<br>Analyze Variable 변수의 개수와 이름을 분석하였다면, 그 분석 결과를 저장하는 과정이 필요한데, 문서상에선 명시되지 않았다.<br>Analyze Preprocessor 전처리기의 개수와 이름을 분석하였다면, 그 분석 결과를 저장하는 과정이 필요한데, 문서상에선 명시되지 않았다.<br>Analyze Annotation 프로그램 상에서는 주석의 수로 유사도를 비교하게 되는데, 문서상에서는 주석의 수를 저장하는 과정이 나타나지 않는다.<br>Analyze Line 프로그램 상에서는 코드의 라인 수로 유사도를 비교하게 되는데, 문서상에서는 코드의 라인수를 저장하는 과정이 나타나지 않는다. |
| 대응 | Typical Courses of Events 내용에 '(S) : Files 인스턴스 내 변수에 결과값을 저장한다.'는 문구를 추가하였다. |

| | |
|---|---|
| 위치 | Stage 2030 - Activity 2131. Define Essential Use Cases |
| 문제 | Calculate File, Calculate Line Sync-Rate, Calculate Variable Sync-Rate, Calculate Preprocessor Sync-Rate, Calculate Annotation Sync-Rate 모두 각 과정에서 일치율을 계산하고 저장하는 과정이 명시되어 있지 않다. |
| 대응 | 각 Use Case의 Typical Courses of Events 내용에 '(S) : 변수에 결과값을 저장한다.'를 추가하였다. |

# System Test Respond Report _ Specification, 5조

| | |
|---|---|
| 위치 | Stage 1000 - Activity 1010. Refine Plan |
| 문제 | “A. 소스 코드 파일 분석 작업은 5초 이내로 수행되어야 한다”고 명세되어 있는데, 분석 대상이 되는 코드의 최대량 또는 기준량이 명시되어 있지 않았으며, “B. 기준 소스 코드 파일 변경 시 2초 이내로 비교 결과가 변경되어야 한다.” 역시 같은 문제가 있다. 대응서에는 100개 이하의 파일이 들어있는 폴더를 선택해야 한다고 기술되어 있지만 문서에서는 찾아볼 수가 없다. |
| 대응 | Performance Requirements에 ‘소스 코드 파일이 100개 이하로 들어있는 폴더를 선택해야 한다.’를 추가하였다. |

| | |
|---|---|
| 위치 | Stage 2030 - Activity 2131. Define Essential Use Cases |
| 문제 | Start 대응서에는 소스 코드 파일의 크기는 무관하다고 적혀 있는데 0바이트인 파일은 안 된다고 적혀 있다. |
| 대응 | Stage 1000 - Activity 1004. Record Terms in Glossary 중 ‘소스 코드 파일’ 정의를 ‘파일 크기가 0바이트 초과인 소스 코드가 담긴 .c 확장자 파일. 파일명 양식은 ‘학번_이름.c’ 형태여야 한다.’로 수정하였다. |

| | |
|---|---|
| 위치 | Stage 2040 - Activity 2141. Design Real Use Cases |
| 문제 | Change Annotation C 스타일 주석의 /*은 //로 치환한다고 기술되어 있지만 */는 어떻게 처리할지 기술되어 있지 않다. |
| 대응 | Typical Courses of Events 내용에 ‘(S) : ‘*/’는 변환하지 않고 그대로 방치한다.’를 추가하였다. |

# Static Analysis Report _ PMD Warnings

## Analyze.java

✓ CollapsibleIfStatements : 2개

중첩된 if문이 존재하여 발생한 문제로, 하나의 if문에 '&&'로 처리하여 해결하였다.

```
061         if(source.get(i).contains("(") && source.get(i).indexOf("(") > source.get(i).indexOf(listType.get(j))){
062           if(!source.get(i).contains(";")){
063             numOfFunction++;
064             listFunction.add(source.get(i).substring(source.get(i).indexOf(listType.get(j)), source.get(i).indexOf(")")+1));
065           }
066         }
```

```
if( source.get(i).contains(listType.get(j)) && source.get(i).contains("(") &&
        source.get(i).indexOf("(") > source.get(i).indexOf(listType.get(j)) && !source.get(i).contains(";")){
    numOfFunction++;
    listFunction.add(source.get(i).substring(source.get(i).indexOf(listType.get(j)), source.get(i).indexOf(")")+1));
}
```

# Static Analysis Report _ PMD Warnings

## Analyze.java

✓ UselessParentheses : 1개

무의미한 괄호가 존재하여 발생한 문제로, 삭제하여 해결하였다.

```
084                listVariable.add( (tempString.substring(0, tempString.indexOf(","))).trim() );
```

```
listVariable.add( tempString.substring(0, tempString.indexOf(",")).trim() );
```

# Static Analysis Report _ PMD Warnings

## Calculate.java

- ✓ UselessParentheses : 6개

  무의미한 괄호가 존재하여 발생한 문제로, 삭제하여 해결하였다.

