prepare for….

Mutation Testing

..from jUnit to Mutation-Testing

@SvenRuppert

has been coding java since 1996

Developer Advocate

Vaadin

Germany - Munich

@SvenRuppert

has been coding java since 1996

@SvenRuppert

has been coding java since 1996

Projects in the field of:

- Automobile-industry
- Energy
- Finance / Leasing
- Space- Satellit-
- Government / UN / World-bank

Where?

- Europe
- Asia - from India up to Malaysia

Save harbor statement

Save harbor statement

The following is intended for information purposes only. I can not be held responsible for the overuse of effects and animations in this presentation. If any person in this room has a medical condition that is triggered by fast moving objects on the screen and/or explosions, he/she should probably better leave now…

(I got carried away by the topic.)

The Environment

Do you have bugs in your code ?

The Environment

Do you have bugs in your code ?

no

The Environment

Do you have bugs in your code ?

no

since years
you are
working hard
on this.....

The Environment

Do you have bugs in your code ?

no

since years
you are
working hard
on this.....

The Environment

Codebase is > 13 years old

The Environment

Codebase is > 13 years old

- no test coverage

The Environment

Codebase is > 13 years old

- no test coverage
  - no dedicated refactoring budget

The Environment

- Codebase is > 13 years old
  - no test coverage
    - no dedicated refactoring budget
      - decrease complexity

The Environment

Codebase is > 13 years old

no test coverage

no dedicated refactoring budget

decrease complexity

but… lets start with the basics

TDD with jUnit

@SvenRuppert

TDD with jUnit

are you using jUnit?

TDD with jUnit

are you using jUnit?

- assume that the following would make sense.. ;-)

TDD with jUnit

are you using jUnit?

assume that the following would make sense.. ;-)

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

TDD with jUnit

are you using jUnit?

assume that the following would make sense.. ;-)

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

TDD with jUnit

are you using jUnit?

assume that the following would make sense.. ;-)

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

TDD with jUnit

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

TDD with jUnit

@SvenRuppert

```
public class Service {
 public int add(int a, int b){
  if(a<2){
   return (a+b) * -1;
  } else {
   return a+b;
  }
 }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage

TDD with jUnit

@SvenRuppert

```
public class Service {
 public int add(int a, int b){
  if(a<2){
   return (a+b) * -1;
  } else {
   return a+b;
  }
 }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage 2

TDD with jUnit

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage 2

but will this be enough?

TDD with jUnit

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage 2

but will this be enough? **maybe ;-)**

TDD with jUnit

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage   2

but will this be enough? **maybe ;-)**

TDD with jUnit

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage 2

but will this be enough? **maybe ;-)**

how to find out, what will be enough?

TDD with jUnit

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests you will need ?

it depends ;-)

for line 100% coverage   2

but will this be enough? **maybe ;-)**

how to find out, what will be enough?

how to find the right tests?

TDD with jUnit

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests
you will need ?

TDD with jUnit

@SvenRuppert

How many tests you will need ?

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

```
@Test
public void testAdd001() throws Exception {
  final int add = new Service().add(0, 0);
  Assertions.assertThat(add).isEqualTo(0);
}
```

TDD with jUnit

```
public class Service {
  public int add(int a, int b){
    if(a<2){
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

How many tests
you will need ?

```
@Test
public void testAdd001() throws Exception {
  final int add = new Service().add(0, 0);
  Assertions.assertThat(add).isEqualTo(0);
}

@Test
public void testAdd002() throws Exception {
  final int add = new Service().add(3, 0);
  Assertions.assertThat(add).isEqualTo(3);
}
```

Mutation Testing

Mutation Testing

Mutation Testing is a structural testing method

Mutation Testing

Mutation Testing is a structural testing method

- we want to find a way to write "good" tests

Mutation Testing

- Mutation Testing is a structural testing method
  - we want to find a way to write "good" tests
    - how to find "good" tests?

Mutation Testing

- Mutation Testing is a structural testing method
  - we want to find a way to write "good" tests
    - how to find "good" tests?
      - let the machine find the targets

Mutation Testing

Mutation Testing is a structural testing method

we want to find a way to write "good" tests

how to find "good" tests?

let the machine find the targets

let´s mutate it... but how?

Mutation Testing - the Idea

@SvenRuppert

Mutation Testing - the Idea

a mutation is a small change in the code

Mutation Testing - the Idea

a mutation is a small change in the code

.. small enough to be a small defect

Mutation Testing - the Idea

a mutation is a small change in the code

.. small enough to be a small defect

P will be the program

Mutation Testing - the Idea

a mutation is a small change in the code

.. small enough to be a small defect

P will be the program

T will be the collection of all tests / Test Suite

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

.. **Px** will have only one mutation compared to **P**

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

.. **Px** will have only one mutation compared to **P**

running all tests from **T** against **Px**

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

.. **Px** will have only one mutation compared to **P**

running all tests from **T** against **Px**

green: **T** will kill the mutation

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

.. **Px** will have only one mutation compared to **P**

running all tests from **T** against **Px**

green: **T** will kill the mutation

.. at leased one test from **T** will fail

Mutation Testing - the Idea

P will be the program

T will be the collection of all tests / Test Suite

we will create a sequence of mutations / **P1**,**P2**,**P3**...

.. **Px** will have only one mutation compared to **P**

running all tests from **T** against **Px**

green: **T** will kill the mutation

.. at leased one test from **T** will fail

red: if all tests are green

Mutation Testing - the Idea

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

-> we are not good enough ;-)

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

-> we are not good enough ;-)

we are **perfect** enough if we are reaching : **k == n**

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

-> we are not good enough ;-)

we are **perfect** enough if we are reaching : **k == n**

how to create all versions of Px ?

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

-> we are not good enough ;-)

we are **perfect** enough if we are reaching : **k == n**

how to create all versions of Px ?

.. the good thing..

Mutation Testing - the Idea

if we kill **k** out of **n** mutants

-> we are not good enough ;-)

we are **perfect** enough if we are reaching : **k == n**

how to create all versions of Px ?

.. the good thing..

we could almost generate/
automate everything

Mutation Testing

practical TDD with Mutation Testing

Mutation Testing

practical TDD with Mutation Testing

generating the mutants and

Mutation Testing

practical TDD with Mutation Testing

generating the mutants and

running all junit tests

Mutation Testing

practical TDD with Mutation Testing

generating the mutants and

running all junit tests

check the reports

Mutation Testing

practical TDD with Mutation Testing

generating the mutants and

running all junit tests

check the reports

write more / better tests

Mutation Testing

practical TDD with Mutation Testing

generating the mutants and

running all junit tests

check the reports

write more / better tests

loop until quality target reached

Mutation Testing

mutants are a good approach / model to estimate the default rate of defects per 1k lines of the **P**

Mutation Testing

mutants are a good approach / model to estimate the default rate of defects per 1k lines of the **P**

estimate that:

Mutation Testing

mutants are a good approach / model to estimate the default rate of defects per 1k lines of the **P**

estimate that:

the defects are independent

Mutation Testing

Mutation Testing

no need to know that Mutation Testing will be done, independent creation of **T**

Mutation Testing

no need to know that Mutation Testing will be done,
independent creation of **T**

for **K==n**
we need a high number of mutants (**P1**, **P2**, .., **Px**)

Mutation Testing

no need to know that Mutation Testing will be done,
independent creation of **T**

for **K==n**
we need a high number of mutants (**P1**, **P2**, .., **Px**)

.. mostly it will lead into exponential numbers of **Px**

Mutation Testing

no need to know that Mutation Testing will be done, independent creation of **T**

for **K==n**
we need a high number of mutants (**P1**, **P2**, .., **Px**)

.. mostly it will lead into exponential numbers of **Px**

.. how to find YOUR barrier you have to reach?

Mutation Testing

no need to know that Mutation Testing will be done, independent creation of **T**

for **K==n**
we need a high number of mutants (**P1**, **P2**, .., **Px**)

.. mostly it will lead into exponential numbers of **Px**

.. how to find YOUR barrier you have to reach?

but.. what is a mutation?

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation

changing constants,
loop bounds (adding/subtracting values)

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation

Decision Mutation

for example **<** will be changed to **<=**

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation **Decision Mutation**

Statement Mutation

for example swapping/deleting/duplicating lines of code

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

for **Java** you could think about more language spec. mutations

Mutation Testing - Kinds of Mutation

@SvenRuppert

but.. what is a mutation?

Value Mutation **Decision Mutation**
Statement Mutation

for **Java** you could think about more language spec. mutations

.. changing modifiers

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

for **Java** you could think about more language spec. mutations

.. changing modifiers

.. changing between static / non static

Mutation Testing - Kinds of Mutation

@SvenRuppert

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

for **Java** you could think about more language spec. mutations

.. changing modifiers

.. changing between static / non static

.. delete member initialization

Mutation Testing - Kinds of Mutation

@SvenRuppert

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

for **Java** you could think about more language spec. mutations

.. changing modifiers

.. changing between static / non static

.. delete member initialization

.. delete this.

Mutation Testing - Kinds of Mutation

but.. what is a mutation?

Value Mutation Decision Mutation

Statement Mutation

for **Java** you could think about more language spec. mutations

.. changing modifiers

.. changing between static / non static

.. delete member initialization

.. delete this.

.. argument order change

Mutation Testing - in short words

@SvenRuppert

Mutation Testing - in short words

mutation testing is an add on to normal jUnit TDD

Mutation Testing - in short words

mutation testing is an add on to normal jUnit TDD

tools are supporting it well

Mutation Testing - in short words

mutation testing is an add on to normal jUnit TDD

tools are supporting it well

generating and running all tests are time consuming

Mutation Testing - in short words

@SvenRuppert

mutation testing is an add on to normal jUnit TDD

tools are supporting it well

generating and running all tests are time consuming

but most important

Mutation Testing - in short words

mutation testing is an add on to normal jUnit TDD

tools are supporting it well

generating and running all tests are time consuming

but most important

will effect your project structure

Mutation Testing - Frameworks

Mutation Testing - Frameworks

muJava

Mutation Testing - Frameworks

muJava

2003. First released as JMutation (Java Mutation System).
2004. The name was changed to MuJava (Mutation System for Java).
2005. Software Copyright Registration, ALL RIGHTS RESERVED.
2005. Version 2 released with several fault fixes and modified mutation operators.
2008. Version 3 released with minimal support for Java 1.5 and 1.6.
2013. Version 4 released to support JUnit tests and Java 1.6 language features, including generics, annotations, enumerations, varargs, enhanced for-each loops, and static imports.
2015. Additional and improved error messages. Bug fixes for OpenJava. Licensing changed to the Apache license.

Mutation Testing - Frameworks

muJava

2003. First released as JMutation (Java Mutation System).
2004. The name was changed to MuJava (Mutation System for Java).
2005. Software Copyright Registration, ALL RIGHTS RESERVED.
2005. Version 2 released with several fault fixes and modified mutation operators.
2008. Version 3 released with minimal support for Java 1.5 and 1.6.
2013. Version 4 released to support JUnit tests and Java 1.6 language features, including generics, annotations, enumerations, varargs, enhanced for-each loops, and static imports.
2015. Additional and improved error messages. Bug fixes for OpenJava. Licensing changed to the Apache license.

https://cs.gmu.edu/~offutt/mujava/

https://github.com/jeffoffutt/muJava/graphs/contributors

Mutation Testing - Frameworks

@SvenRuppert

muJava

inactive

2003. First released as JMutation (Java Mutation System).
2004. The name was changed to MuJava (Mutation S
2005. Software Copyright Registration, ALL RI
2005. Version 2 released with several fault fix nutation operators.
2008. Version 3 released with minimal support .5 and 1.6.
2013. Version 4 released to support JUnit tests and Java 1.6 language features, including generics, annotations, enumerations, varargs, enhanced for-each loops, and static imports.
2015. Additional and improved error messages. Bug fixes for OpenJava. Licensing changed to the Apache license.

https://cs.gmu.edu/~offutt/mujava/

https://github.com/jeffoffutt/muJava/graphs/contributors

Mutation Testing - Frameworks

Mutation Testing - Frameworks

2012. started around 2012 with a small codebase.
2014. very active since 2014

Mutation Testing - Frameworks

http://pitest.org/

2012. started around 2012 with a small codebase.
2014. very active since 2014

Mutation Testing - Frameworks

http://pitest.org/

2012. started around 2012 with a small codebase.
2014. very active since 2014

active ;-)

Mutation Testing - Hello World

@SvenRuppert

http://pitest.org/

Mutation Testing - Hello World

http://pitest.org/

assume the following would make sense ;-)

Mutation Testing - Hello World

http://pitest.org/

assume the following would make sense ;-)

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

how many test you will need for..

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

how many test you will need for..

100% Line Coverage… and…

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

how many test you will need for..

100% Line Coverage… and… **to be save ?**

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

how many test you will need for..

100% Line Coverage… and…

to be save ?

2 for Line Coverage

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

how many test you will need for..

100% Line Coverage… and…

to be save ?

2 for Line Coverage

we will see ;-)

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

100% Line Coverage… and…

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

100% Line Coverage… and…

we have one if statement

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

100% Line Coverage… and…

we have one if statement   with an else branch

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) {
      return (a+b) * -1;
    } else {
      return a+b;
    }
  }
}
```

100% Line Coverage… and…

we have one if statement   with an else branch

this will lead to 2 jUnit Tests to get 100 %

Mutation Testing - Hello World

```
public class Service {
  public int add(int a, int b){
    if (a<2) { return (a+b) * -1; }
    else     { return a+b;        }
  }
}
```

100% Line Coverage… and…

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) { return (a+b) * -1; }
    else     { return a+b;        }
  }
}
```

100% Line Coverage… and…

```
@Test
public void testAdd001() throws Exception {
  final int add = new Service().add(0, 0);
  Assertions.assertThat(add).isEqualTo(0);
}
```

Mutation Testing - Hello World

@SvenRuppert

```
public class Service {
  public int add(int a, int b){
    if (a<2) { return (a+b) * -1; }
    else     { return a+b;        }
  }
}
```

100% Line Coverage… and…

```
@Test
public void testAdd001() throws Exception {
  final int add = new Service().add(0, 0);
  Assertions.assertThat(add).isEqualTo(0);
}
    @Test
    public void testAdd002() throws Exception {
      final int add = new Service().add(3, 0);
      Assertions.assertThat(add).isEqualTo(3);
    }
```

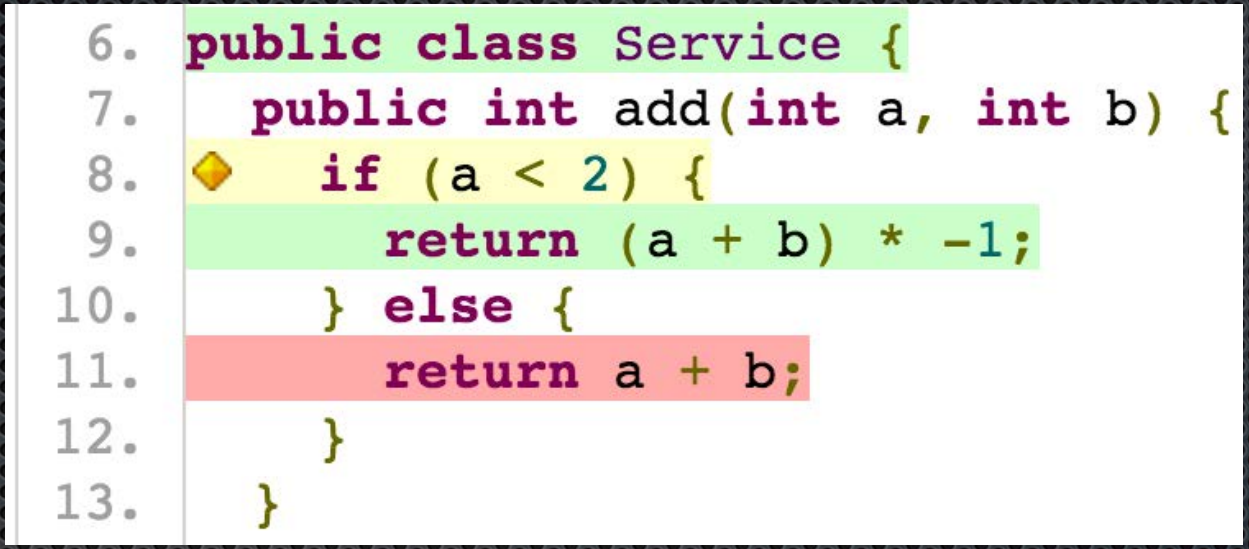
Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
```

```
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
```



