

Software Verification

1st System Testing

T 3

200711453 류진렬

200711477 황진수

200711428 박기성

201360220 황 민

200312489 유현덕

Contents

- Category Partitioning Testing
- Pairwise Combinational Testing
- Test Result
- JDepend & Clover
- Additional Errors

Category Partitioning Testing

Category Partitioning

▪ Step 1. Choosing categories

- 요구사항 명세서의 Functional Requirement를 기반으로 항목을 분류하고 unit들을 정의

분류	항목
File I/O	파일 저장하기
	파일 불러오기
그리기	그리기 도구
	색 편집
	선 두께
효과	색 선택
	데칼코마니
	그림자효과
	점선으로 변경
	색 채우기
	초기화

Category Partitioning

- Step 2. Identify Representative Value

- 파일 불러오기
 - 모든 종류의 이미지 파일 오픈
 - 이미지 파일이 아닌 파일 오픈

- 파일 저장하기
 - 이미지 파일 저장

Category Partitioning

- Step 2. Identify Representative Value
- 그리기 도구
 - 연필
 - 선
 - 브러시 – normal
 - 브러시 – special1
 - 브러시 – special2
 - 도형 – 삼각형
 - 도형 – 원
 - 도형 – 사각형
 - 지우개

Category Partitioning

- Step 2. Identify Representative Value
- 색 선택
 - 선택
 - 미선택
- 데칼코마니
 - 선택
 - 미선택
- 그림자효과
 - 선택
 - 미선택

Category Partitioning

- Step 2. Identify Representative Value
- 점선으로 변경
 - 선택
 - 미선택
- 색 채우기
 - 선택
 - 미선택
- 초기화
 - 선택
 - 미선택

Category Partitioning

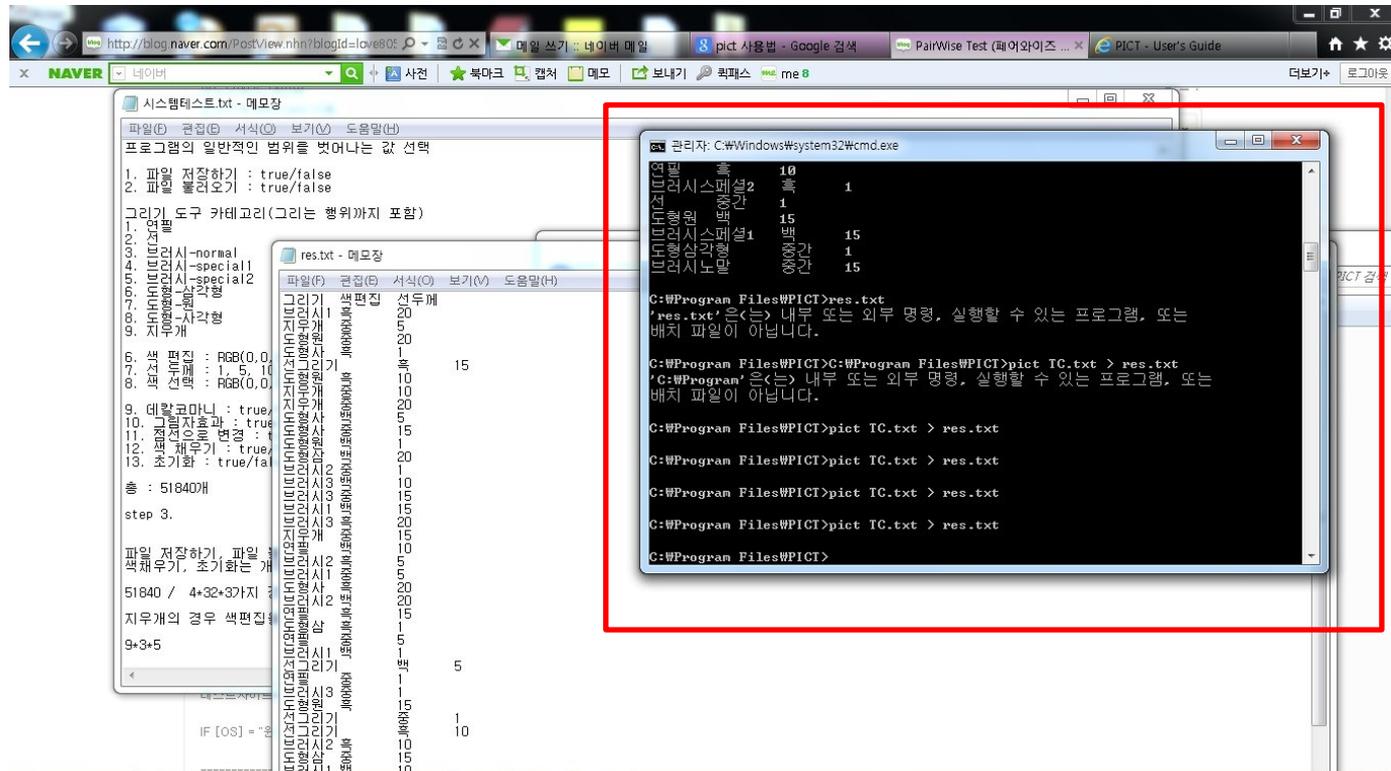
▪ Step 3. Generate Test Case Specifications

- 파일 저장하기 (2) * 파일 불러오기 (1) * 그리기도구 (9) * 색 편집 (3) * 선 두께 (5) * 색 선택 (2) * 데칼코마니 (2) * 그림자효과 (2) * 점선으로 변경(2) * 색 채우기(2) * 초기화 (2) = 17280 개의 test cases.
- Single constraints를 적용
 - 파일 저장하기, 파일 불러오기, 색 선택, 데칼코마니, 그림자효과, 점선으로 변경, 색채우기, 초기화는 모두 single constraint
- Single constraints 적용 후 test cases = 그리기도구(9) * 색 편집(3) * 선 두께(5) + 9 = 135 + 9 = 143 가지

Pairwise Combination Testing

Pairwise Combination Testing

- Single constraints 적용 후 test cases = 그리기 도구(9) * 색 편집(3) * 선 두께(5) = 135가지의 test case들을 PICT(pairwise combination test tool)을 사용하여 추림



Pairwise Combination Testing

- 135가지의 test case들이 45가지로 추려짐
(조건을 주어 지우개의 경우 색이 포함되지 않으므로 임의로 중 값을 설정)

TC.txt 의 내용 :

그리기 : 연필, 선그리기, 브러시1, 브러시2, 브러시3, 도
형삼, 도형원, 도형사, 지우개

색편집 : 흑, 중, 백

선두께 : 1, 5, 10, 15, 20

IF [그리기] = "지우개" THEN [색편집] = "중";

Pairwise Combination Testing

- 135가지의 test case들이 45가지로 추려짐 +
(single constraint test case 9개)
(조건을 주어 지우개의 경우 색이 포함되지 않으므로 임의로 중 값을 설정)