- ✓ Unnecessary : 6개

  불필요한 삽입어구가 존재하여 발생한 문제로, 삭제하여 해결하였다.

```
$ pmd pmd -dir src/main/java/ -R java-unnecessary
Analyze.java:84:        Useless parentheses.
Calculate.java:100:     Useless parentheses.
Calculate.java:103:     Useless parentheses.
Calculate.java:170:     Useless parentheses.
Calculate.java:173:     Useless parentheses.
Calculate.java:236:     Useless parentheses.
Calculate.java:239:     Useless parentheses.
```

- ✓ Unused Code : 2개

  사용하지 않는 코드가 존재하여 발생한 문제로, 삭제하여 해결하였다.

```
Calculate.java:203:     Avoid unused local variables such
    as 'PreDiff'.
Calculate.java:204:     Avoid unused local variables such
    as 'PreDiff1'.
```

# Static Analysis Report _ PMD Warnings

## Controller.java

- ✓ CollapsibleIfStatements : 1개

  중첩된 if문이 존재하여 발생한 문제로, 하나의 if문에 '&&'로 처리하여 해결하였다.

- ✓ JumbledIncrementer : 2개

  for문의 incrementer를 for문 내에서 사용하여 발생한 문제였다.

- ✓ Unnecessary : 4개

  불필요한 삽입어구가 존재하여 발생한 문제로, 삭제하여 해결하였다.

```
Controller.java:371:    Useless parentheses.
Controller.java:379:    Useless parentheses.
Controller.java:435:    Useless parentheses.
Controller.java:514:    Useless parentheses.
```

- ✓ Imports : 7개

  중복으로 import된 패키지가 존재하여

  발생한 문제로, 삭제하여 해결하였다.

```
$ pmd pmd -dir src/main/java/ -R java-imports
Controller.java:7:      Avoid duplicate imports such as '
   java.io.BufferedReader'
Controller.java:8:      Avoid duplicate imports such as '
   java.io.File'
Controller.java:9:      Avoid duplicate imports such as '
   java.io.FileNotFoundException'
Controller.java:10:     Avoid duplicate imports such as '
   java.io.FileReader'
Controller.java:11:     Avoid duplicate imports such as '
   java.io.IOException'
Controller.java:12:     Avoid importing anything from the
    package java.lang
Controller.java:198:    Unnecessary use of fully
   qualified name 'java.io.File' due to existing import '
   java.io.*'
```

# Static Analysis Report _ FindBugs Result

## Analyze.java

✓ Unused field : 1개

사용하지 않는 변수(String tempFolderPath)가 존재하여 발생한 문제로, 변수를 삭제하여 해결하였다.

## Calculate.java

✓ DM_STRING_CTOR : 3개

String 선언 시 생성자를 사용하여 메모리 낭비가 발생한 문제로, String 선언 시 직접 선언하여 해결하였다.

```
String str1 =  new String(files.get(centerNum).getListFunction().get(i));
```

```
String str1 = files.get(centerNum).getListFunction().get(i);
```

# Static Analysis Report _ FindBugs Result

## Controller.java

- ✓ NM_METHOD_NAMING_CONVENTION : 2개

  메소드 이름이 대문자로 시작하여 발생한 문제로, 메소드 이름을 소문자로 시작하도록 변경하여 해결하였다.

```
213⊖    public void DisplayResult() {
```

```
213⊖    public void displayResult() {
```

# Static Analysis Report _ FindBugs Result

## Controller.java

- ✓ UWF_UNWRITTEN_FIELD : 1개

  JLabel 변수가 초기화되지 않고 사용이 되어 발생한 문제로, 선언 시 초기화하여 해결하였다.

```
JLabel lbTempTotalSyncRate;
```

```
JLabel lbTempTotalSyncRate = new JLabel();
```

# Static Analysis Report _ FindBugs Result

## Controller.java

- ✓ DM_DEFAULT_ENCODING : 1개

```
BufferedReader br = new BufferedReader(new FileReader(folderPath + fileName));
```

```
filePath = folderPath + fileName;

BufferedReader br = new BufferedReader(new FileReader(filePath));
```

# Static Analysis Report _ CheckStyle Warnings

## Analyze.java

- ✓ LeftCurlyCheck : 5개

  '{'의 위치 때문에 발생한 문제로, 수정하였다.