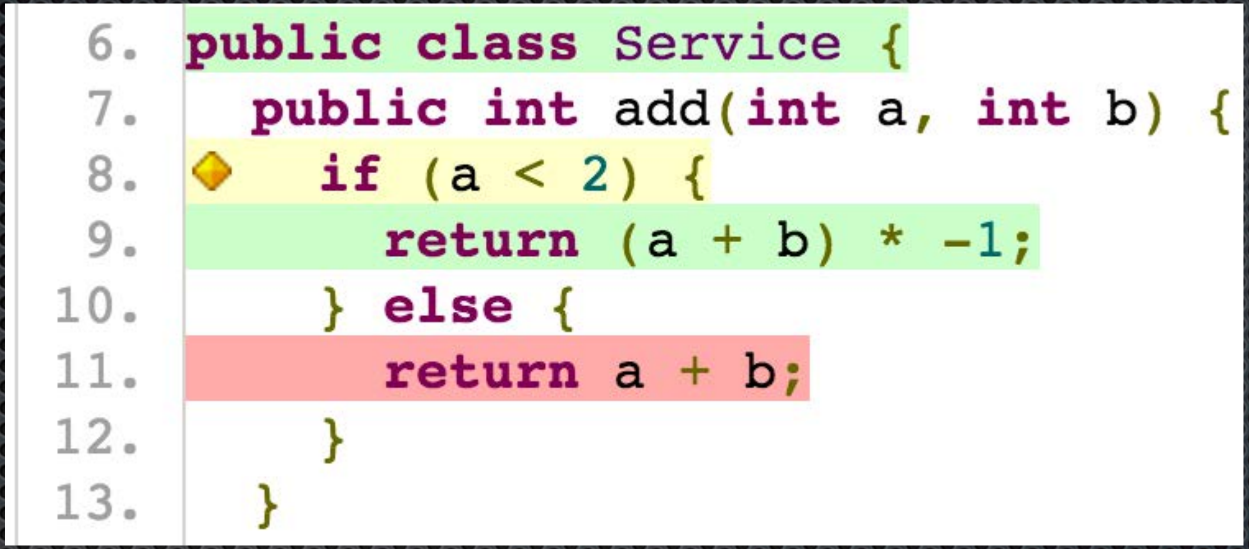
```
public class Service {
  public int add(int a, int b) {
    if (a < 2) {
      return (a + b) * -1;
    } else {
      return a + b;
    }
  }
```

```
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

Mutation Testing - Hello World

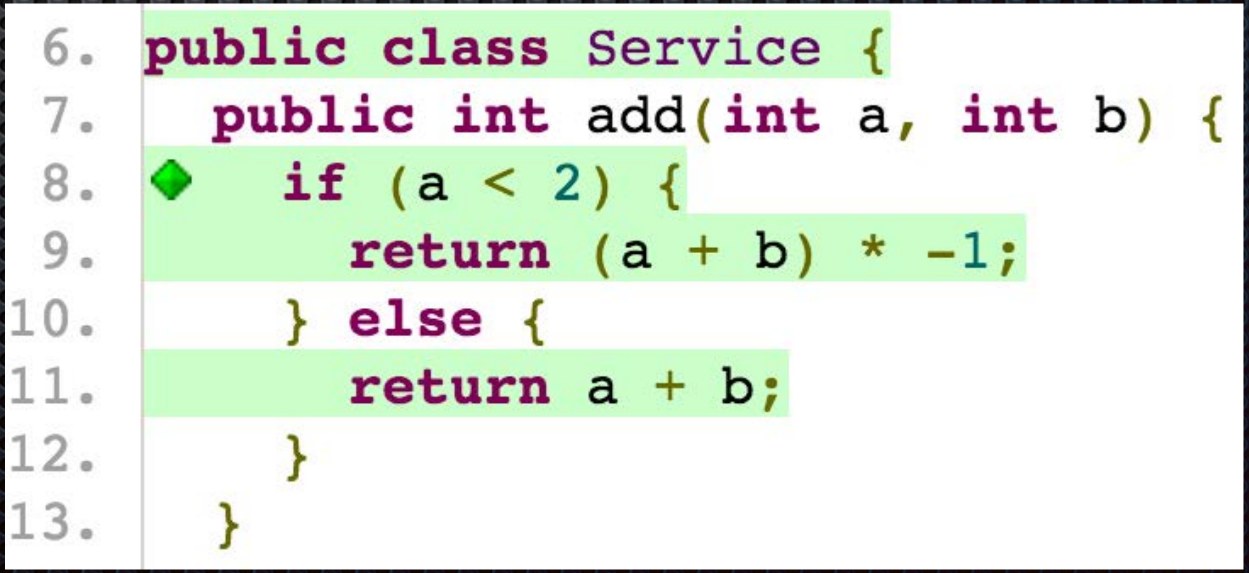
@SvenRuppert

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
```



```
public class Service {
  public int add(int a, int b) {
    if (a < 2) {
      return (a + b) * -1;
    } else {
      return a + b;
    }
  }
```

```
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```



```
public class Service {
  public int add(int a, int b) {
    if (a < 2) {
      return (a + b) * -1;
    } else {
      return a + b;
    }
  }
```

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

we got 100% Line Coverage

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

we got 100% Line Coverage

How good these tests are?

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

we got 100% Line Coverage

How good these tests are?

How to measure if these test are good?

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);
Assertions.assertThat(add).isEqualTo(0);
final int add = new Service().add(3, 0);
Assertions.assertThat(add).isEqualTo(3);
```

we got 100% Line Coverage

How good these tests are?

How to measure if these test are good?

How to find the good tests?

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

- let´s generate a the mutation report

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
```

```
final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

org.pitest……mutators.**ConditionalsBoundaryMutator**

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

org.pitest……mutators.**ConditionalsBoundaryMutator**
org.pitest……mutators.**IncrementsMutator**

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

org.pitest……mutators.**ConditionalsBoundaryMutator**
org.pitest……mutators.**IncrementsMutator**
org.pitest……mutators.**ReturnValsMutator**

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

org.pitest……mutators.**ConditionalsBoundaryMutator**
org.pitest……mutators.**IncrementsMutator**
org.pitest……mutators.**ReturnValsMutator**
org.pitest……mutators.**MathMutator**

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);

final int add = new Service().add(3, 0);
```

How to find the good tests?

let´s generate a the mutation report

with maven : **pitest: mutationCoverage**

>> Generated 54 mutations Killed 3

org.pitest……mutators.**ConditionalsBoundaryMutator**
org.pitest……mutators.**IncrementsMutator**
org.pitest……mutators.**ReturnValsMutator**
org.pitest……mutators.**MathMutator**
org.pitest……mutators.**NegateConditionalsMutator**

Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

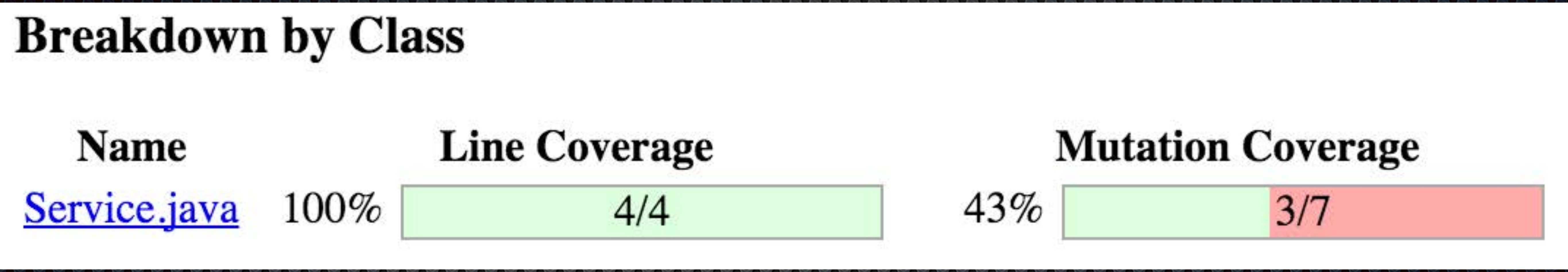
>> Generated 54 mutations   Killed 3

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations Killed 3



Breakdown by Class

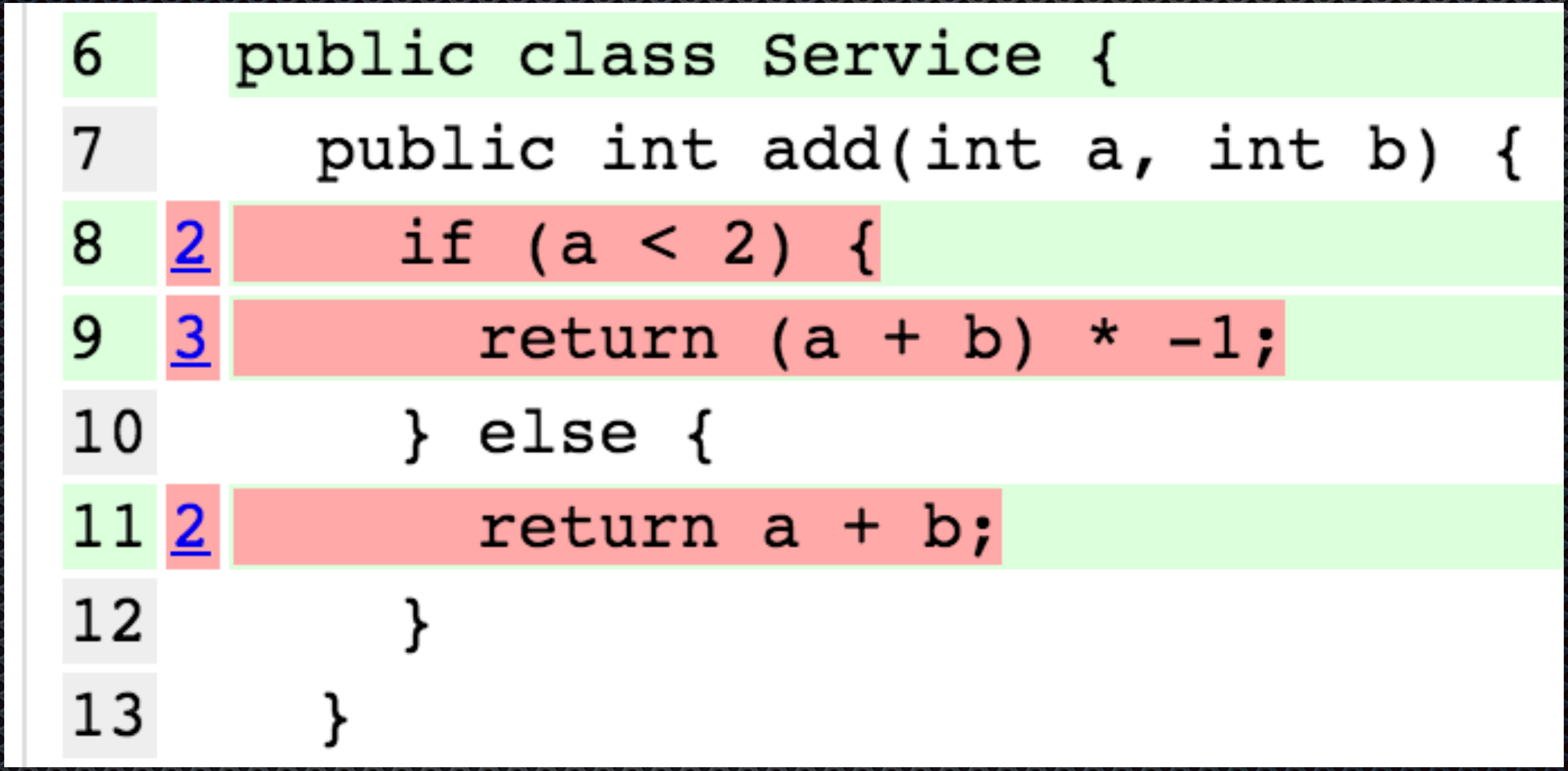
Name	Line Coverage		Mutation Coverage	
Service.java	100%	4/4	43%	3/7

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations   Killed 3



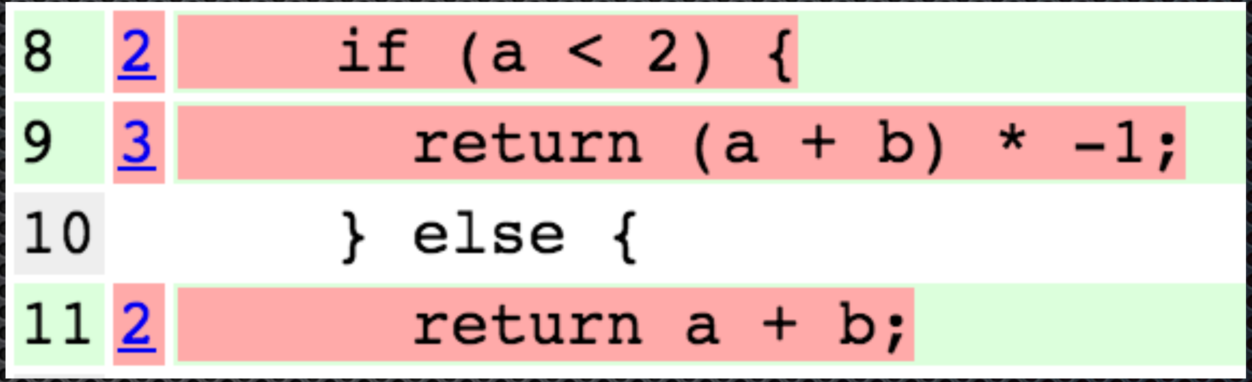
```
     public class Service {
       public int add(int a, int b) {
2        if (a < 2) {
3          return (a + b) * -1;
         } else {
2          return a + b;
         }
       }
```

Mutation Testing - Hello World

@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations Killed 3



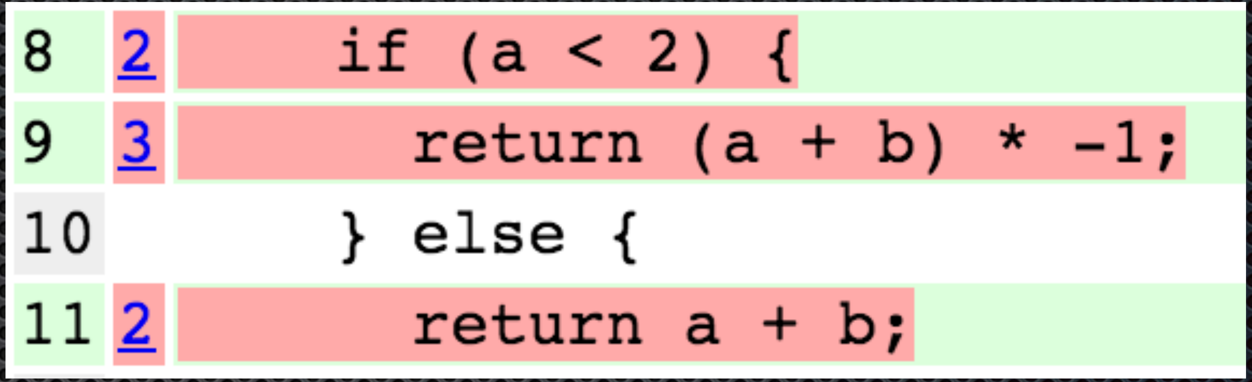
```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
```

Mutation Testing - Hello World

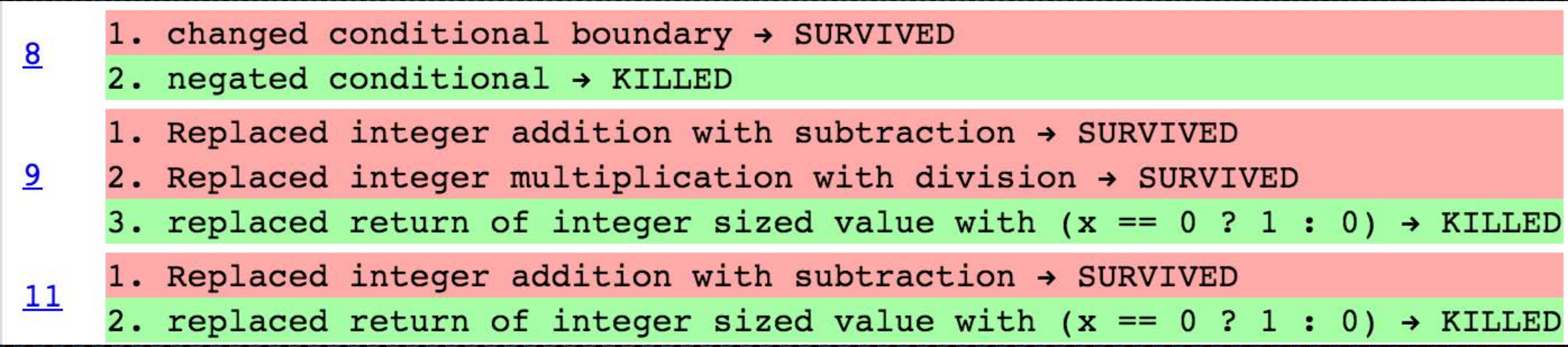
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations Killed 3



```
8  2        if (a < 2) {
9  3          return (a + b) * -1;
10          } else {
11 2          return a + b;
```



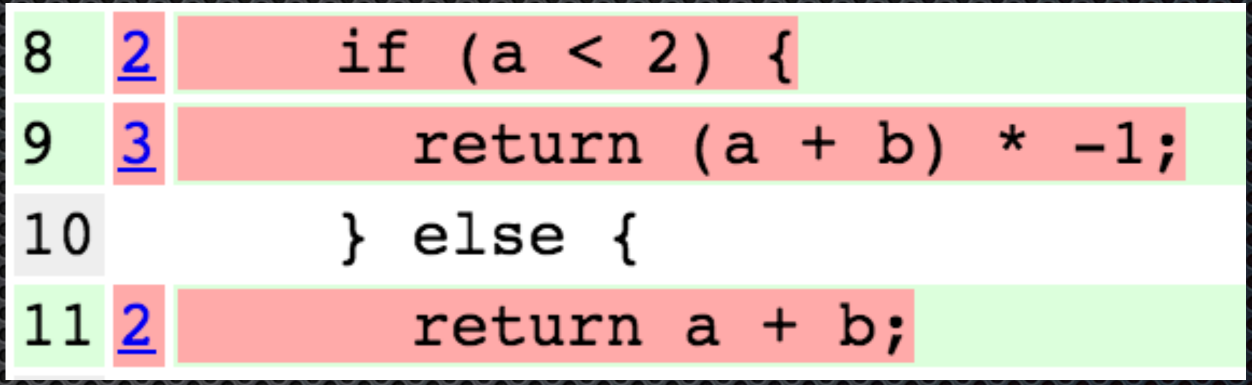
```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