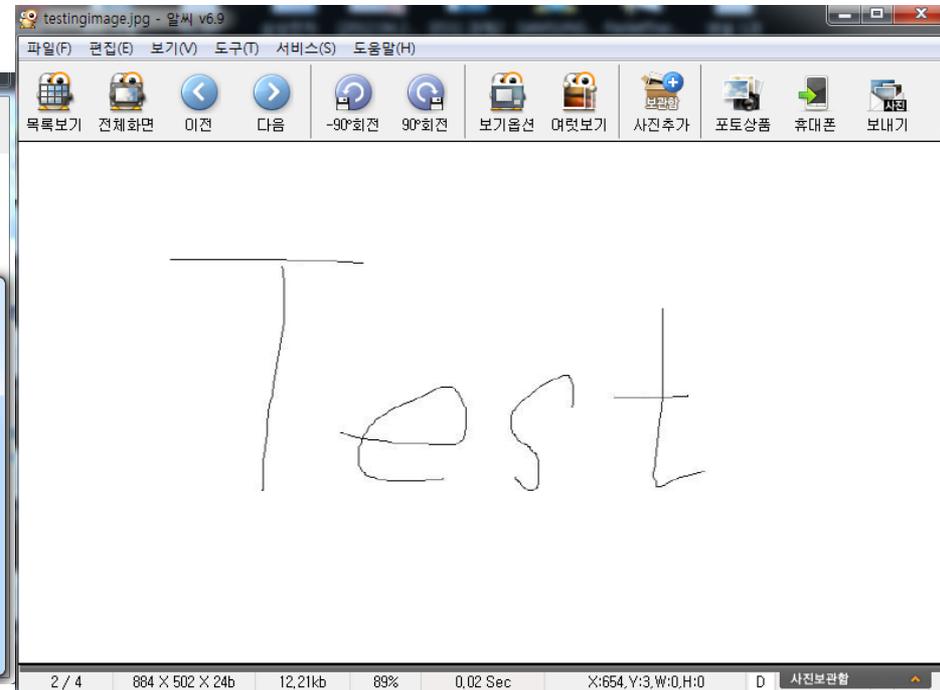
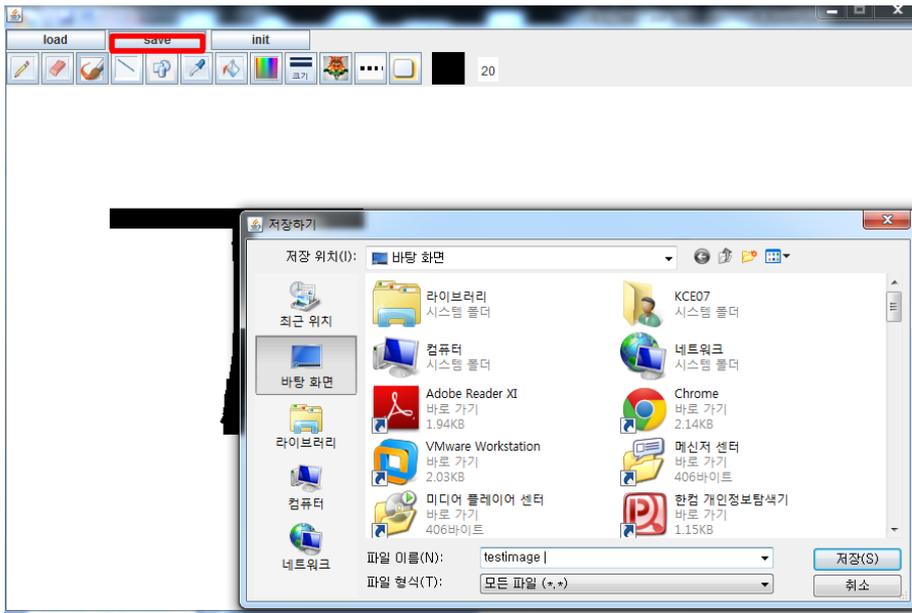
res.txt 의 내용 :

파일(F)	편집(E)	서식(O)	보기(V)	도움말(H)
그리	1	선	두	께
리	1	20		
개	1	5		
원	1	20		
사	1	20	15	
기	1	10		
개	1	10		
개	1	20		
사	1	5		
원	1	15		
사	2	1		
시	3	1		
시	3	1		
시	3	1		
시	3	1		
개	2	1		
시	2	1		
시	2	1		
시	2	1		
삼	1	1		
시	1	5		5
리	1	1		
기	1	1		
시	3	1		
원	1	1		
리	2	1		10
기	2	1		
시	1	1		
삼	1	1		
시	1	1		
원	1	1		
기	1	1		20

Test Result

Test Case No1. (single constraint)

- 카테고리 항목 (input)
 - 파일 저장하기
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 그림판의 현재 그림을 경로와 파일명을 입력 받아 이미지 파일(jpg)로 저장한다.
- 테스트 결과(Pass)



Test Case No2. (single constraint)

- 카테고리 항목 (input)
 - 파일 불러오기 - 여러 포맷의 이미지 파일
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 기존의 이미지 파일(.bmp, .jpg, .gif, .png)을 그림판으로 불러온다.
- 테스트 결과(Pass)

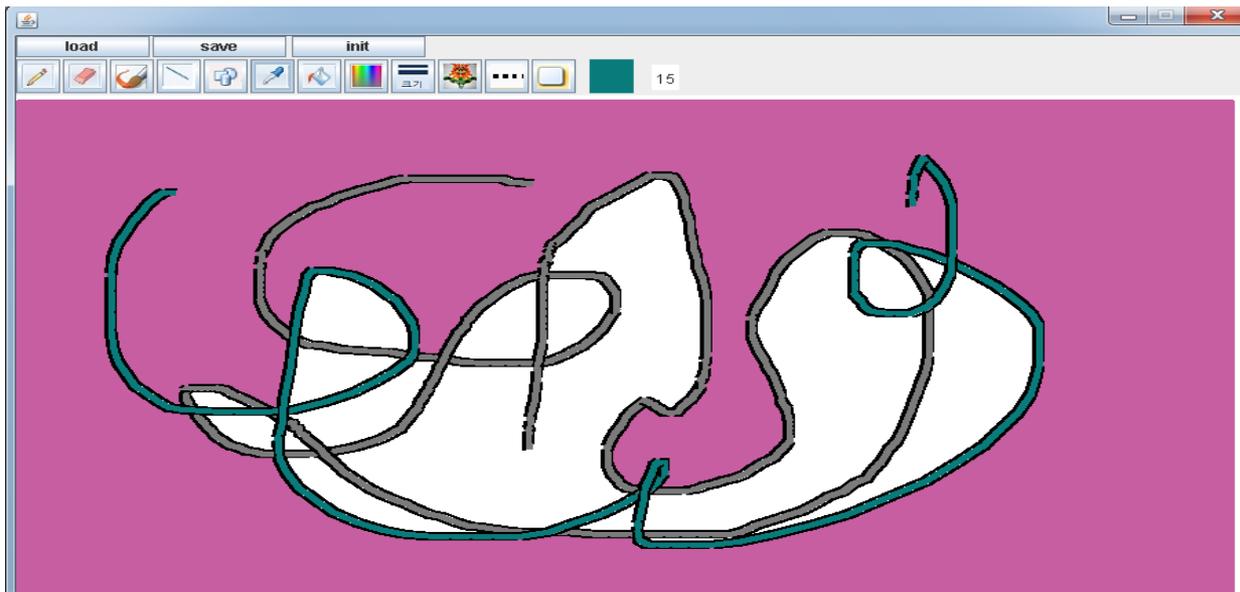


Test Case No3. (single constraint)

- 카테고리 항목 (input)
 - 파일 불러오기 - 이미지 파일이 아닌 경우
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 불러오는 파일이 이미지 파일이 아닌 경우 오류 메시지를 출력한다.
- 테스트 결과(Fail)
 - 아무 것도 출력 되지 않음.