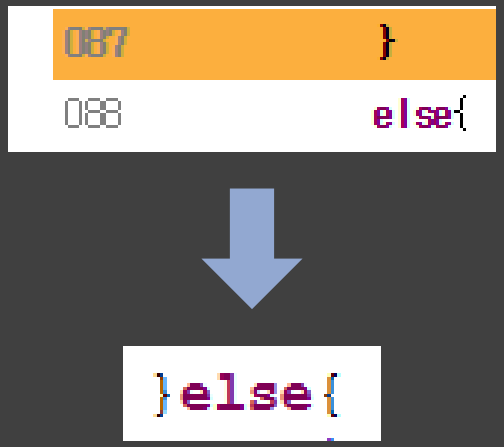
```
209     if(tempString1.contains(";") && !tempString1.contains("for"))
210     {
```

```
if(tempString1.contains(";") && !tempString1.contains("for")){
```

# Static Analysis Report _ CheckStyle Warnings

## Analyze.java

- ✓ RightCurlyCheck : 7개
  '}'의 위치 때문에 발생한 문제로, 수정하였다.



# Static Analysis Report _ CheckStyle Warnings

## Analyze.java

- ✓ NeedBracesCheck : 1개

  if문에 '{ }'가 없어서 발생한 문제로, 추가하였다.

```
259             if(source.get(i+j).contains("}"))
260                 break;
```

```
if(source.get(i+j).contains("}")){
    break;
}
```