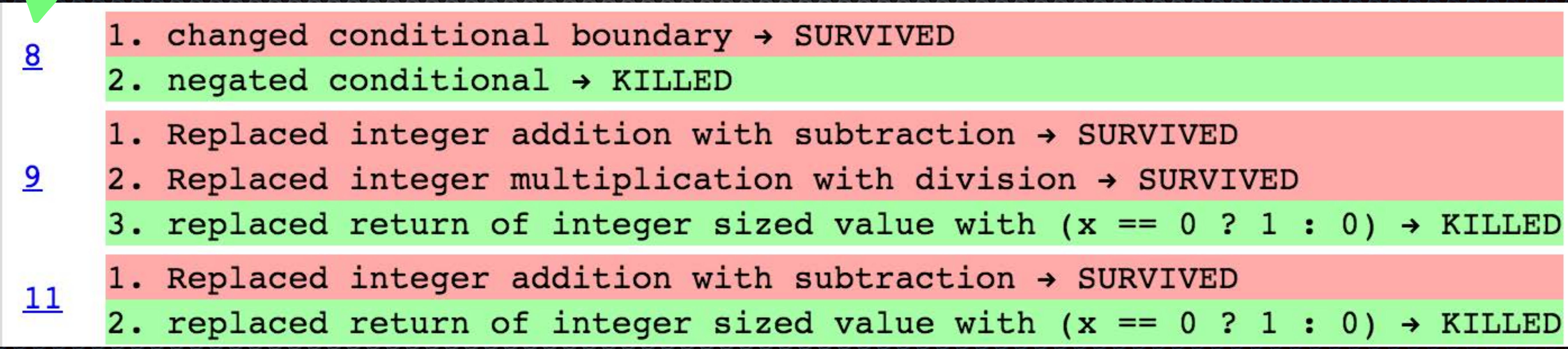
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations Killed 3



```
8  2     if (a < 2) {
9  3       return (a + b) * -1;
10       } else {
11 2       return a + b;
```



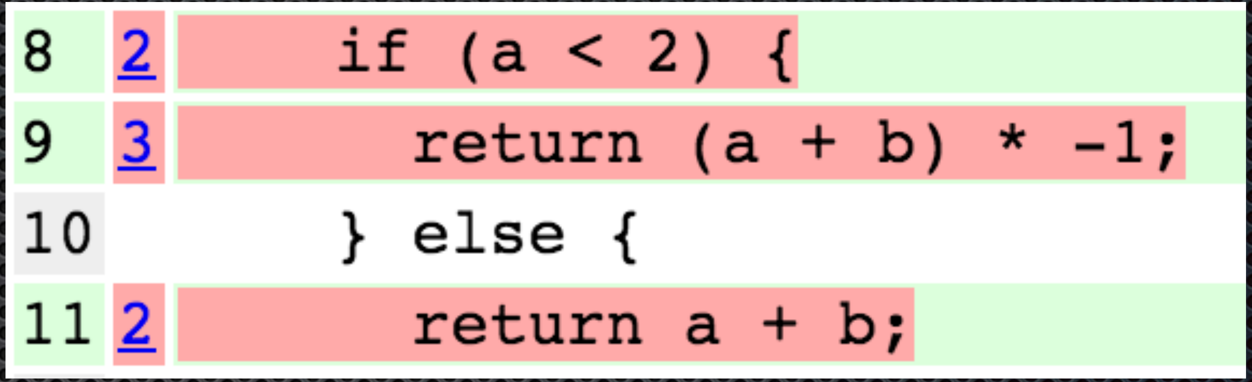
```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

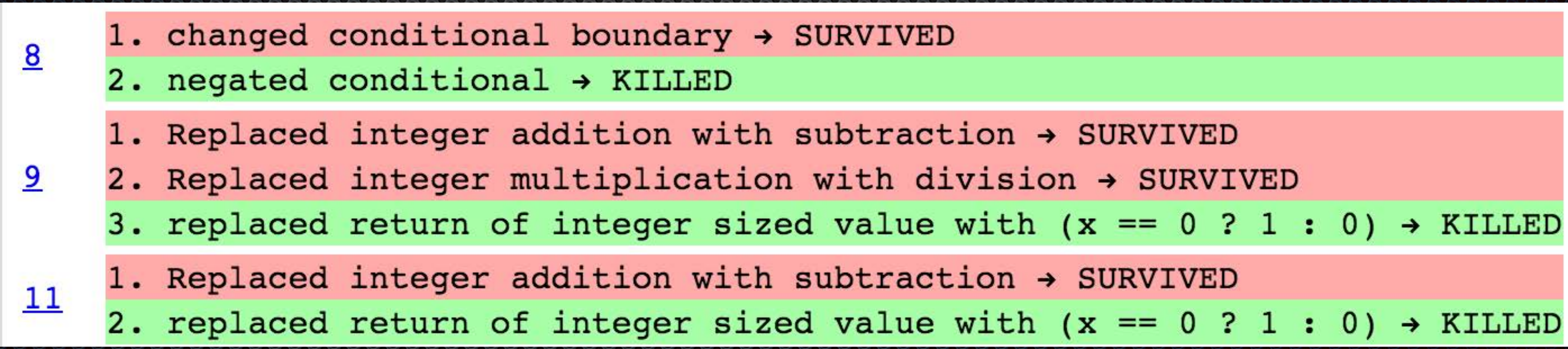
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
```

>> Generated 54 mutations Killed 3



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
```



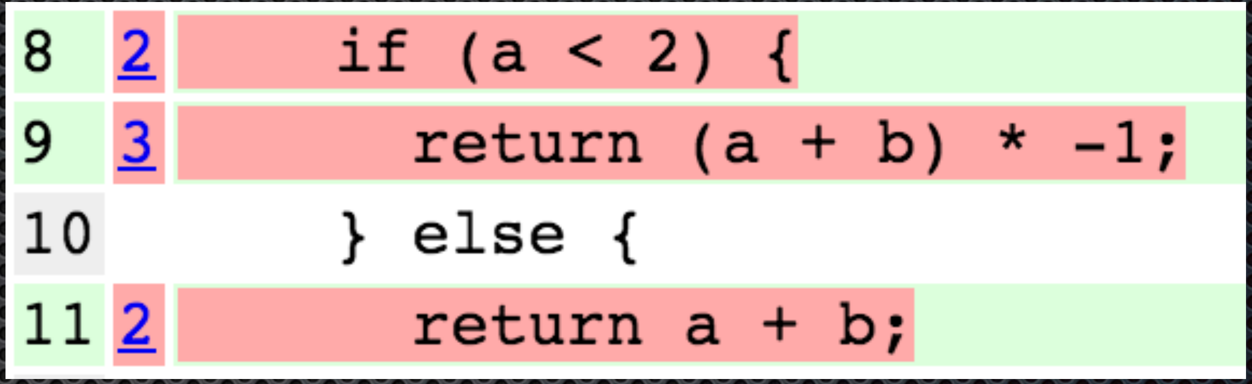
```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

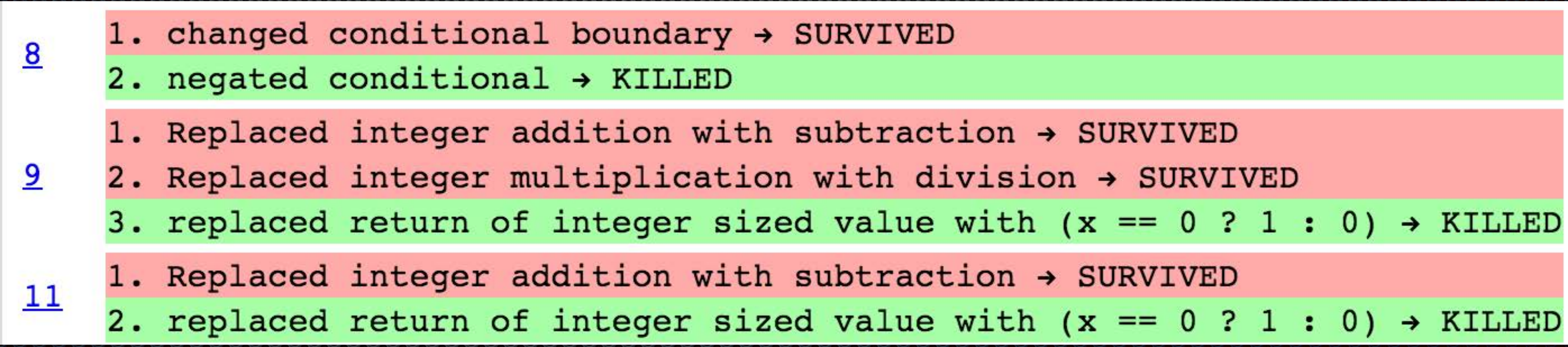
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations Killed 3



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
```



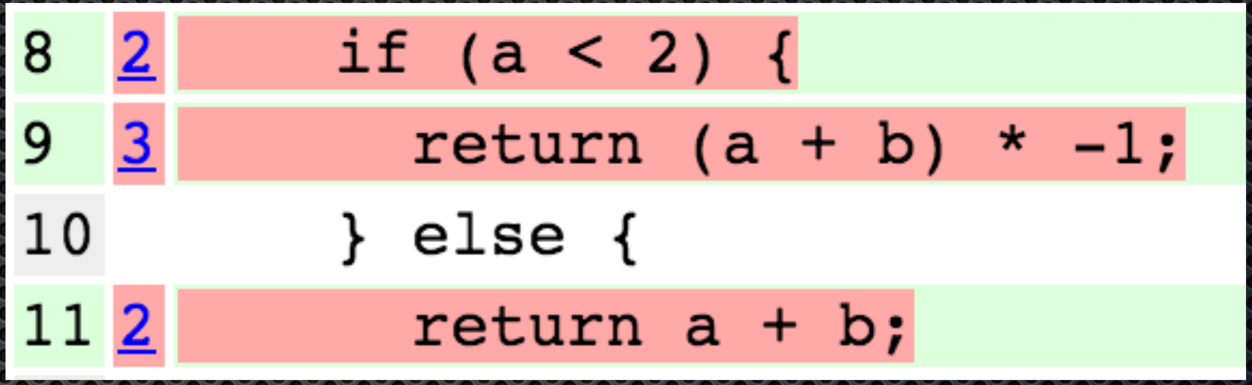
```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

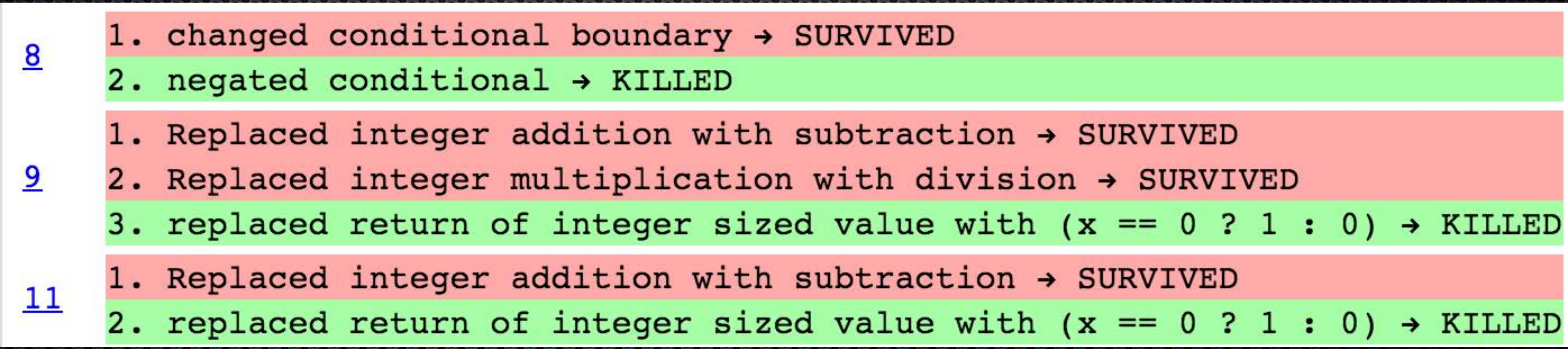
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
```

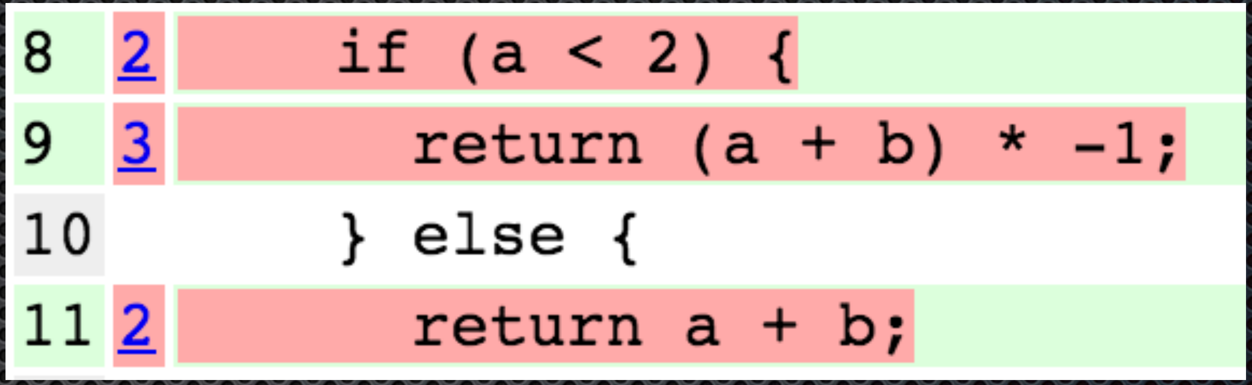


```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

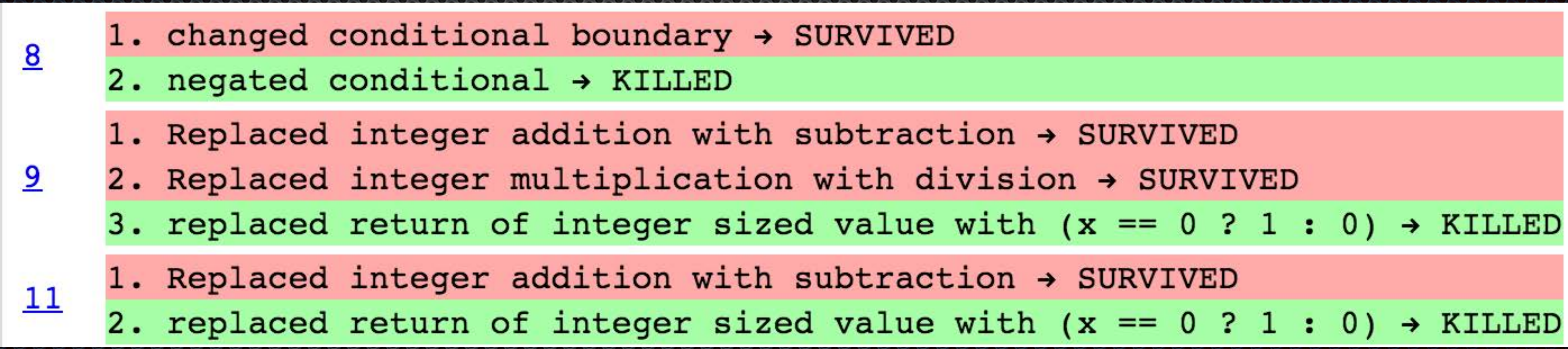
Mutation Testing - Hello World

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations   Killed 4



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
```



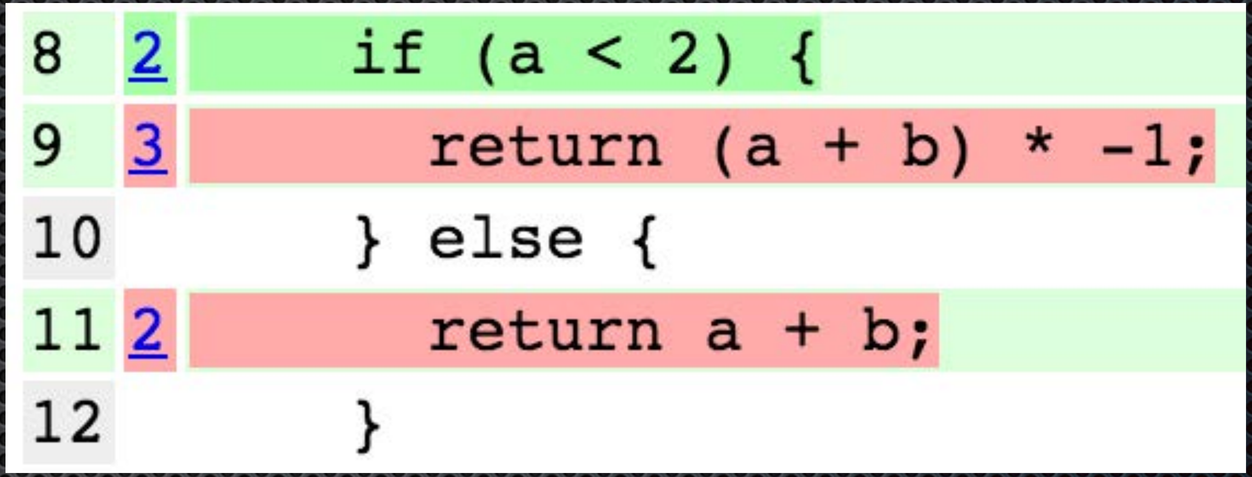
```
8   1. changed conditional boundary → SURVIVED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