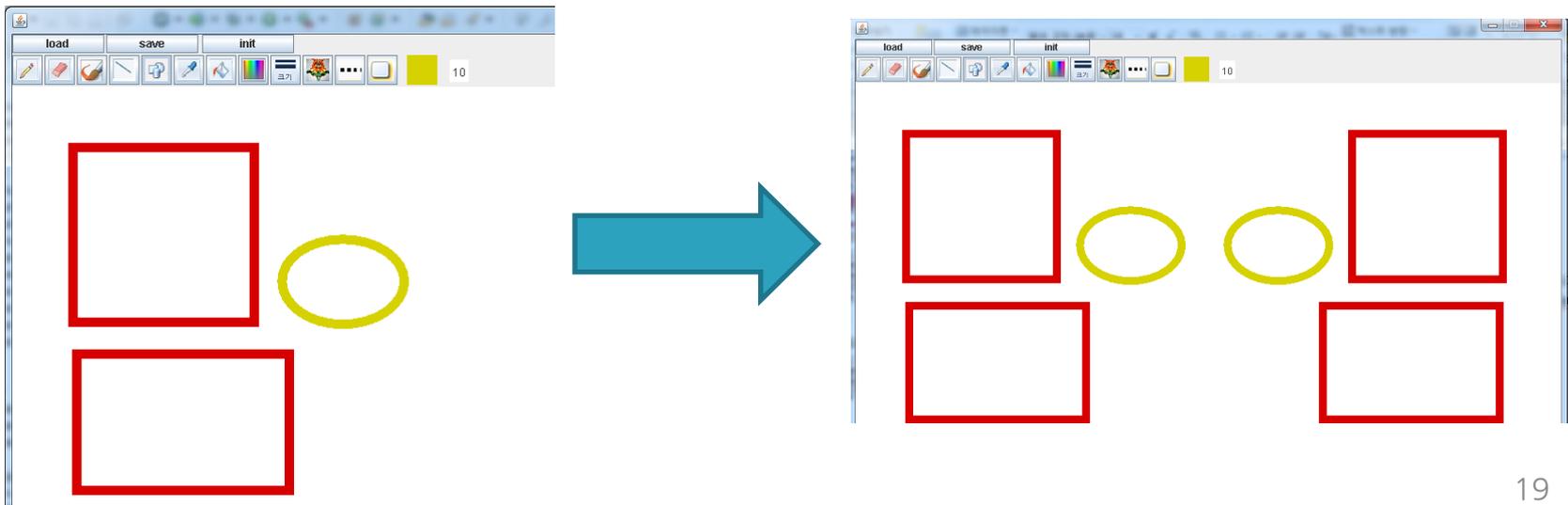
Test Case No4. (single constraint)

- 카테고리 항목 (input)
 - 효과 : 색 선택
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 색 선택 버튼을 누르고 그림의 색깔을 클릭하면 해당 색깔로 선택이 되어야 한다.
- 테스트 결과(Pass)



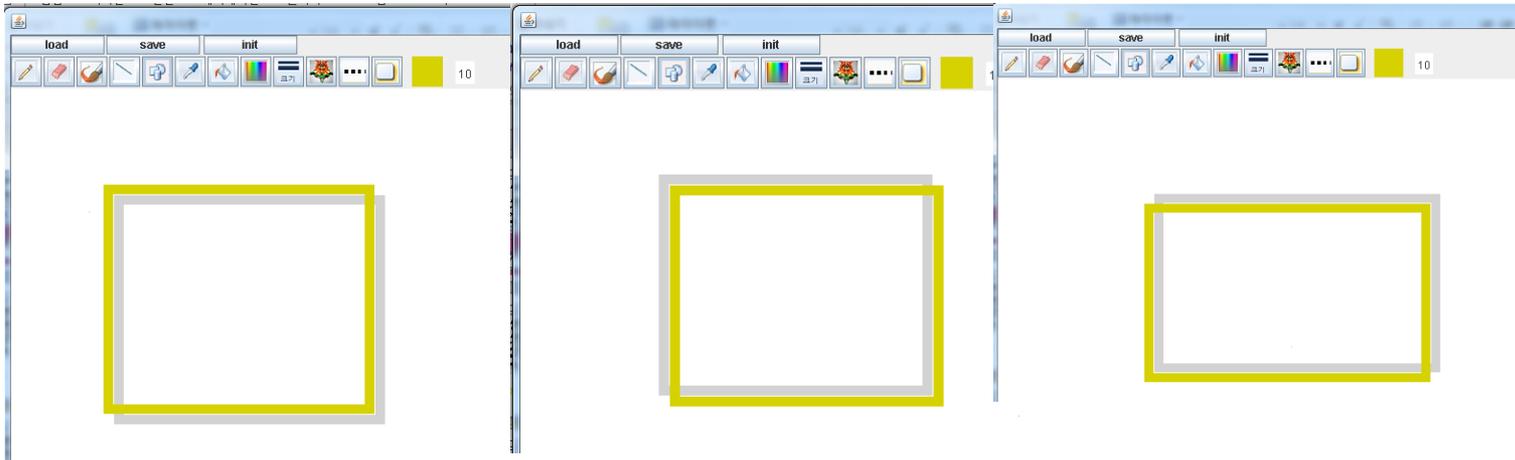
Test Case No5. (single constraint)

- 카테고리 항목 (input)
 - 효과 : 데칼코마니
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 데칼코마니 효과를 선택할 경우 현재 그려진 그림이 복사되어 양쪽으로 데칼코마니 효과를 형성할 수 있어야 한다.
- 테스트 결과(Pass)



Test Case No6. (single constraint)

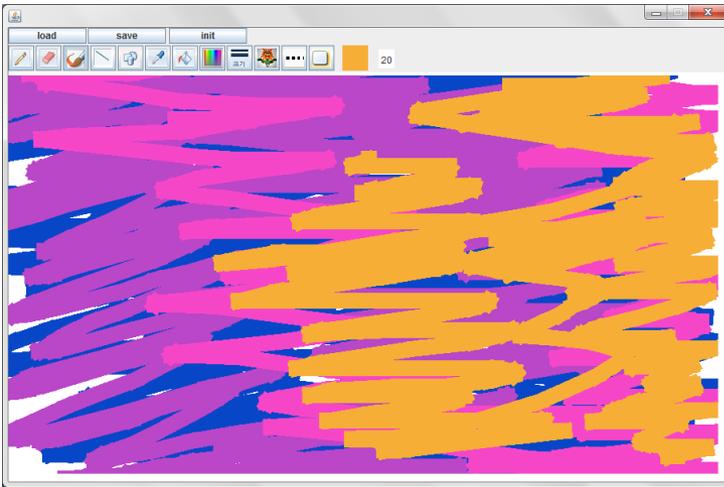
- 카테고리 항목 (input)
 - 효과 : 그림자효과
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 그림자효과를 선택할 경우 그려져 있는 그림의 그림자효과가 클릭하는 위치에 따라 4방위로 생겨야 한다.
- 테스트 결과(Fail) – 어디를 클릭해야 어느 방향으로 그림자가 생기는지 명확하지 않음



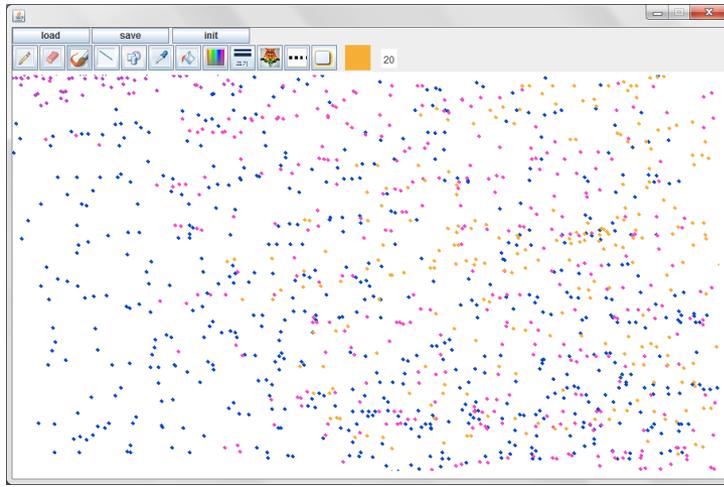
Test Case No7. (single constraint)