# Static Analysis Report _ Code Coverage

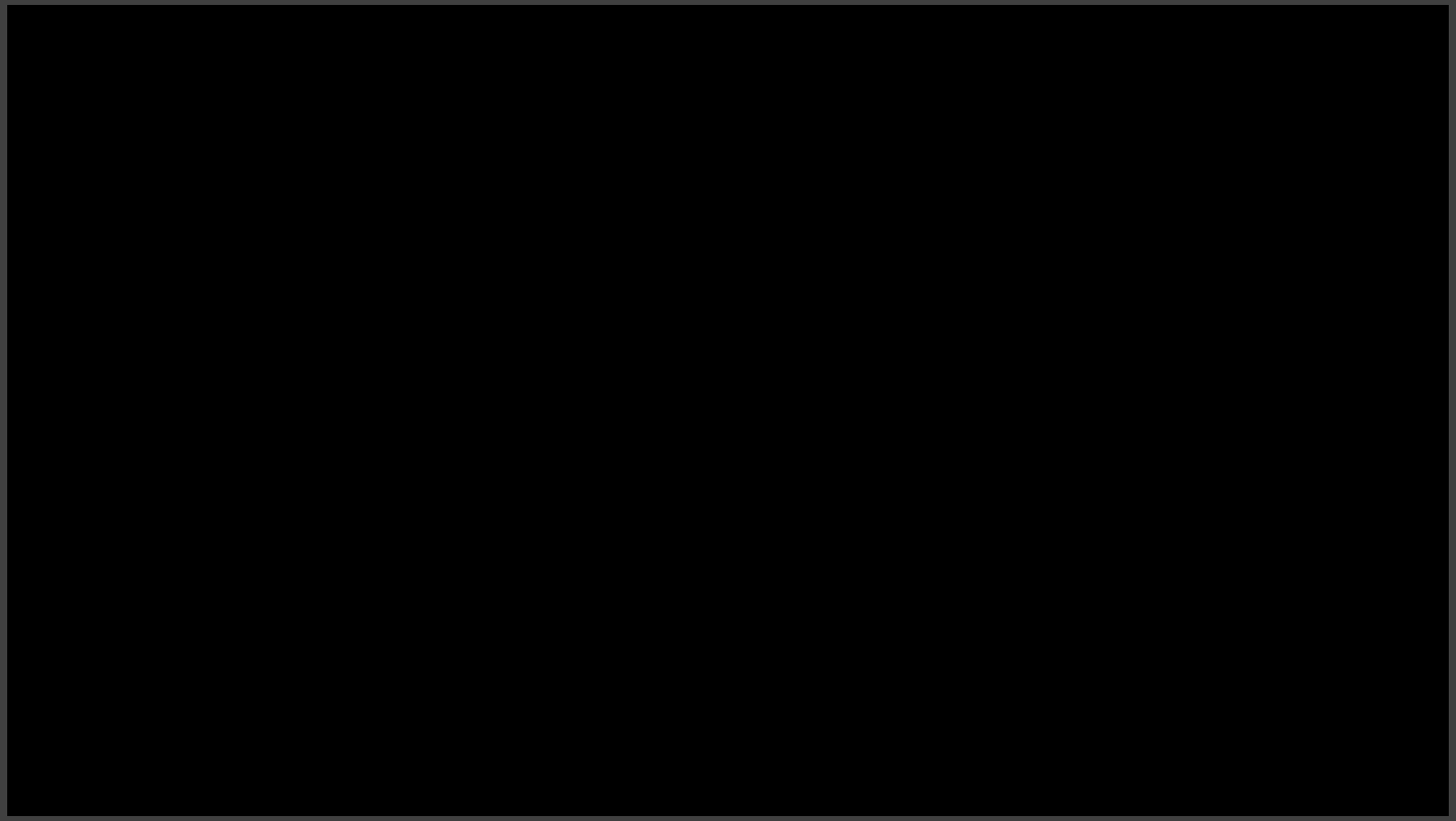
## JUnit 테스트 범위 확대

✓ 수정 전 : Analyze 클래스, Calculate 클래스

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty | Missed | Lines | Missed | Methods | Missed | Classes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Files | | 92% | | n/a | 2 | 19 | 3 | 33 | 2 | 19 | 0 | 1 |
| Main | | 0% | | n/a | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 1 |
| Controller.new ActionListener() { ... } | | 0% | | n/a | 2 | 2 | 3 | 3 | 2 | 2 | 1 | 1 |
| Controller.new ActionListener() { ... } | | 0% | | n/a | 2 | 2 | 7 | 7 | 2 | 2 | 1 | 1 |
| Controller.new MouseAdapter() { ... } | | 0% | | 0% | 4 | 4 | 5 | 5 | 2 | 2 | 1 | 1 |
| Controller.new KeyAdapter() { ... } | | 0% | | 0% | 4 | 4 | 6 | 6 | 2 | 2 | 1 | 1 |
| Controller.RoundedBorder | | 0% | | n/a | 5 | 5 | 13 | 13 | 5 | 5 | 1 | 1 |
| Controller.new MouseAdapter() { ... } | | 0% | | 0% | 9 | 9 | 23 | 23 | 4 | 4 | 1 | 1 |
| Analyze | | 72% | | 75% | 24 | 73 | 36 | 160 | 1 | 12 | 0 | 1 |
| Calculate | | 63% | | 66% | 24 | 39 | 28 | 94 | 8 | 14 | 0 | 1 |
| Controller | | 0% | | 0% | 41 | 41 | 260 | 260 | 13 | 13 | 1 | 1 |
| Total | 2,665 of 4,269 | 38% | 121 of 246 | 51% | 119 | 200 | 382 | 602 | 43 | 77 | 8 | 11 |

✓ 수정 후 : Controller 클래스까지 테스트 범위 확대하여 Code Coverage 증가

# Demonstration



https://youtu.be/6asJ5fEBCX0

# Epilogue _ OOPT에 대하여

## 장점

- ✓ 프로그램을 체계적으로 제작할 수 있다.
- ✓ 조원들과 생각을 동기화하기 쉽다.
- ✓ 프로젝트를 쉽게 이해할 수 있다.
- ✓ 프로그램 구현이 쉽고 빠르다.
- ✓ 프로그램 구현 시 실수를 줄일 수 있다.
- ✓ 프로그램 관리가 수월하다.

## 단점

- ✓ 많은 Stage와 Activity로 인하여
  작은 규모의 프로젝트 또한 작업 시간이 길다.
- ✓ 관련된 정보를 쉽게 찾을 수 없다.
- ✓ 다량의 보고서를 작성해야 하므로
  프로그램 구현까지의 시간이 길어진다.

**프로그램 구현 단계부터 진가가 나타나는 OOPT**

# Epilogue _ SV팀과 협업

## 1조

- ✓ Member
  강성길, 김민재, 이종찬, 한지승
- ✓ 많은 양의 피드백
- ✓ Slack을 이용한 수월한 의사소통
- ✓ 다양한 Tool 경험

## 5조

- ✓ Member
  라가영, 서지혁
- ✓ 이해하기 쉬운 테스트 명세와 테스트 프레임
- ✓ 어려웠던 의사소통
- ✓ 초기 CTIP 환경 구축 단계에서의 어려움

CTIP 환경을 100% 활용하지 못한 아쉬움

# 쾌적한
# Clone Checker

고맙습니다.

Software Modeling & Analysis

유준범 교수님