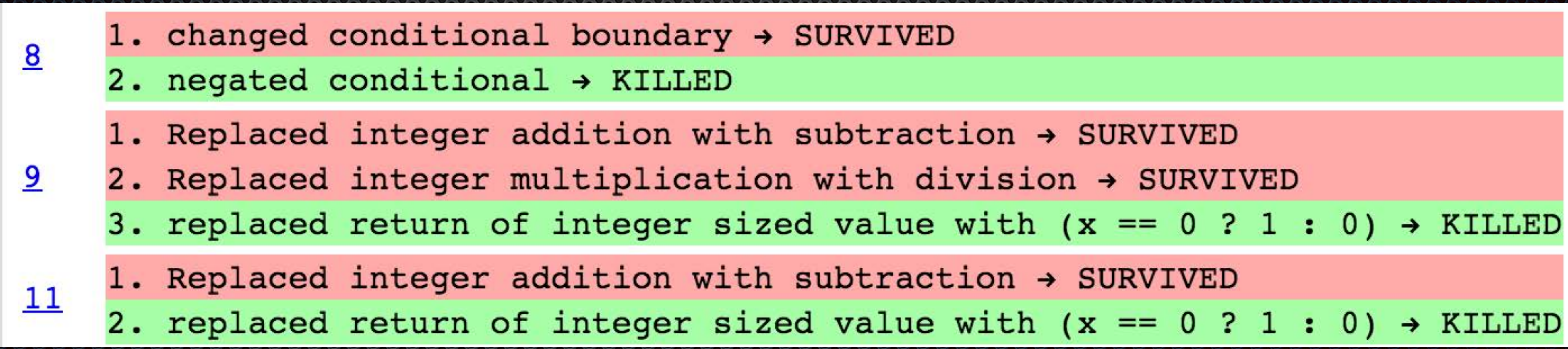
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations Killed 4



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



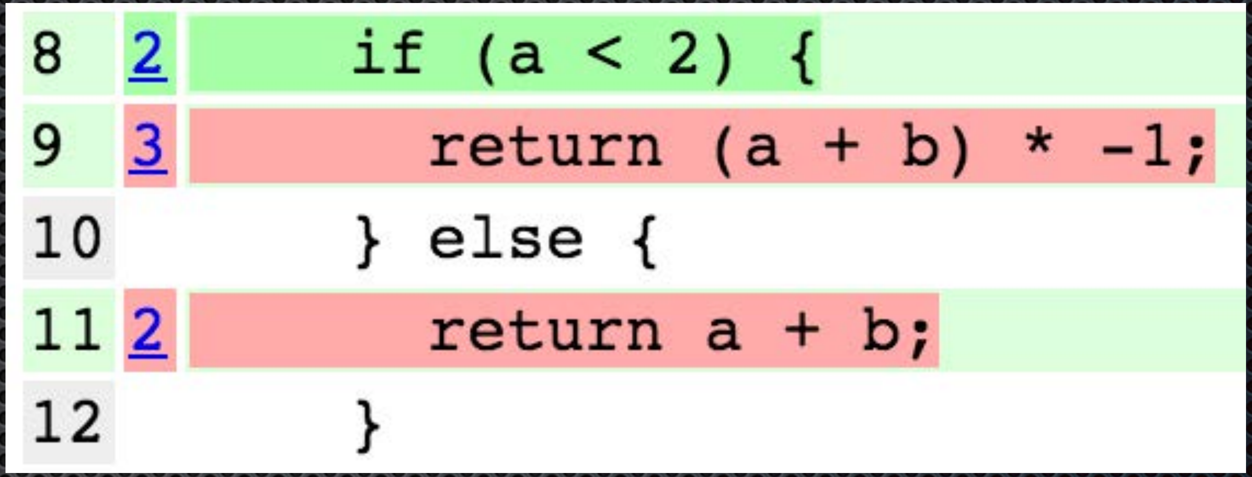
```
8    1. changed conditional boundary → SURVIVED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → SURVIVED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

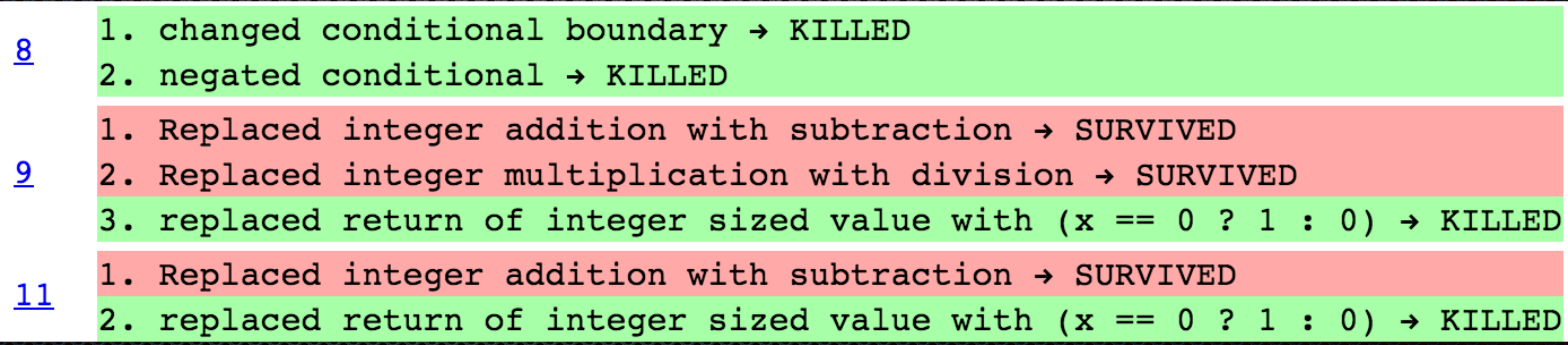
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations   Killed 4



```
8  2     if (a < 2) {
9  3       return (a + b) * -1;
10       } else {
11 2       return a + b;
12       }
```



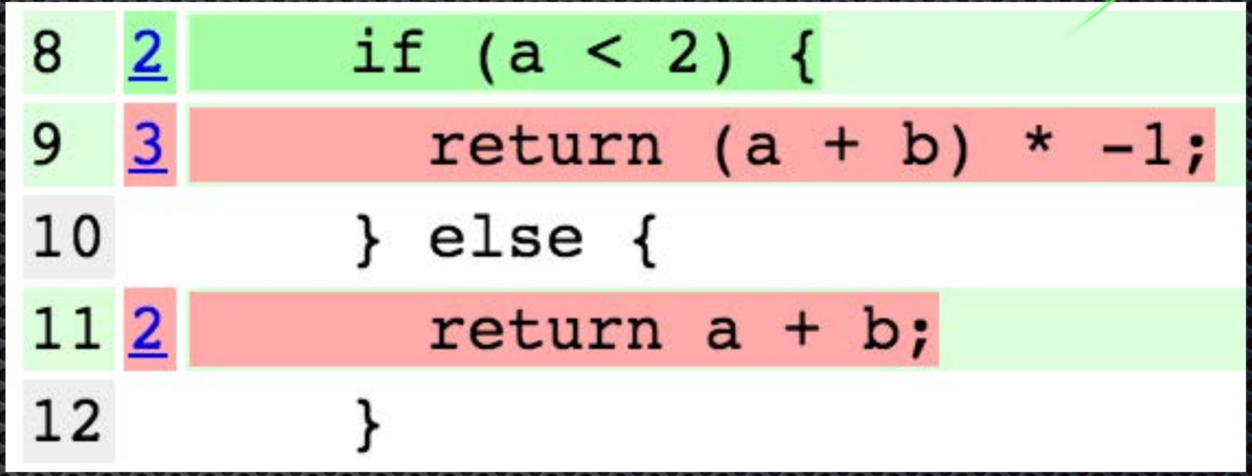
```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → SURVIVED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

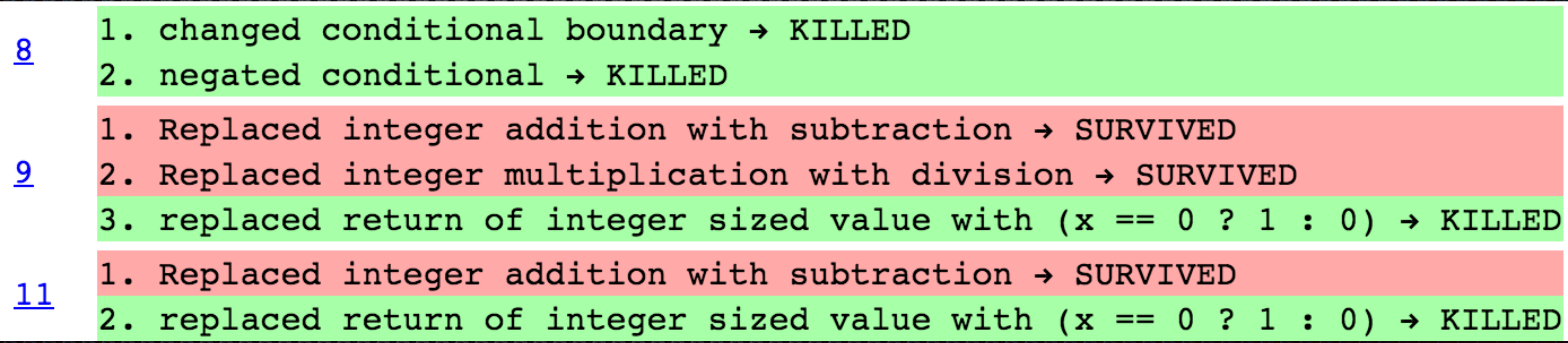
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(3, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations   Killed 4



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



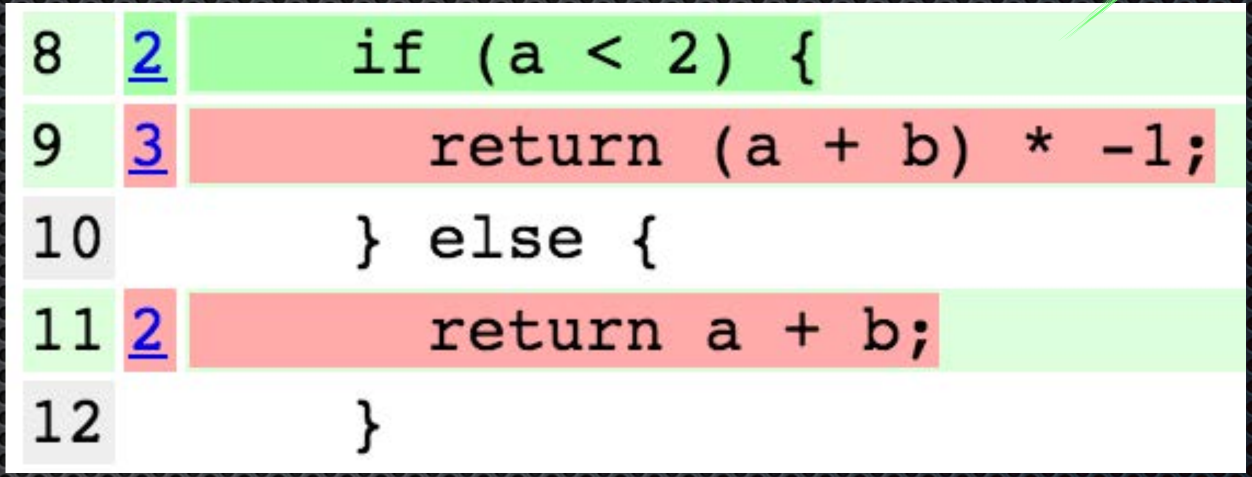
```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

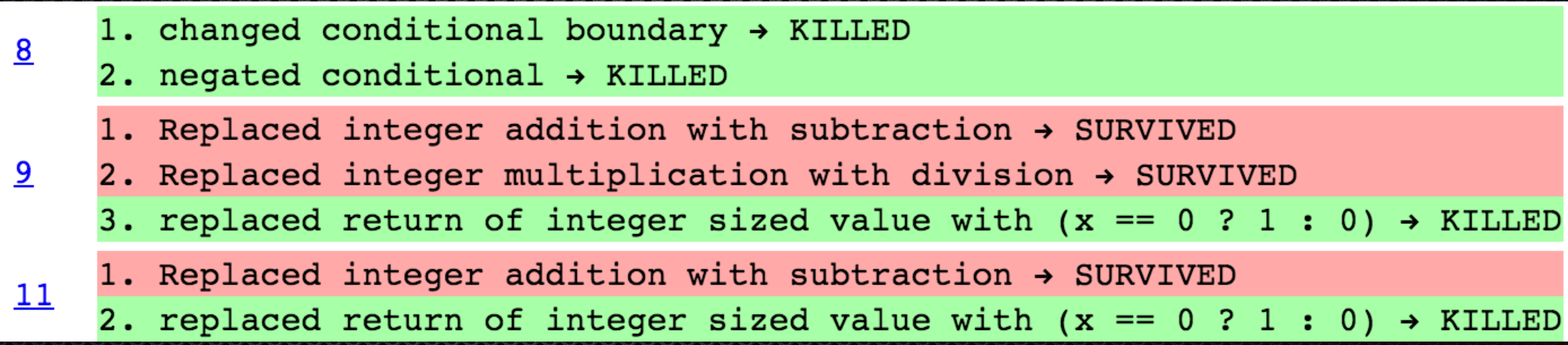
@SvenRuppert

```
final int add = new Service().add(0, 0);
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations Killed 4



```
 8 2      if (a < 2) {
 9 3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



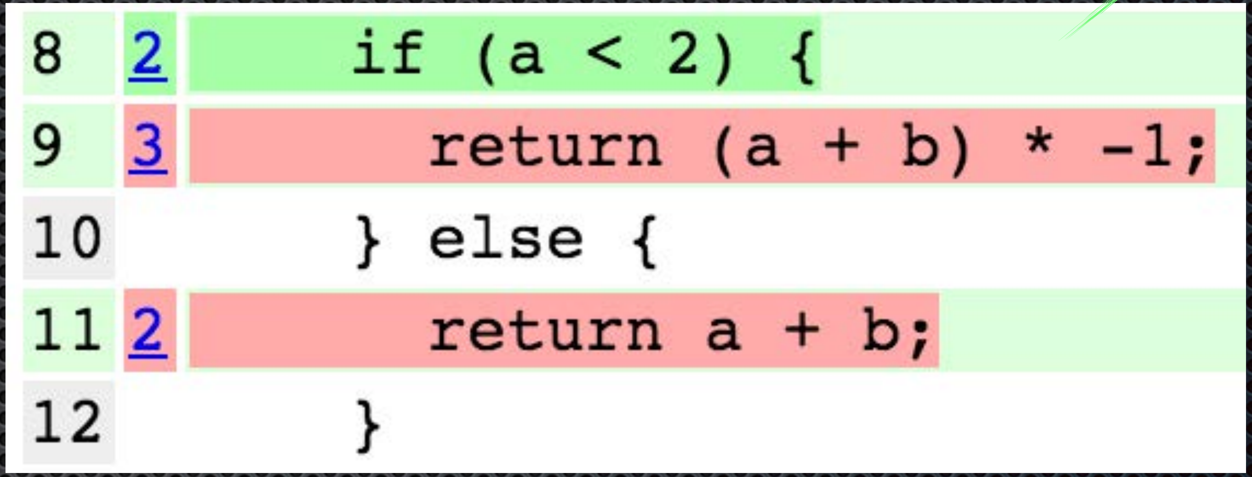
```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

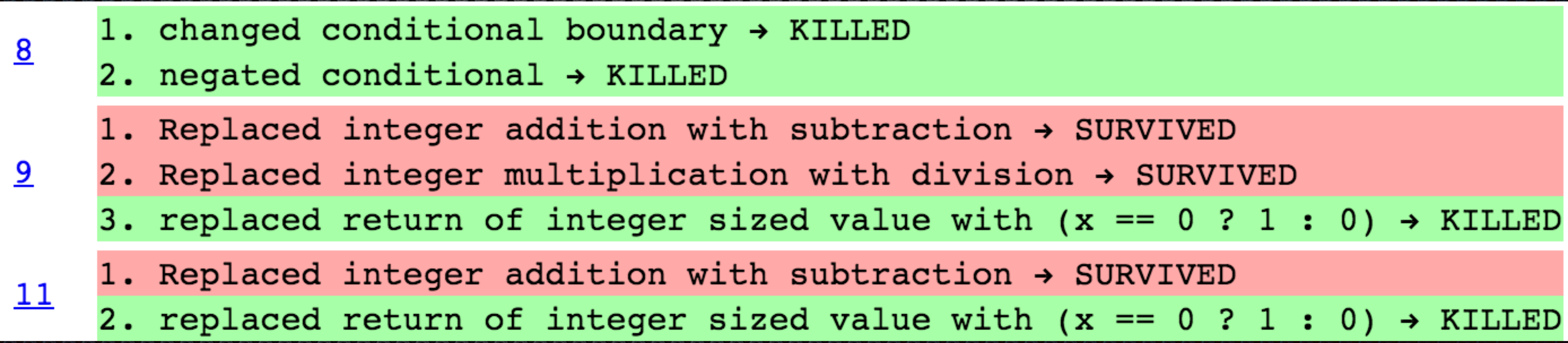
@SvenRuppert

```
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations **Killed 4**