- 카테고리 항목 (input)
 - 효과 : 점선으로 변경
- 예상되는 결과 (output)
 - 그림판 전 영역에서 사용자가 그린 선을 점선으로 바꿔준다.
- 테스트 결과(Fail)
 - 그려진 선을 점들로 변환 시에 색깔은 반영하지만 선의 두께는 반영하지 않고, 선의 두께에 관계없이 똑같은 점들로 구성됨

적용 전

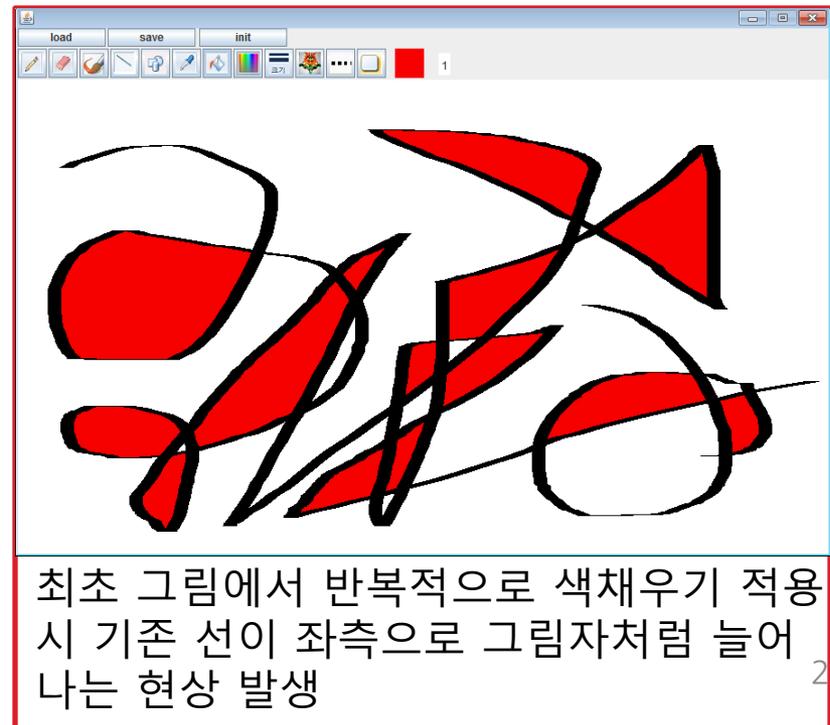
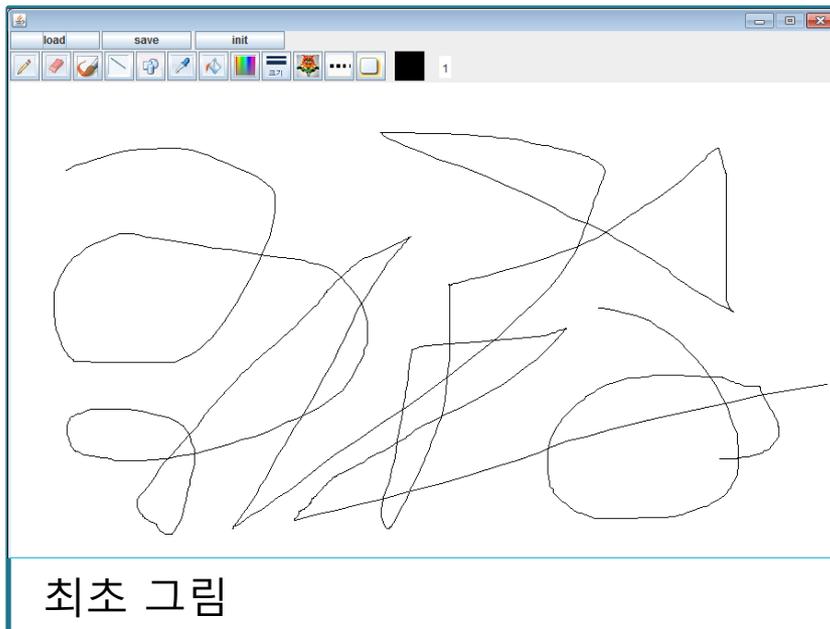


적용 후



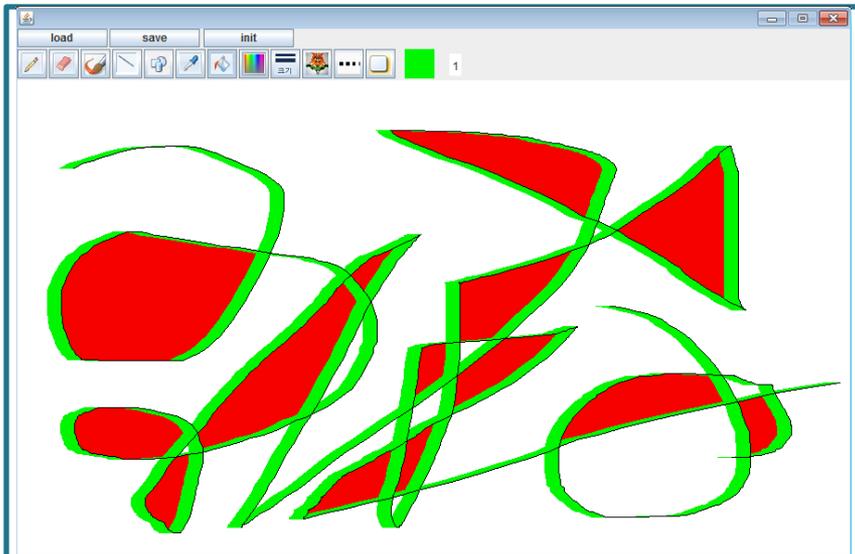
Test Case No8. (single constraint)

- 카테고리 항목 (input)
 - 효과 : 색 채우기
- 예상되는 결과 (output)
 - 마우스로 선택된 영역이 선택된 색으로 채워진다.
- 테스트 결과(Fail) - 다른 그리기 도구 시에도 동일

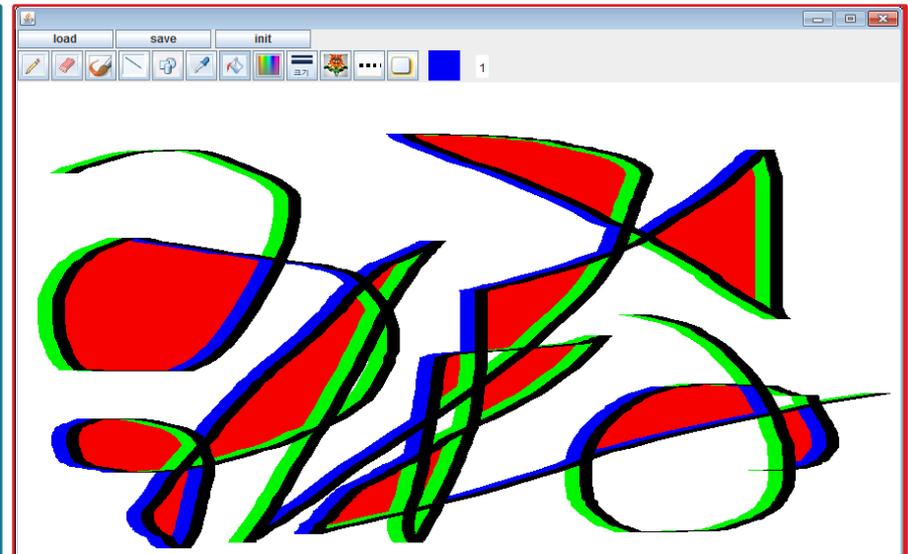


Test Case No8. (single constraint)