```
8  2      if (a < 2) {
9  3          return (a + b) * -1;
10        } else {
11 2          return a + b;
12        }
```



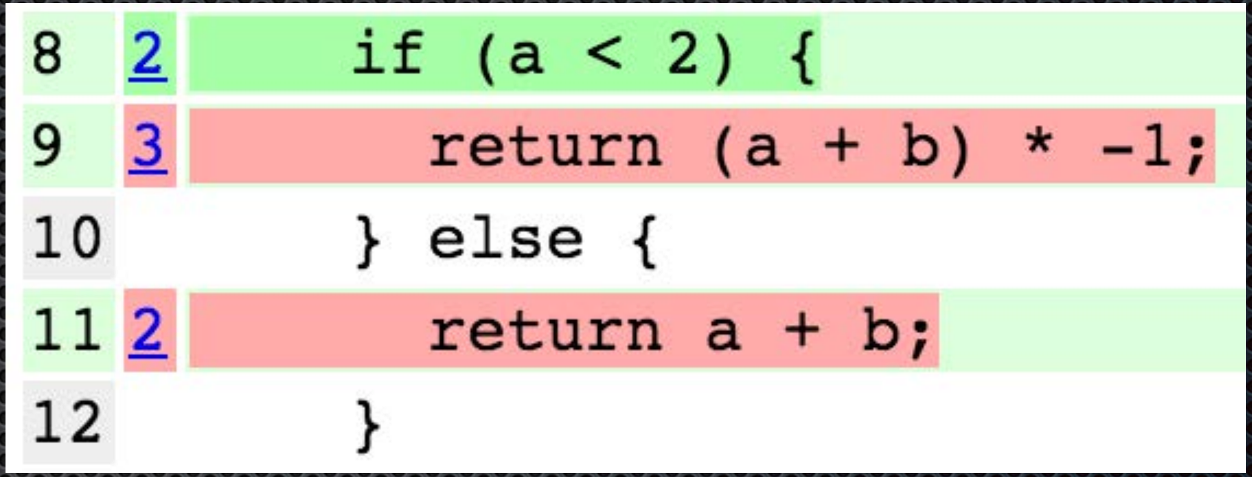
8	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
9	1. Replaced integer addition with subtraction → SURVIVED 2. Replaced integer multiplication with division → SURVIVED 3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11	1. Replaced integer addition with subtraction → SURVIVED 2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED

Mutation Testing - Hello World

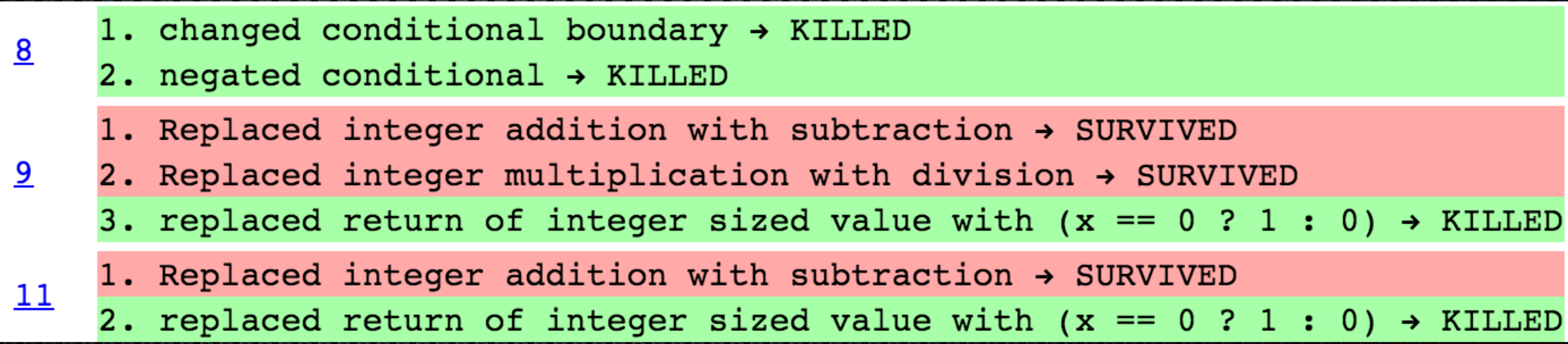
@SvenRuppert

```
final int add = new Service().add(2, 0);
```

>> Generated 54 mutations Killed 4



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



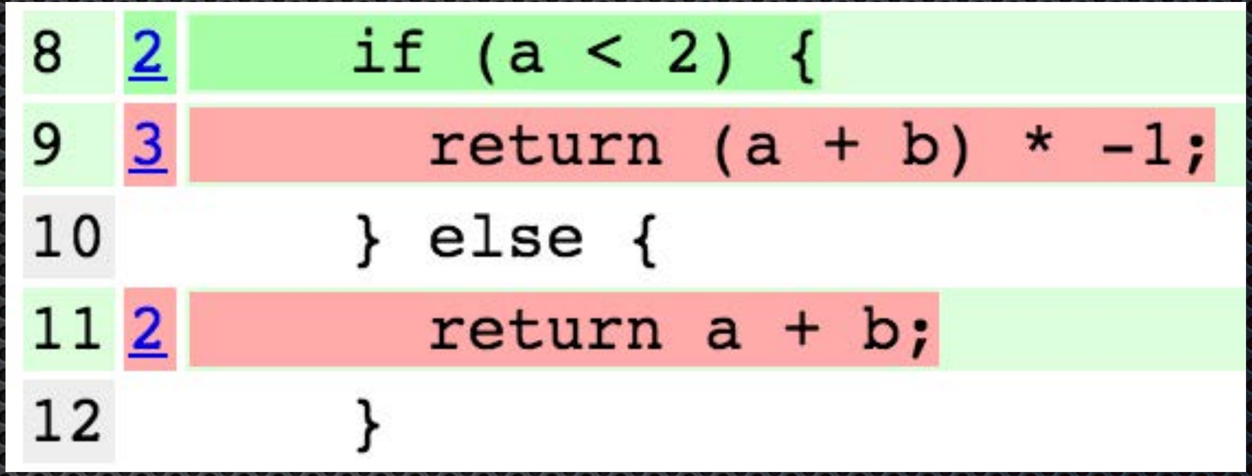
```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

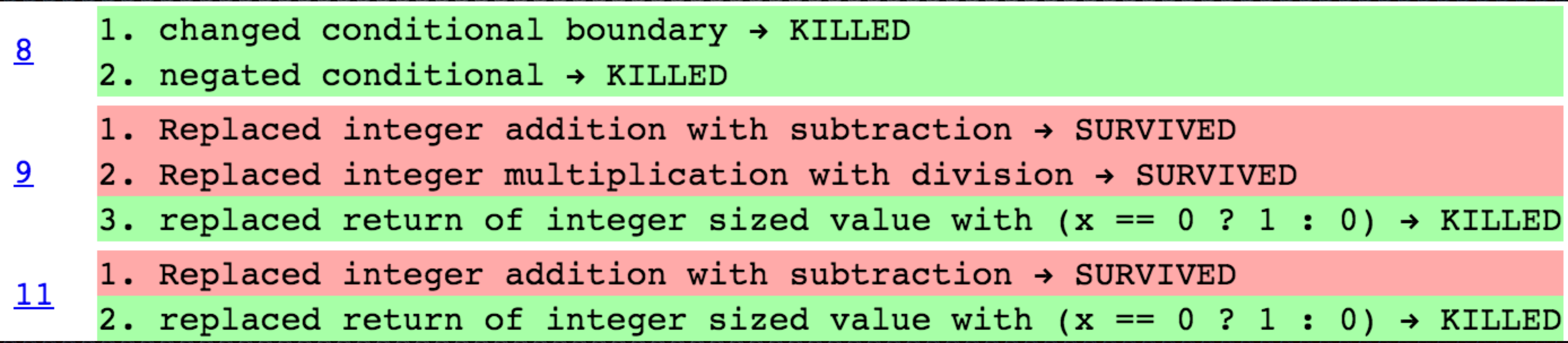
@SvenRuppert

```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
```

>> Generated 54 mutations Killed 4



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



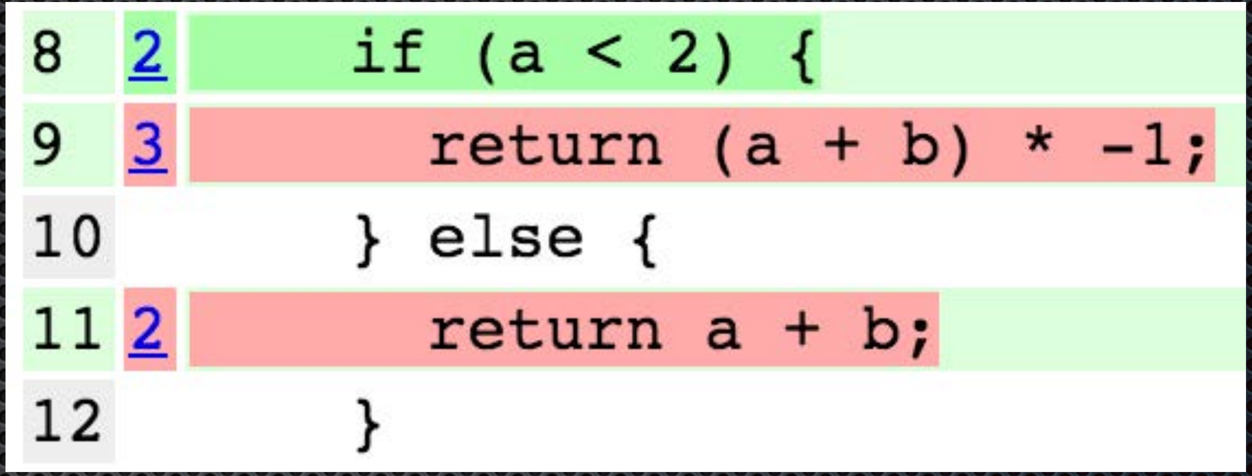
```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → SURVIVED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

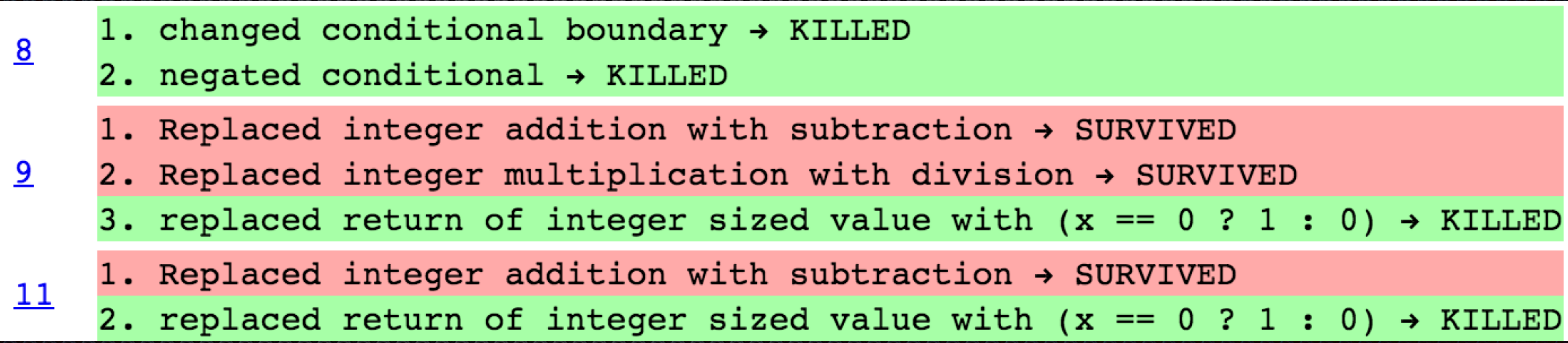
@SvenRuppert

```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
```

>> Generated 54 mutations



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



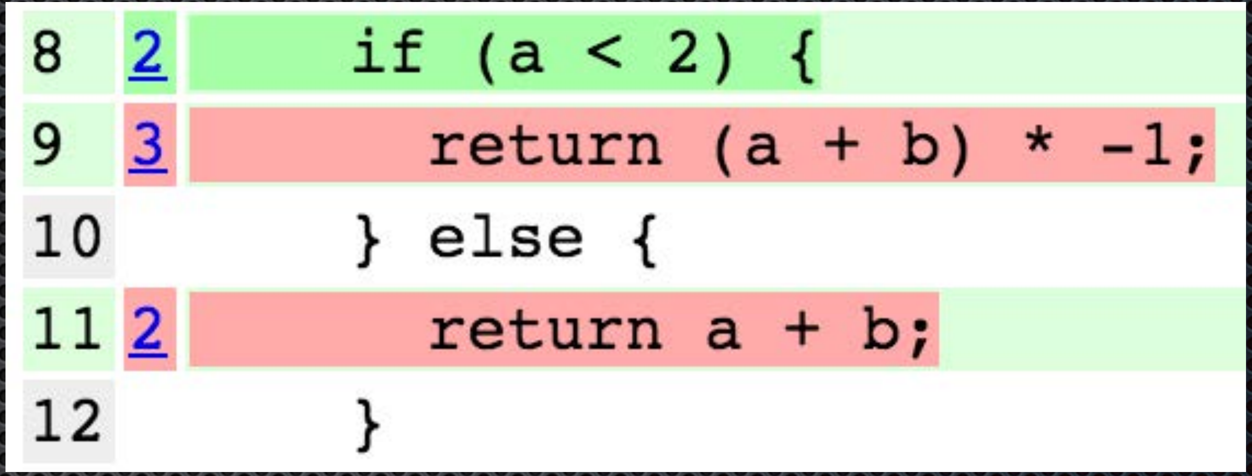
```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → SURVIVED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

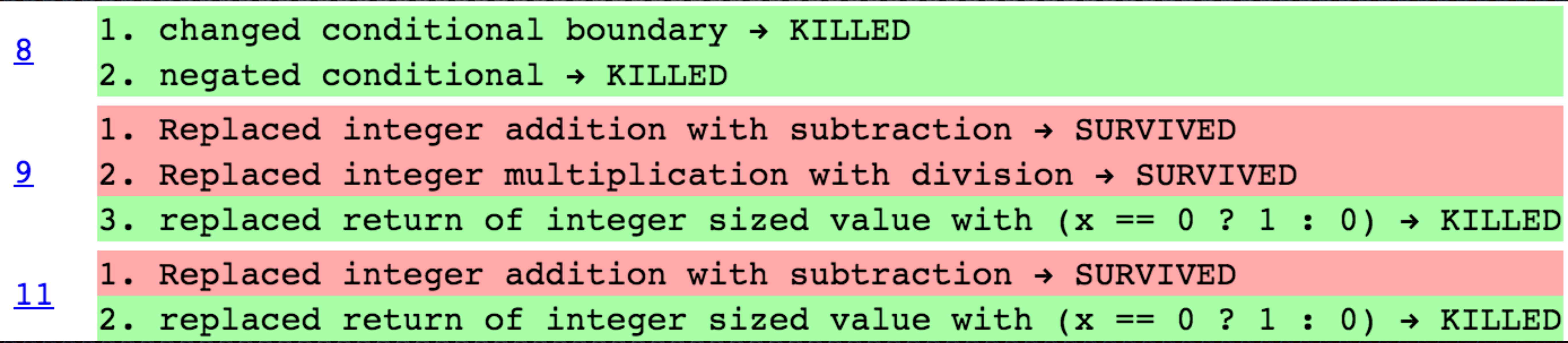
@SvenRuppert

```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
```

>> Generated 54 mutations Killed 5



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



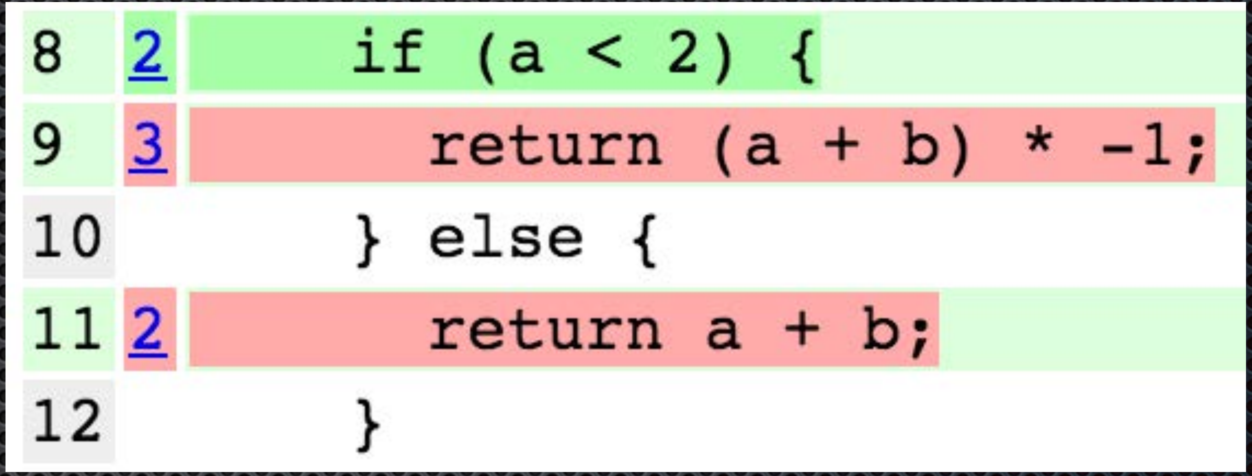
```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → SURVIVED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

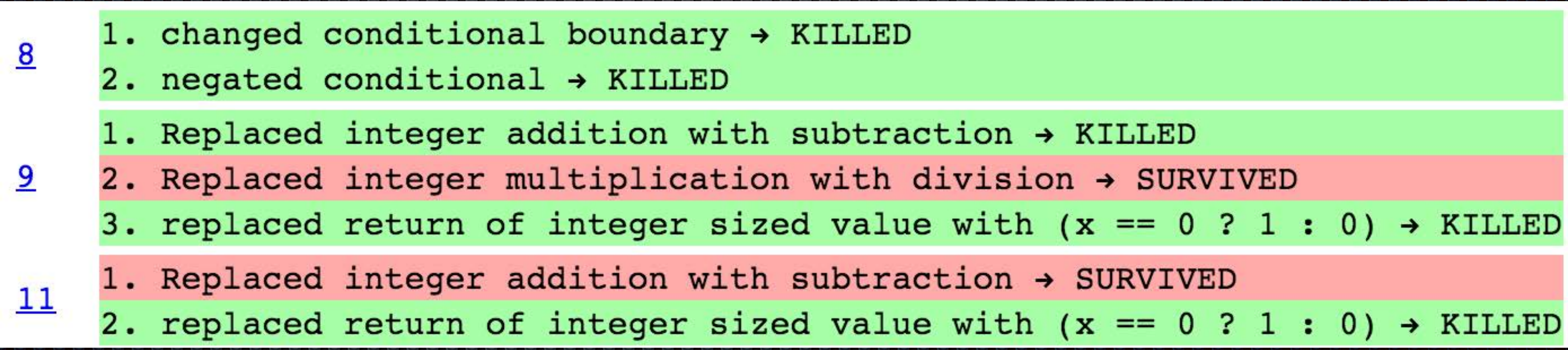
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
```

>> Generated 54 mutations Killed 5

killed 9:1



```
8  2      if (a < 2) {
9  3          return (a + b) * -1;
10        } else {
11 2          return a + b;
12        }
```



8	1. changed conditional boundary → KILLED 2. negated conditional → KILLED
9	1. Replaced integer addition with subtraction → KILLED 2. Replaced integer multiplication with division → SURVIVED 3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11	1. Replaced integer addition with subtraction → SURVIVED 2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED

Mutation Testing - Hello World

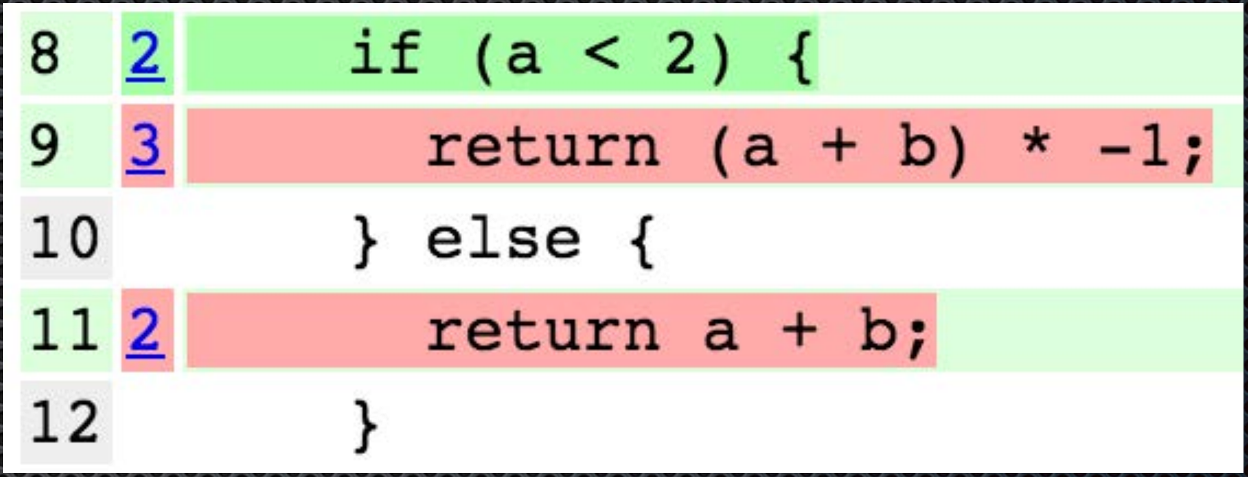
@SvenRuppert

```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
```

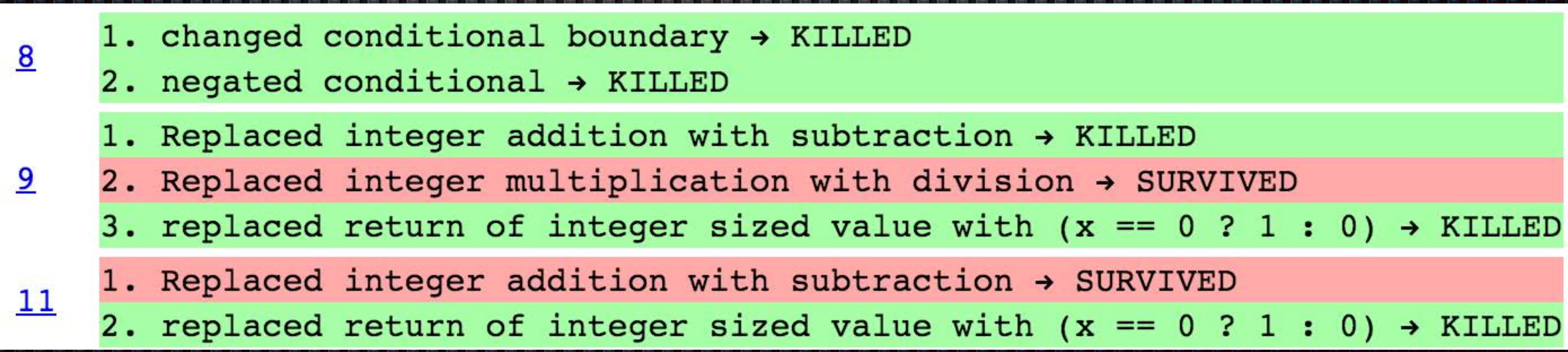
>> Generated 54 mutations

Killed 5

killed 9:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

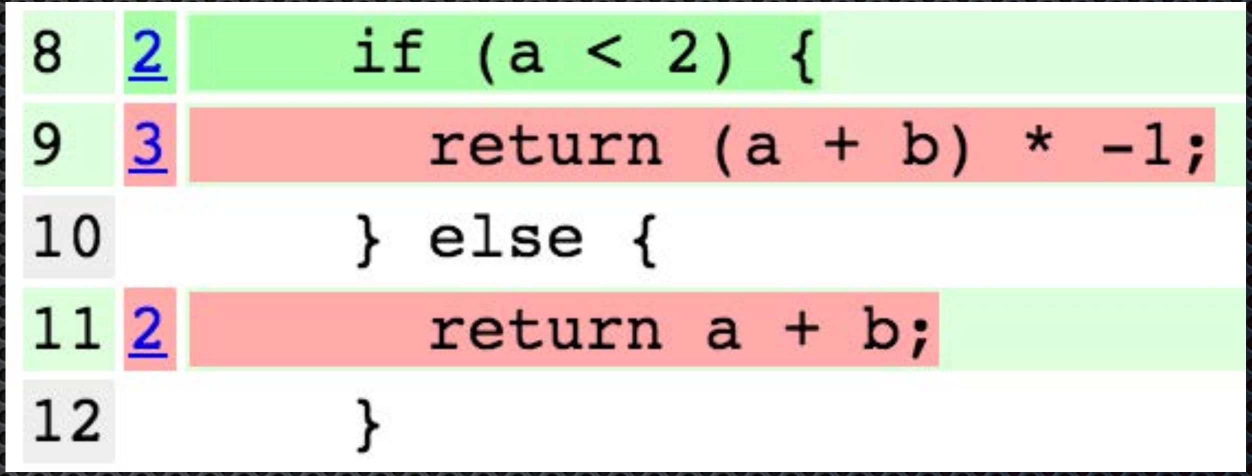
Mutation Testing - Hello World

@SvenRuppert

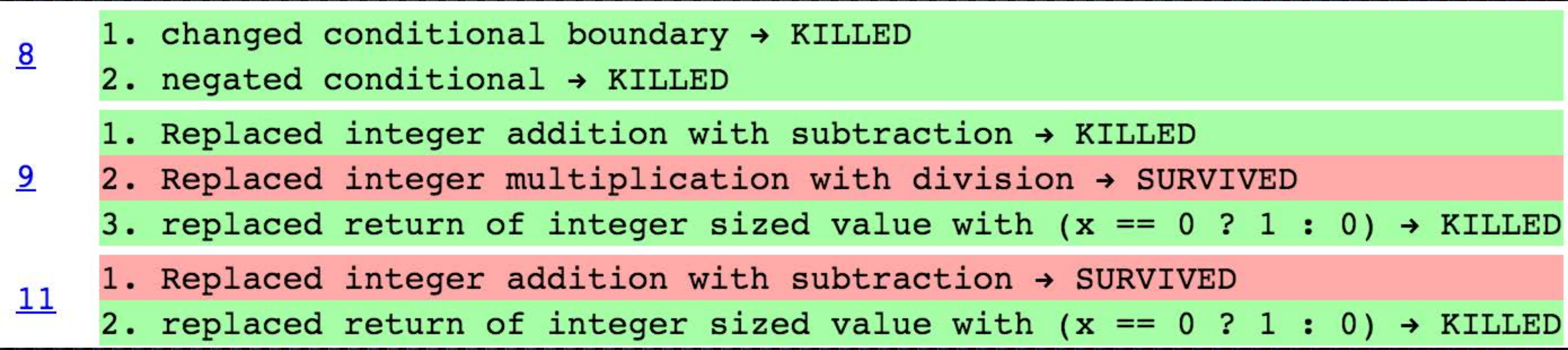
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations **Killed 5**

killed 9:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

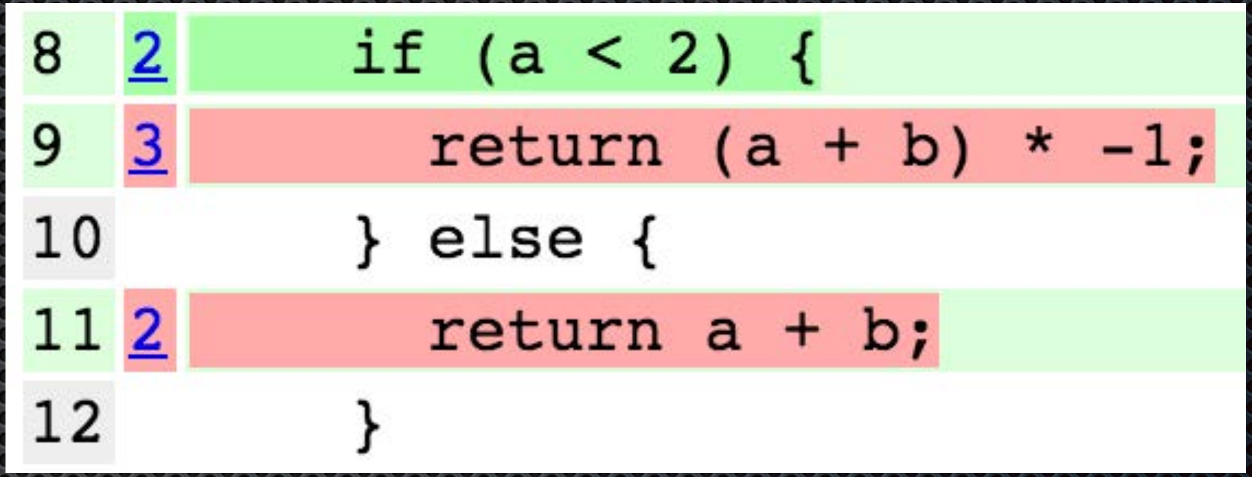
Mutation Testing - Hello World

@SvenRuppert

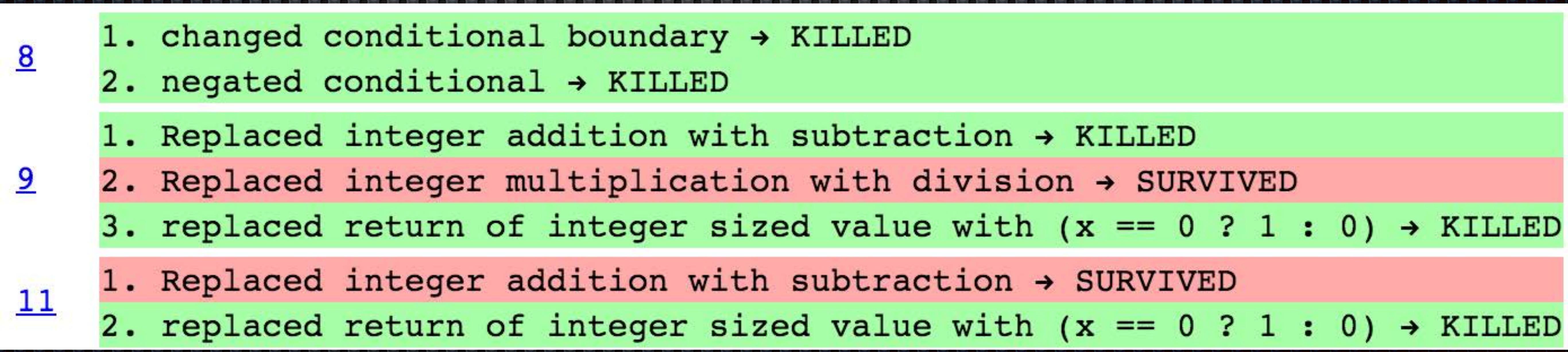
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations

killed 9:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → KILLED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

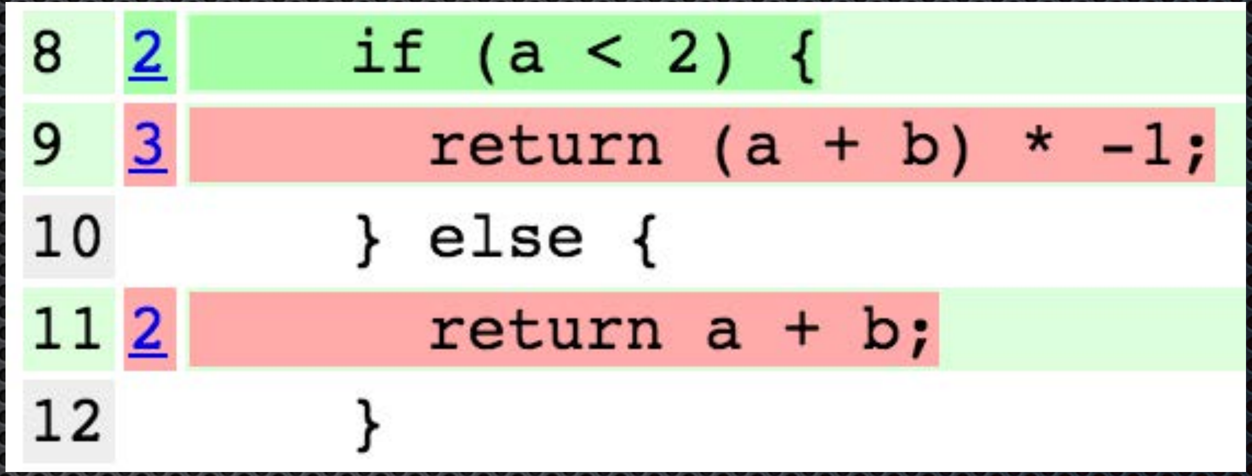
Mutation Testing - Hello World

@SvenRuppert

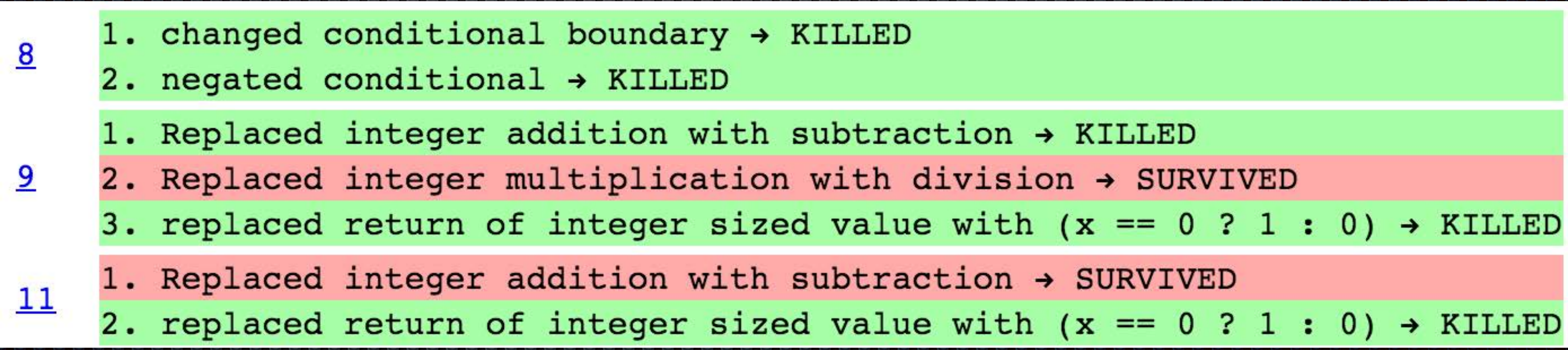
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations Killed 6