- 테스트 결과(Fail) - 다른 그리기 도구 시에도 동일



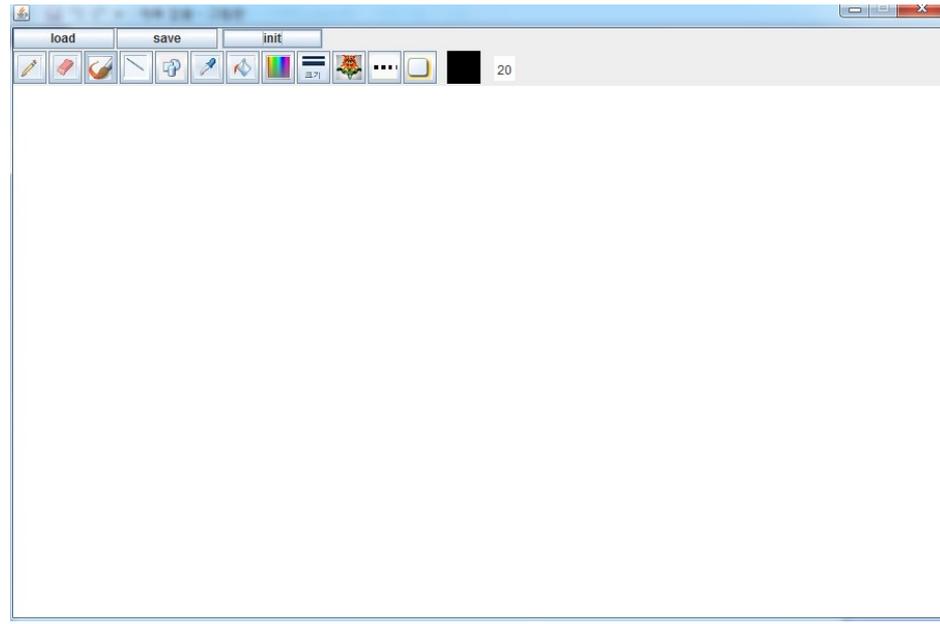
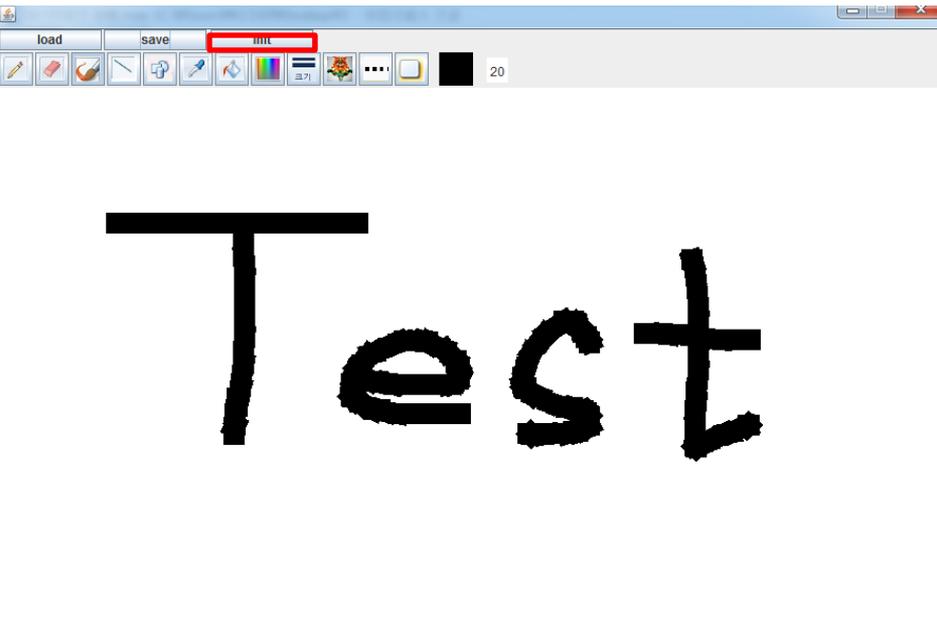
그림자 현상과 같이 늘어난 영역을 색채우기 적용하면 한 영역으로 인식하여 색채우기 적용됨



늘어난 영역도 구분된 영역인 것처럼 개별적으로 색채우기 적용됨

Test Case No9. (single constraint)

- 카테고리 항목 (input)
 - 초기화
- 예상되는 결과 (output)
 - Requirement specification에 명시된 내용처럼 그림판의 모든 그림을 지우고 초기 화면으로 바꾼다.
- 테스트 결과(Pass)



Test Case No10. (combinational test case)

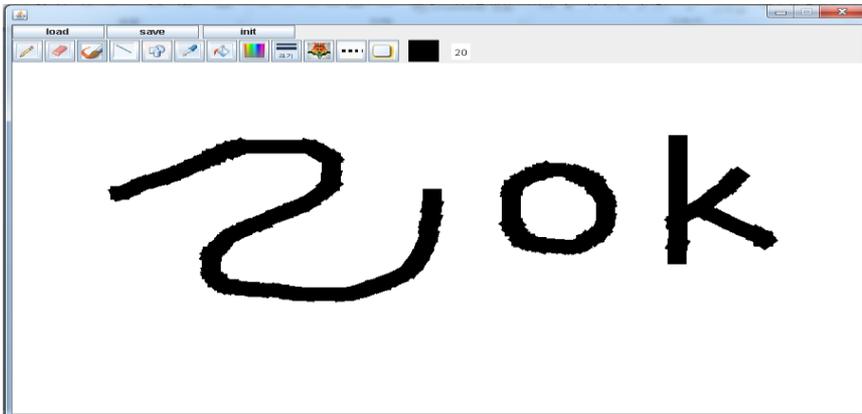
▪ 카테고리 항목 (input)

- 그리기 도구 : 브러시(노말)
- 색 편집 : RGB(0, 0, 0)
- 선 두께 : 20

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 브러시-노말 의 형태로 검정색의 선 두께 20짜리 그림이 이상 없이 그려져야 한다.

▪ 테스트 결과(Pass)



Test Case No12. (combinational test case)

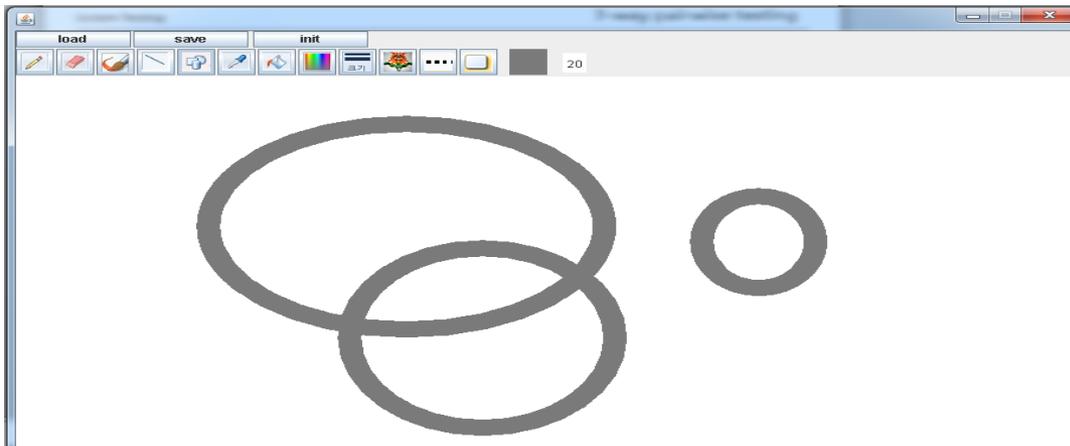
▪ 카테고리 항목 (input)

- 그리기 도구 : 도형 - 원
- 색 편집 : 회색 (RGB : 127, 127, 127)
- 선 두께 : 20

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 도형(원)을 두께 20, 회색으로 그릴 수 있어야 한다.