killed 9:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



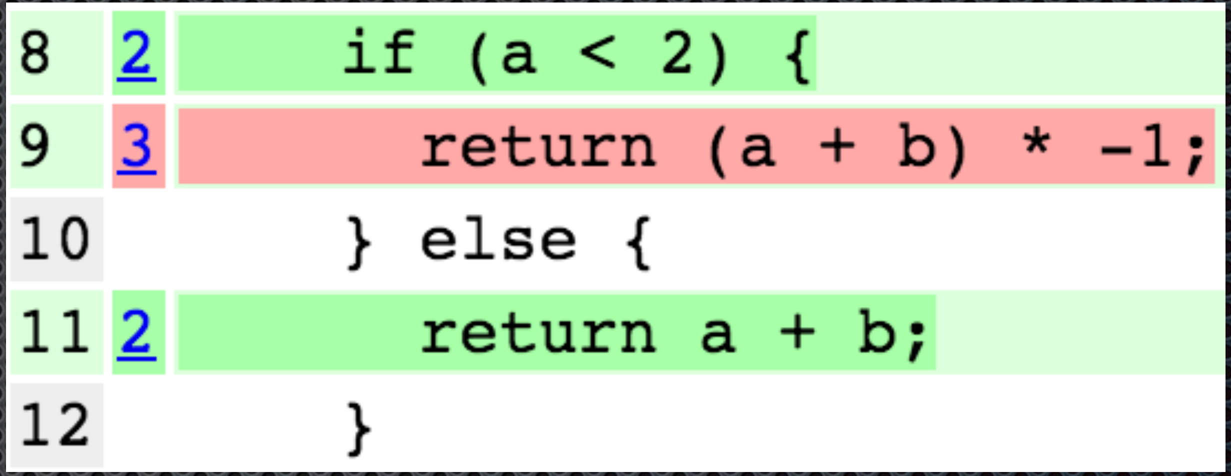
```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → SURVIVED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

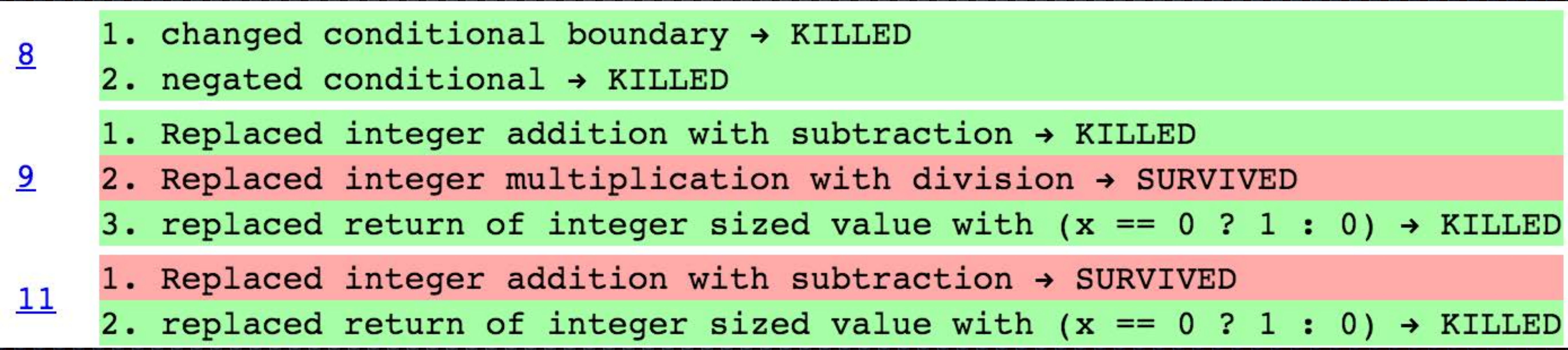
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations Killed 6

killed 9:1



```
8  2     if (a < 2) {
9  3       return (a + b) * -1;
10       } else {
11 2       return a + b;
12       }
```



```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → KILLED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → SURVIVED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

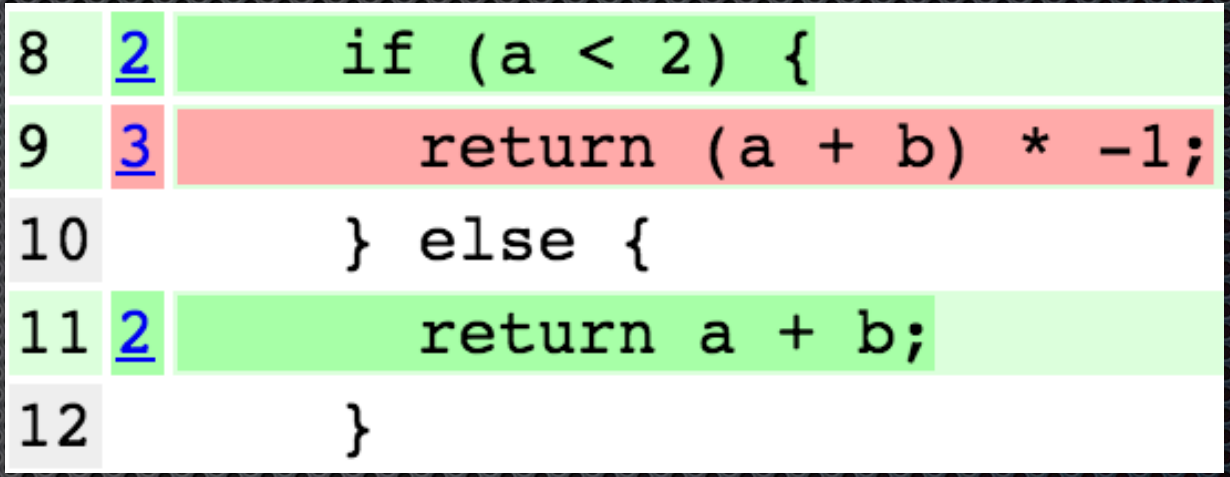
Mutation Testing - Hello World

@SvenRuppert

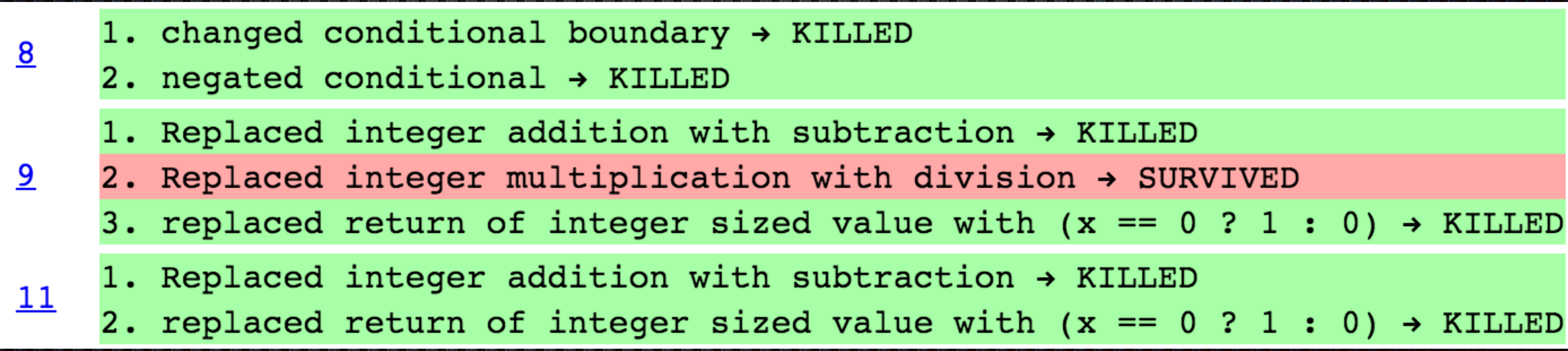
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations Killed 6

killed 9:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → KILLED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

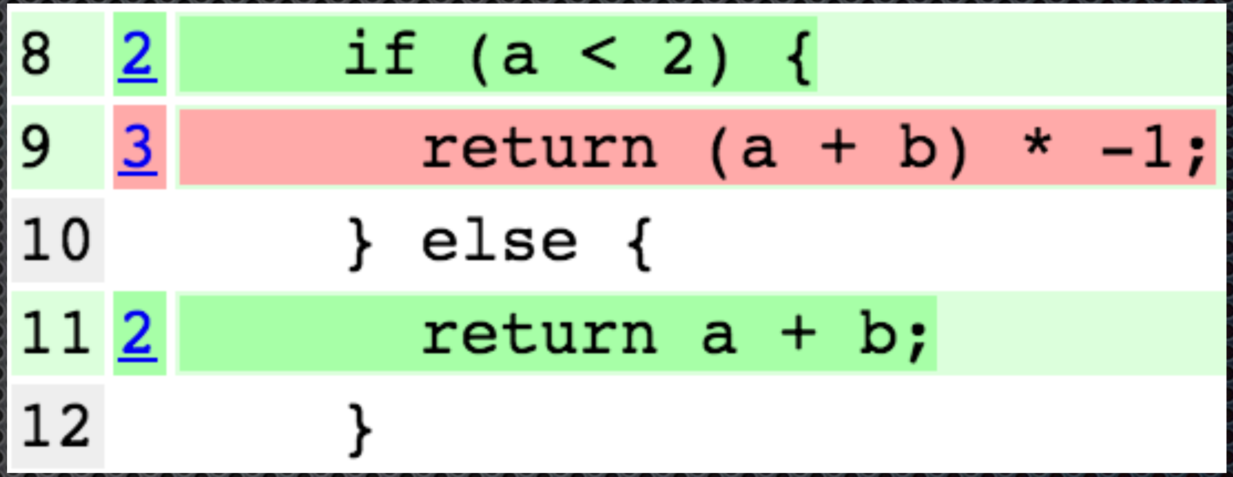
Mutation Testing - Hello World

@SvenRuppert

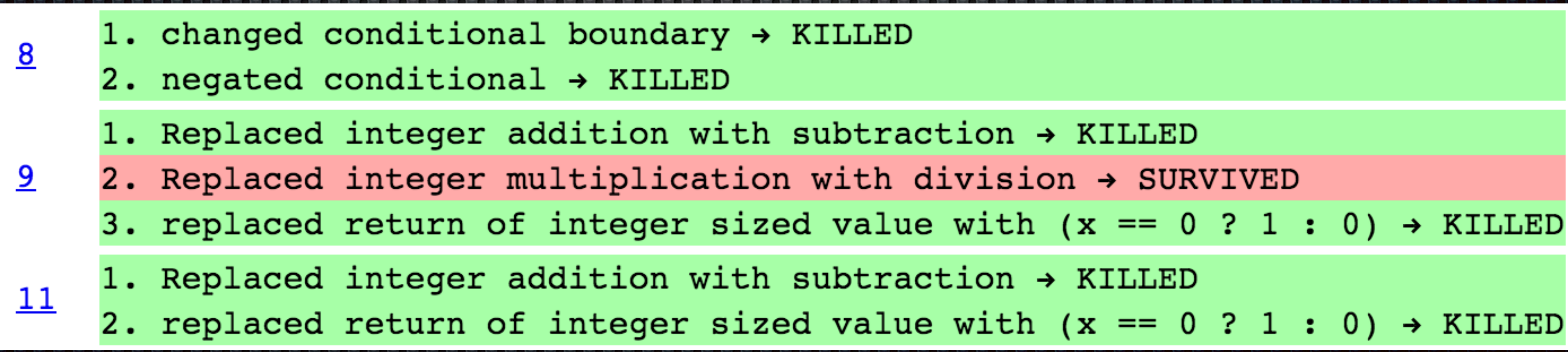
```
final int add = new Service().add(2, 0);
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations   Killed 6

killed 11:1



```
8   2       if (a < 2) {
9   3         return (a + b) * -1;
10          } else {
11  2         return a + b;
12          }
```



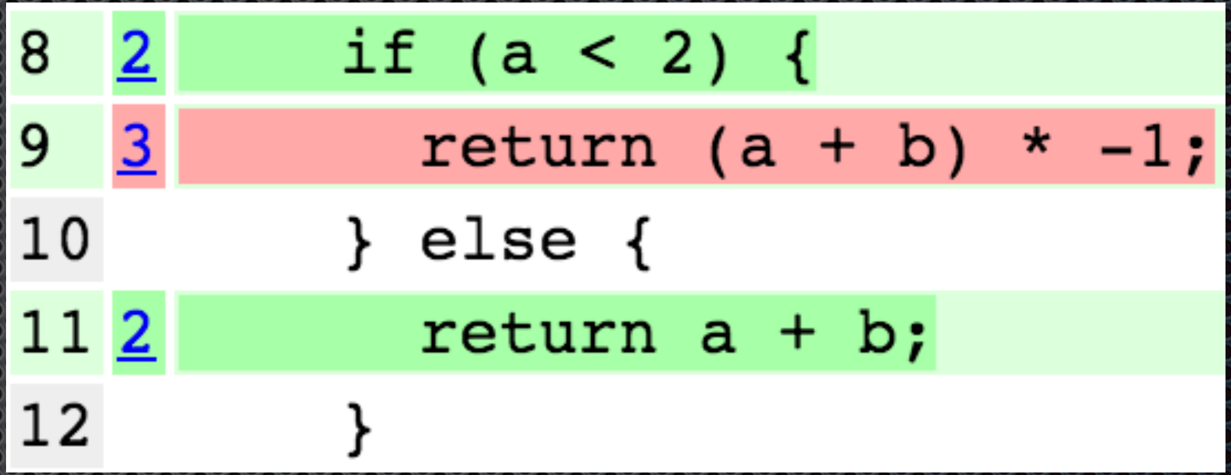
```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → KILLED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → KILLED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

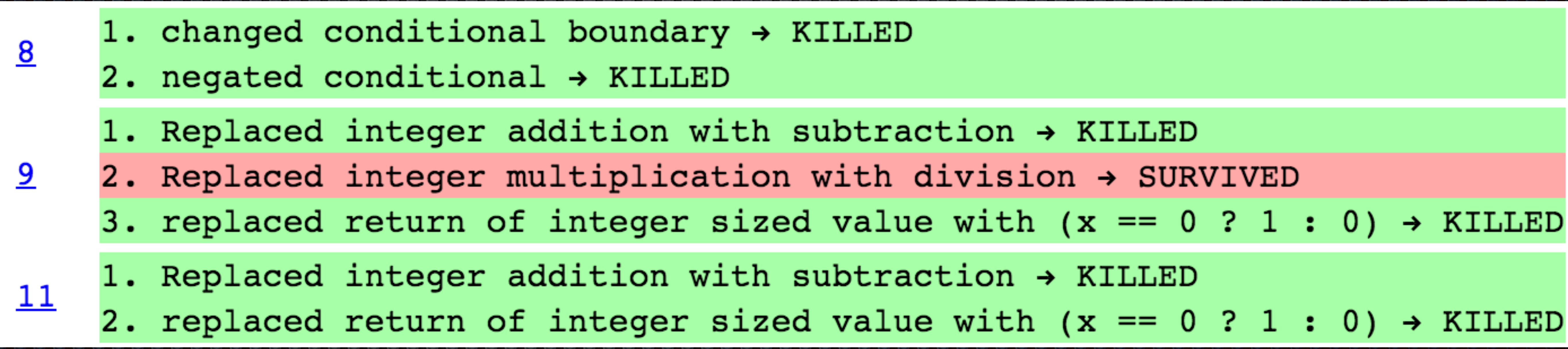
```
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations **Killed 6**

killed 11:1