▪ 테스트 결과(Pass)



Test Case No14. (combinational test case)

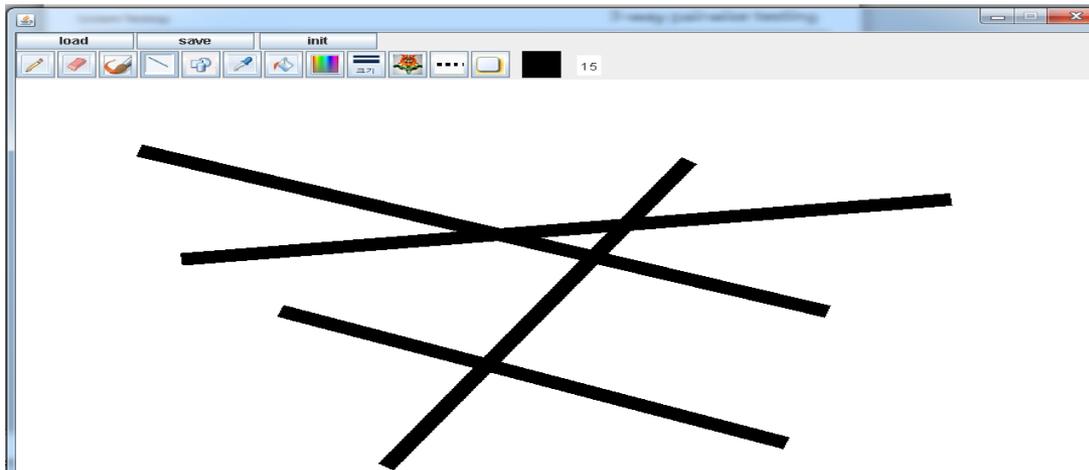
▪ 카테고리 항목 (input)

- 그리기 도구 : 선그리기
- 색 편집 : 흑색 (RGB : 0, 0, 0)
- 선 두께 : 15

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 선을 두께 15, 검정색으로 그릴 수 있어야 한다.

▪ 테스트 결과(Pass)



Test Case No18. (combinational test case)

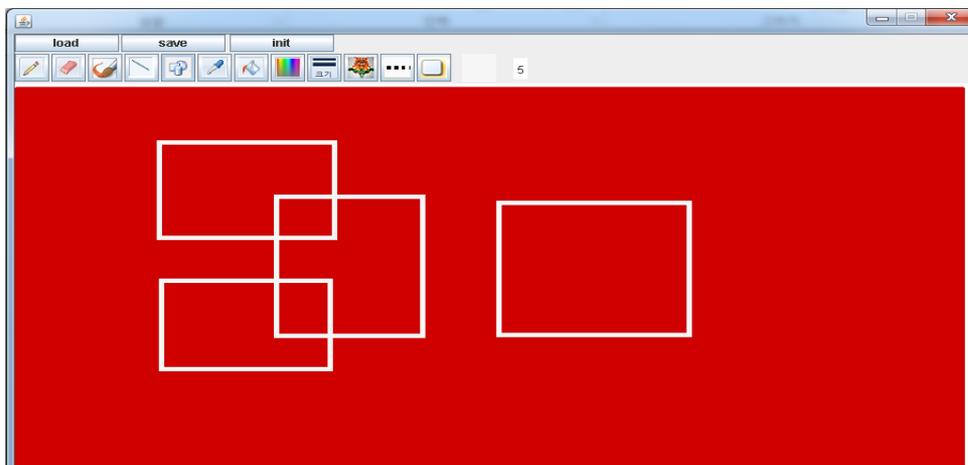
▪ 카테고리 항목 (input)

- 그리기 도구 : 도형 - 사각형
- 색 편집 : 흰색 (RGB : 246, 246, 246)
- 선 두께 : 5

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 도형(사각형)을 두 개 5, 흰색으로 그릴 수 있어야 한다.

▪ 테스트 결과(Pass)



Test Case No21. (combinational test case)

▪ 카테고리 항목 (input)

- 그리기 도구 : 도형 - 삼각형
- 색 편집 : 흰색 (RGB : 246, 246, 246)
- 선 두께 : 20

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 도형(삼각형)을 두 개 20, 흰색으로 그릴 수 있어야 한다.

▪ 테스트 결과(Fail)

- 도형 - 삼각형 그리기가 메뉴에 존재하지 않아서 테스트 불가능

Test Case No23. (combinational test case)

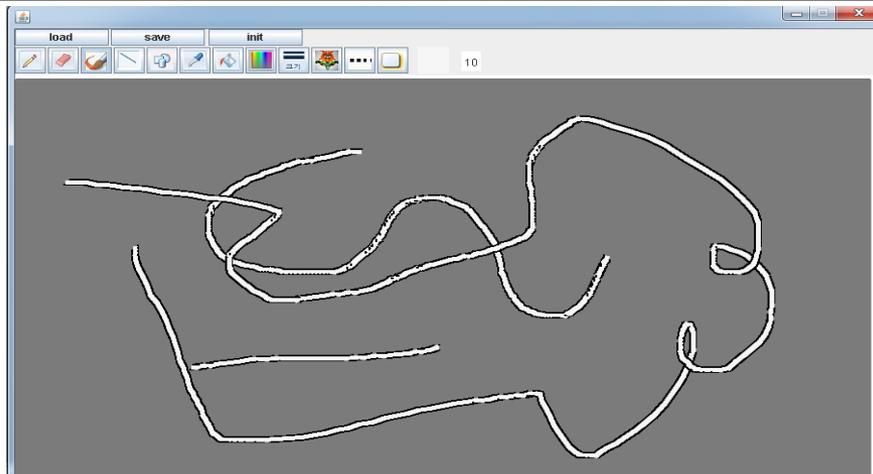
카테고리 항목 (input)

- 그리기 도구 : 브러시 - special2
- 색 편집 : 흰색 (RGB : 246, 246, 246)
- 선 두께 : 10

예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 브러시(special2) 그리기 도구로 두께 10, 흰색을 그릴 수 있어야 한다.

- **테스트 결과(Fail)** - 두께를 선택해도 적용되지 않고, 무조건 같은 두께로 나온다.



Test Case No28. (combinational test case)

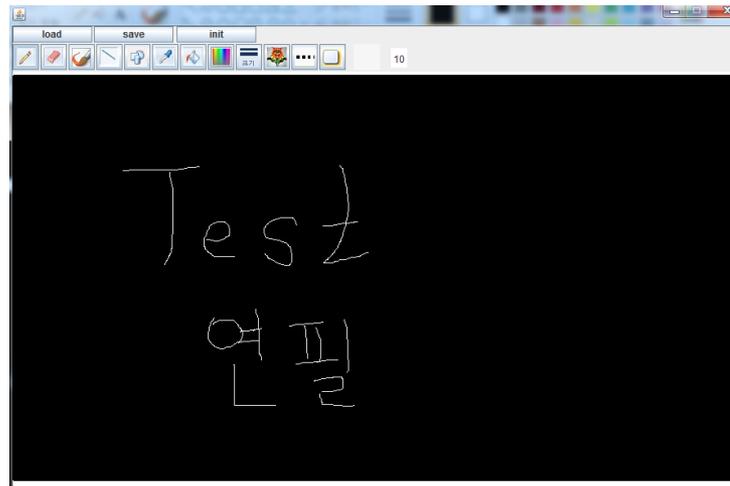
▪ 카테고리 항목 (input)

- 그리기 도구 : 연필
- 색 편집 : RGB(246, 246, 246)
- 선 두께 : 10

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 연필의 형태로 백색의 선 두께 10짜리 그림이 이상 없이 그려져야 한다.

- 테스트 결과(Fail) - 두께를 선택해도 적용되지 않고, 무조건 같은 두께로 나온다.



Test Case No32. (combinational test case)

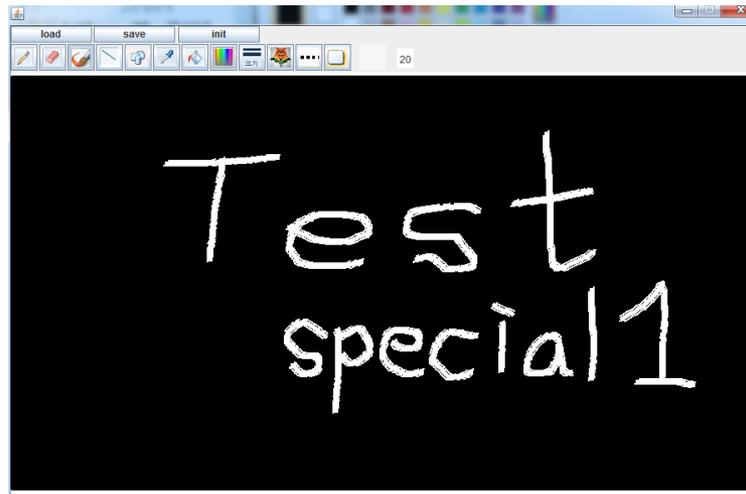
카테고리 항목 (input)

- 그리기 도구 : 브러시(special1)
- 색 편집 : RGB(246, 246, 246)
- 선 두께 : 20

예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 브러시-special1 의 형태로 백색의 선 두께 20짜리 그림이 이상 없이 그려져야 한다.

- 테스트 결과(Fail) - 두께를 선택해도 적용되지 않고, 무조건 같은 두께로 나온다.



Test Case No38. (combinational test case)

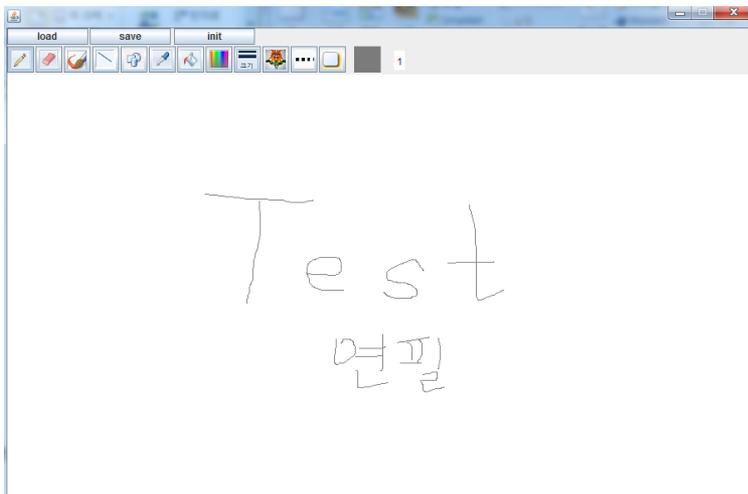
▪ 카테고리 항목 (input)

- 그리기 도구 : 연필
- 색 편집 : RGB(123, 123, 123)
- 선 두께 : 1

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 연필의 형태로 회색의 선 두께 1짜리 그림이 이상 없이 그려져야 한다.

▪ 테스트 결과(Pass) – 연필에서 두께 1일 경우만 pass



Test Case No50. (combinational test case)

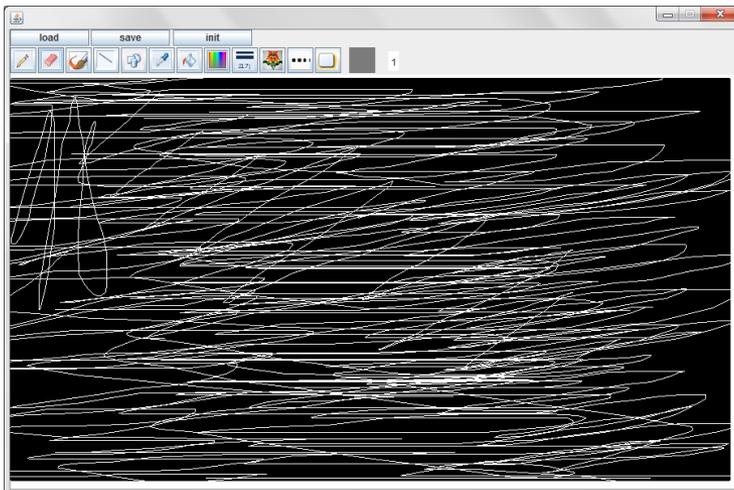
▪ 카테고리 항목 (input)

- 그리기 도구 : 지우개
- 색 편집 : 없음 (RGB : 123, 123, 123)
- 선 두께 : 1

▪ 예상되는 결과 (output)

- 선 두께 1만큼의 영역이 사용자가 드래그하는 대로 지워져야 한다.

▪ 테스트 결과(Pass)



Test Case No51. (combinational test case)

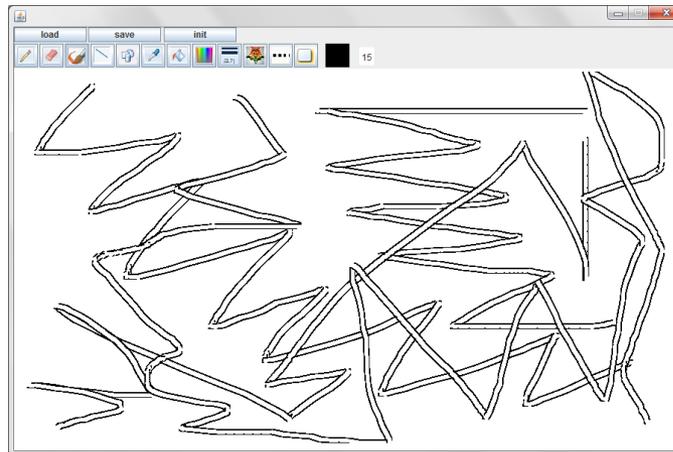
▪ 카테고리 항목 (input)

- 그리기 도구 : 브러시(special1)
- 색 편집 : RGB(0, 0, 0)
- 선 두께 : 15

▪ 예상되는 결과 (output)

- Requirement specification에 명시된 내용처럼 브러시-special1의 형태로 검정색의 선 두께 15짜리 그림이 이상 없이 그려져야 한다.

- 테스트 결과(Fail) - 두께를 선택해도 적용되지 않고, 무조건 같은 두께로 나온다.



JDepend & Clover

JDepend

■ 한 개의 Package

- 패키지 간의 의존성을 확인할 수 없었음.

The screenshot displays the JDepend Eclipse SDK interface. The left sidebar shows a tree view of the project structure under 'default', listing various Java files like DrawEllipse.java, DrawingObject.java, DrawLine.java, DrawRectangle.java, DrawShape.java, DrawTool.java, FileIO.java, Screen.java, SpecialEffect.java, and SpecialEffectTest.java.

The main area shows the 'Dependencies' view for the selected object(s). It contains four tables:

- Selected object(s):** Shows dependencies for the 'Default', 'junit.framework', and 'org.junit' packages.
- Packages with cycle:** An empty table.
- Depends upon - efferent dependencies:** Shows dependencies from 'junit.framework' and 'org.junit' to other packages.
- Used by - afferent dependencies:** Shows that the 'Default' package is used by other packages.

The 'Metrics' view at the bottom left shows a graph with 'Instability ->' on the y-axis and 'Abstractness' on the x-axis.

Package	CC(conc.cl.)	AC(abstr.cl.)	Ca(aff.)	Ce(eff.)	A	I	D	Cycle!
Default	43	1	0	2	0.02	1.00	0.02	
junit.framework	0	0	1	0	0.00	0.00	1.00	
org.junit	0	0	1	0	0.00	0.00	1.00	

Package	CC(conc...	AC(abstr...	Ca(aff.)	Ce(eff.)	A	I	D	Cycle!
junit.framework	0	0	1	0	0.00	0.00	1.00	
org.junit	0	0	1	0	0.00	0.00	1.00	

Package	CC(conc...	AC(abstr...	Ca(aff.)	Ce(eff.)	A	I	D	Cycle!
Default	43	1	0	2	0.02	1.00	0.02	

Clover

- 총 54개의 Test case들을 모두 실행해보고 난 후 Code coverage 확인

The screenshot displays the Eclipse IDE interface with the Clover Dashboard and Coverage Explorer views. The Coverage Explorer shows a table of code coverage metrics for the 'modling' project and its sub-packages and classes. The 'default package' is highlighted, showing an overall coverage of 86.1%.

Elem	Cov%	Av Me	Cpx	Cpx
modling	86.1%	2.1	370.0	
(default package)	86.1%	2.1	370.0	
DrawShape.java	0.0%	-	0.0	
SpecialEffect.java	73.4%	8.0	88.0	
DrawLine.java	78.6%	1.3	18.0	
DrawRectangle.java	78.6%	1.3	18.0	
DrawEllipse.java	78.6%	1.3	18.0	
DrawEllipse (Methods)				
setshadow()	0.0%	-	1.0	
setdecal()	0.0%	-	1.0	
DrawEllipse(double, double, douk)	58.3%	-	3.0	
DrawEllipse(double, double, douk)	76.9%	-	3.0	
getshadow()	100.0%	-	1.0	
getdecal()	100.0%	-	1.0	
DrawEllipse(double, double, douk)	100.0%	-	1.0	
getcolor()	100.0%	-	1.0	
getthickness()	100.0%	-	1.0	
getsx()	100.0%	-	1.0	
getsy()	100.0%	-	1.0	
getex()	100.0%	-	1.0	
getey()	100.0%	-	1.0	
getstate()	100.0%	-	1.0	
FileIO.java	86.4%	3.0	12.0	
DrawingObject.java	87.5%	1.4	21.0	
DrawTool.java	88.7%	1.9	124.0	
Screen.java	98.2%	1.7	71.0	

Metrics for: (default package)

Structure	Test Executions
Packages: -	Executed Tests: 0
Files: 9	Passes: 0
Classes: 10	Fails: 0
Methods: 179	Errors: 0
Statements: 1,026	
Branches: 290	

Source

LOC:	2,200	NC LOC:
Total Cmp:	370	Cmp Density:
Avg Method Cmp:	2.1	

Clover

- 총 54개의 Test case들을 모두 실행해보고 난 후 Code coverage 확인
 - Interface Class의 경우 회색
 - 선정된 54개의 Test case별로 모든 기능들을 수행하여 테스트해보았지만 100%의 Code coverage를 달성할 수 없었음.
 - 수행될 수 없는 예외처리 사항들에 대한 코드들이 포함되었을 것이라고 예상.
 - 시스템 테스트만을 수행한 것이므로 코드에 대한 내용을 분석하지 못해서 구체적으로 어떤 기능에 대한 부분이 cover되지 않았는지 알 수가 없었음.

```
626 public double getsx(){ return sx; }
280 public double getsy(){ return sy; }
453 public double getex(){ return ex; }
280 public double getey(){ return ey; }

0 public void setisex(){
0     this.isex = 1;
0 }
0 public int getisex() {
0     return isex;
0 }

7573 public double getx(int index){
7573     return x[index];
7573 }
6874 public double gety(int index){
6874     return y[index];
```

Additional Errors

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 1

▪ 시나리오

- 이미지를 불러오고 난 후에 데칼코마니 적용.
- 이미지를 불러오고 추가적인 작업 후 데칼코마니 적용.

▪ 예상되는 결과

- 불러온 이미지를 포함해서 데칼코마니 효과가 적용되던지, 아니면 사용자가 작업한 결과에 대해서만 데칼코마니 효과가 적용되어야 한다.

▪ 테스트 결과

- 일단 이미지를 불러온 이후에는 데칼코마니가 적용되지 않는다.
- 초기화를 해도 복구되지 않는다.
- 프로그램을 재시작 해야만 해결된다.

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 2

▪ 시나리오

- 색 채우기 적용 후에 데칼코마니 적용.
- 색 채우기 적용 후 추가적인 작업을 마치고 데칼코마니 적용.

▪ 예상되는 결과

- 데칼코마니 효과가 정상적으로 적용되어야 한다.

▪ 테스트 결과

- 일단 색 채우기를 적용한 후에는 데칼코마니가 적용되지 않는다.
- 초기화를 해도 복구되지 않는다.
- 프로그램을 재시작 해야만 해결된다.

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 3

■ 시나리오

- 그림이 있는 경우와 없는 경우로 나눠 데칼코마니를 반복 적용

■ 예상되는 결과

- 데칼코마니 효과가 정상적으로 적용되어야 한다.

■ 테스트 결과

- 그림이 없는 경우에는 문제가 없으나, 그림이 선 하나라도 있는 경우에 10회~15회 데칼코마니 반복 실행 시 에러 발생
- 데칼코마니를 제외한 다른 기능과 그리기 도구는 일시적으로 문제없이 작동하지만 작업을 지속하면 모든 기능이 적용이 안되고 종료 또한 안 되는 에러 발생 => 외부에서 강제 종료 필요.

```
Problems @ Javadoc Declaration Console Properties LogCat
Screen [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (2013. 5. 23. 오후 9:50:12)
Exception in thread "AWT-EventQueue-0" java.lang.OutOfMemoryError: Java heap space
    at DrawingObject.<init>(DrawingObject.java:32)
    at SpecialEffect.excutedecal(SpecialEffect.java:119)
    at Screen.rdecal(Screen.java:477)
    at Screen$10.actionPerformed(Screen.java:244)
    at javax.swing.AbstractButton.fireActionPerformed(Unknown Source)
    at javax.swing.AbstractButton$Handler.actionPerformed(Unknown Source)
```

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 4

▪ 시나리오

- 그림자 효과를 적용한 이후 추가로 또 다른 그림자 효과를 준다.

▪ 예상되는 결과

- 기 적용된 그림에 또 다른 그림자 효과가 추가로 적용되던지, 아니면 추가로 적용한 그림자 효과로 바뀌어야 한다.

▪ 테스트 결과

- 기 적용된 그림에는 추가적인 그림자 효과 적용에 대한 결과가 나타나지 않는다.

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 5

▪ 시나리오

- 텍스트 입력 / 영역 선택 / 복사 / 잘라내기 / 붙여넣기 기능 실행

▪ 예상되는 결과

- Requirement Specification에 명시된 내용처럼 텍스트 입력 / 영역 선택 / 복사 / 잘라내기 / 붙여넣기 기능이 이상 없이 실행되어야 한다.

▪ 테스트 결과

- 구현되지 않았음.

Combinatorial Testing의 결과가 아닌 추가적으로 발견한 Error 6

▪ 시나리오

- 펜이나 브러시로 작업 후 이미지를 불러온다.
- 이후 점선으로 변경을 누른다.

▪ 예상되는 결과

- 이미지를 불러온 이후의 작업내용에만 점선으로 변경이 적용된다.

▪ 테스트 결과

- 점선으로 변경을 누르면, 이미지를 불러오기 이전 작업내용이 점선으로 변경된 상태로 다시 나타난다.