```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```



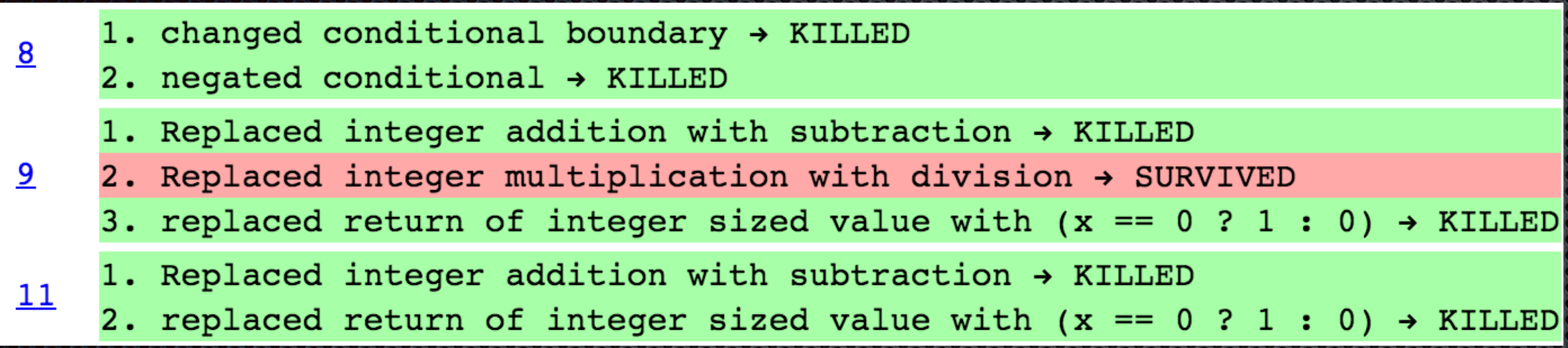
```
8    1. changed conditional boundary → KILLED
     2. negated conditional → KILLED
9    1. Replaced integer addition with subtraction → KILLED
     2. Replaced integer multiplication with division → SURVIVED
     3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11   1. Replaced integer addition with subtraction → KILLED
     2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```

Mutation Testing - Hello World

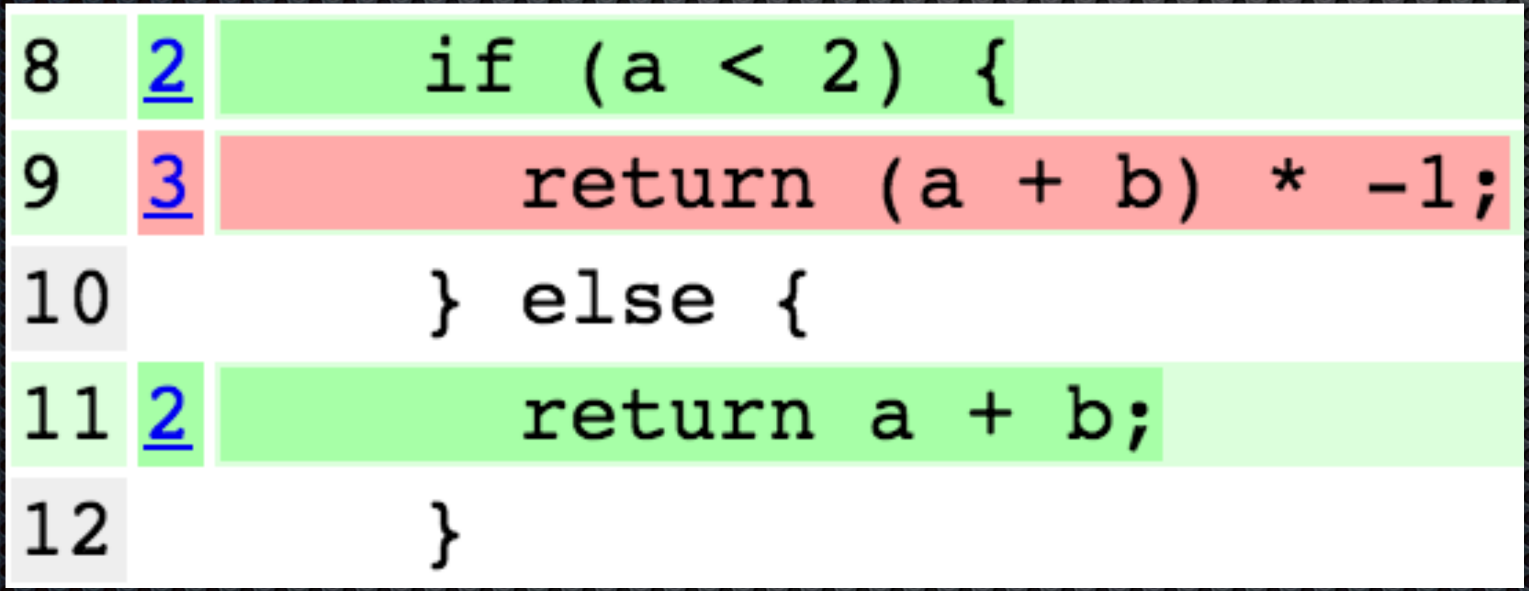
@SvenRuppert

```
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations   Killed 6



```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → KILLED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```



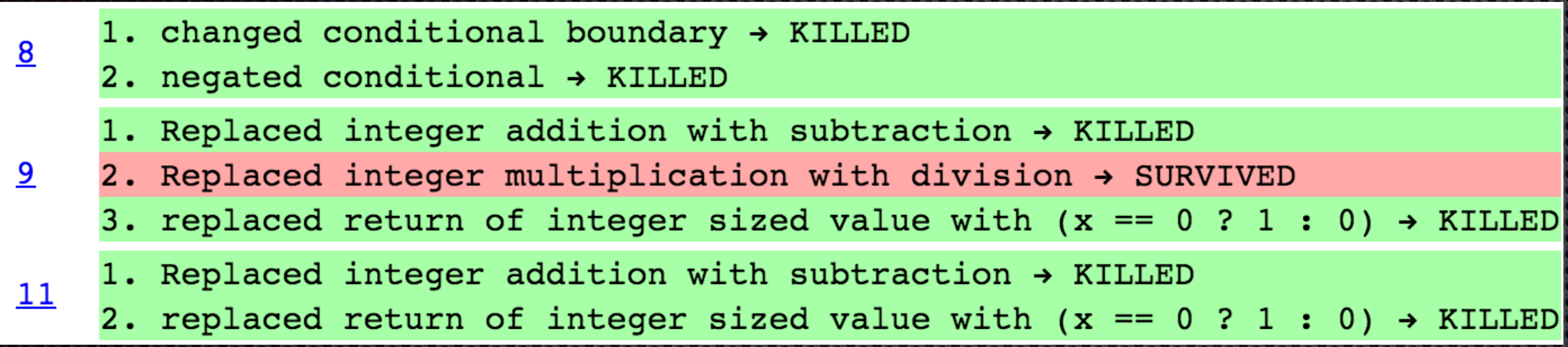
```
8  2      if (a < 2) {
9  3        return (a + b) * -1;
10        } else {
11 2        return a + b;
12        }
```

Mutation Testing - Hello World

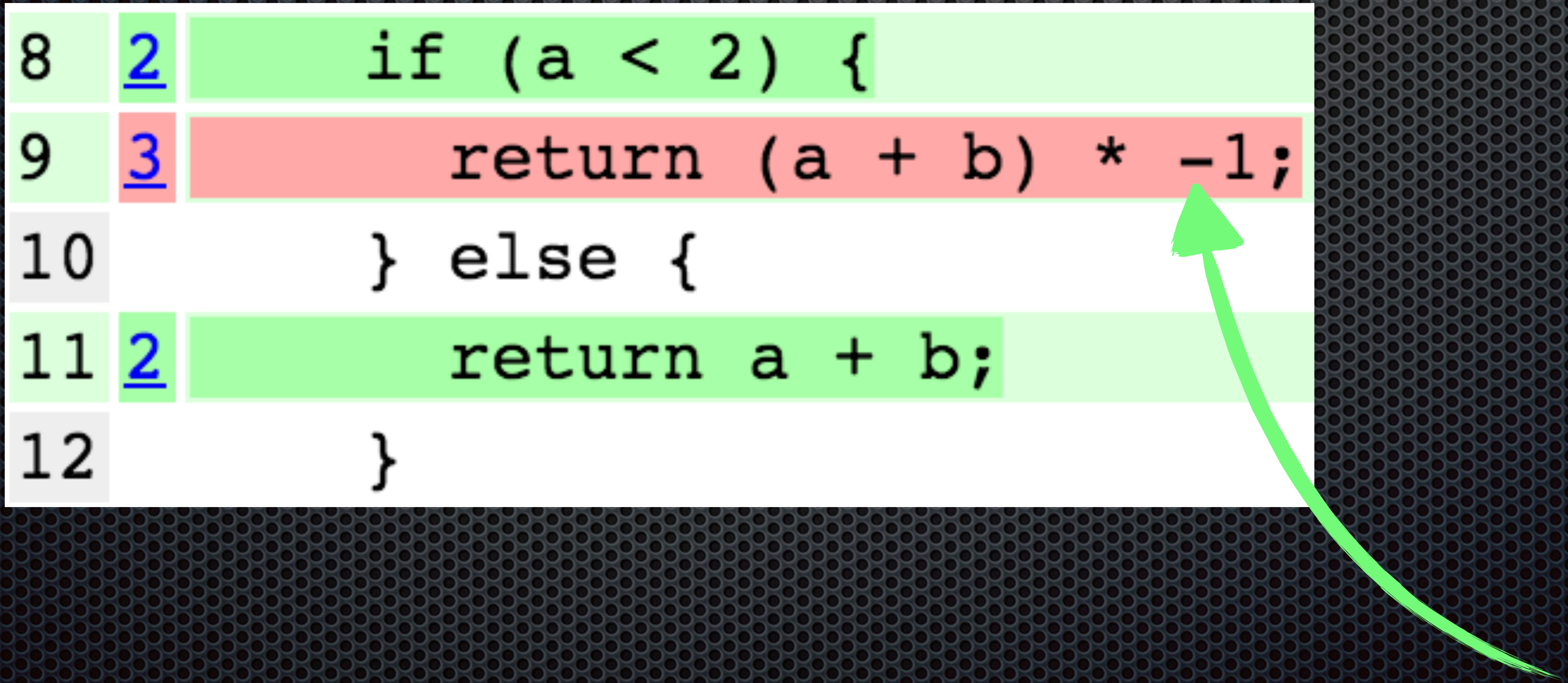
@SvenRuppert

```
final int add = new Service().add(1, 1);
final int add = new Service().add(2, 2);
```

>> Generated 54 mutations Killed 6



```
8   1. changed conditional boundary → KILLED
    2. negated conditional → KILLED
9   1. Replaced integer addition with subtraction → KILLED
    2. Replaced integer multiplication with division → SURVIVED
    3. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
11  1. Replaced integer addition with subtraction → KILLED
    2. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
```



```
8   2      if (a < 2) {
9   3        return (a + b) * -1;
10         } else {
11  2        return a + b;
12         }
```

Mutation Testing - Lesson Learned

Mutation Testing - Lesson Learned

@SvenRuppert

mutation tests are often leading to

Mutation Testing - Lesson Learned

mutation tests are often leading to

…cleaner code compared to jUnit only

Mutation Testing - Lesson Learned

@SvenRuppert

```
public static final String[] discardCommonPrefix(String a, String b) {
  String[] ret = { a, b };
  int l = a.length() < b.length() ? a.length() : b.length();
  for (int i = 0; i < l; i++) {
    if (a.charAt(i) == b.charAt(i)) {
      if (i + 1 < l) { ret[0] = a.substring(i + 1); ret[1] = b.substring(i + 1); }
      else {
        if (a.length() < b.length())   { ret[0] = ""; ret[1] = b.substring(i + 1); }
        if (a.length() == b.length()) { ret[0] = ""; ret[1] = „"; }
        if (a.length() > b.length())   { ret[0] = a.substring(i + 1); ret[1] = „"; }
      }
    } else break;
  }
  return ret;
}
```

Mutation Testing - Lesson Learned

@SvenRuppert

```
public String[] discardCommonPrefix(String a, String b) {
  final String[] ret = new String[2];
  int l;
  if (a.length() < b.length()) { l = a.length(); }
  else                         { l = b.length(); }

  int position = 0;
  for (; position < l; position++) {
    final char charA = a.charAt(position);
    final char charB = b.charAt(position);
    if (charA != charB) { break; }
  }

  if (position >= a.length()) { ret[0] = ""; }
  else                        { ret[0] = a.substring(position); }

  if (position >= b.length()) { ret[1] = ""; }
  else                        { ret[1] = b.substring(position); }
  return ret;
}
```

Mutation Testing - Lesson Learned

Version 1

Version 2

Mutation Testing - Lesson Learned

Version 1

```
for {
   if {
     if
     else {
       if
       if
       if
     }
   } else
 }
```

Version 2

Mutation Testing - Lesson Learned

@SvenRuppert

Version 1

```
for {
   if {
     if
     else {
       if
       if
       if
     }
   } else
}
```

Version 2

```
if else

for {
    if
}

if  else

if  else
```

Mutation Testing - Lesson Learned

mutation tests are often leading to

…cleaner code compared to jUnit only

Mutation Testing - Lesson Learned

mutation tests are often leading to

…cleaner code compared to jUnit only

… smaller modules (shorter mutation runtime)

Mutation Testing - Lesson Learned

mutation tests are often leading to

…cleaner code compared to jUnit only
… smaller modules (shorter mutation runtime)

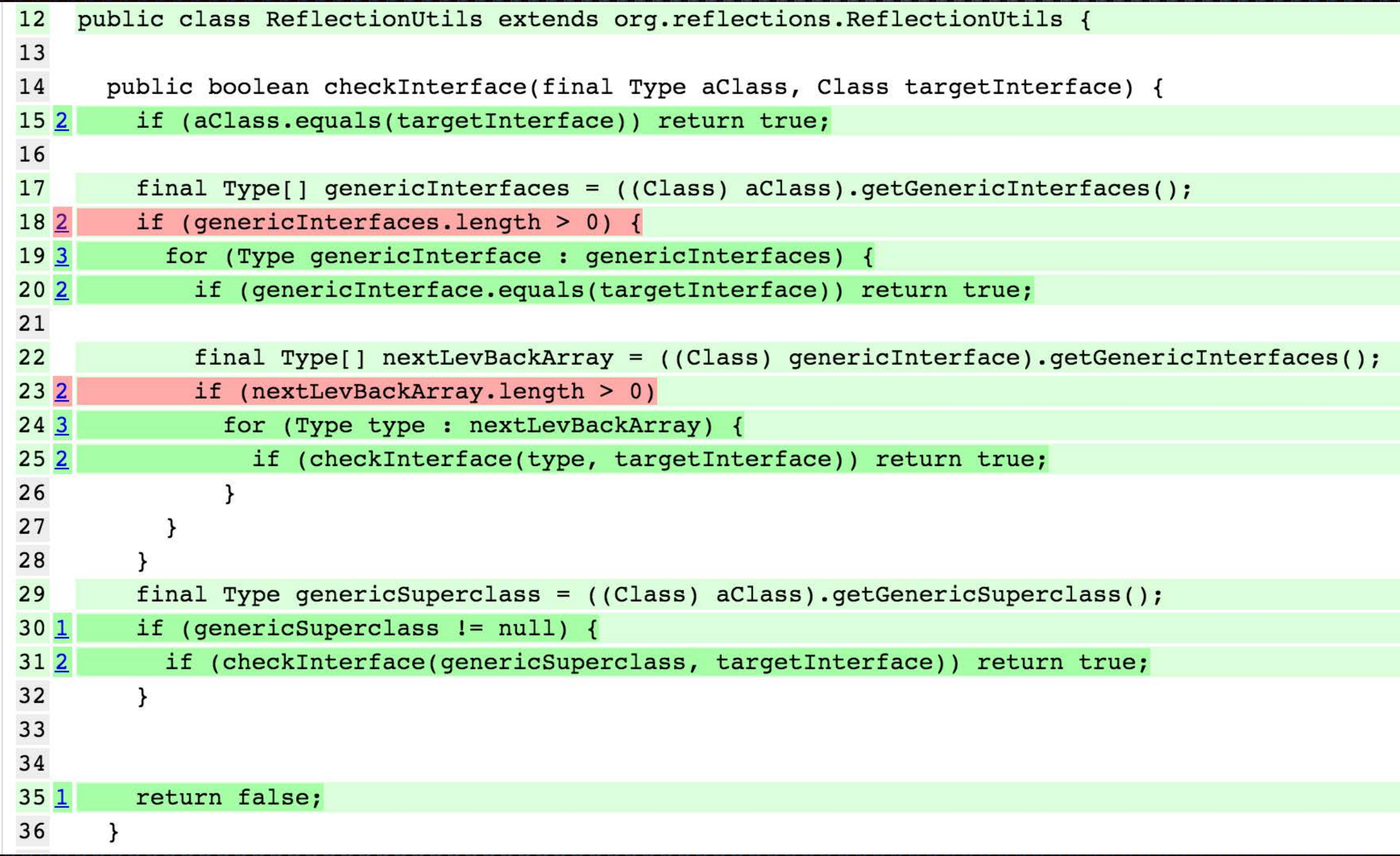
and something nice…
helps to find useless code

Example of useless Code

@SvenRuppert

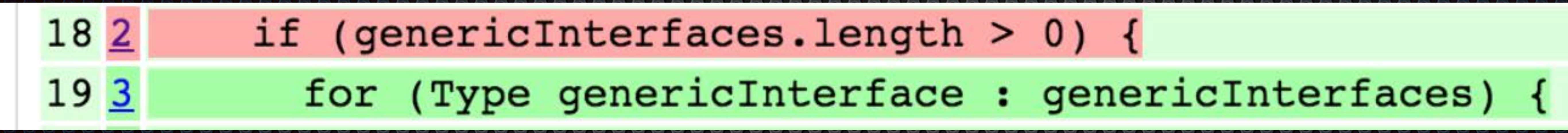
Example of useless Code

@SvenRuppert



```
  public class ReflectionUtils extends org.reflections.ReflectionUtils {

    public boolean checkInterface(final Type aClass, Class targetInterface) {
2     if (aClass.equals(targetInterface)) return true;

      final Type[] genericInterfaces = ((Class) aClass).getGenericInterfaces();
2     if (genericInterfaces.length > 0) {
3       for (Type genericInterface : genericInterfaces) {
2         if (genericInterface.equals(targetInterface)) return true;

          final Type[] nextLevBackArray = ((Class) genericInterface).getGenericInterfaces();
2         if (nextLevBackArray.length > 0)
3           for (Type type : nextLevBackArray) {
2             if (checkInterface(type, targetInterface)) return true;
            }
        }
      }
      final Type genericSuperclass = ((Class) aClass).getGenericSuperclass();
1     if (genericSuperclass != null) {
2       if (checkInterface(genericSuperclass, targetInterface)) return true;
      }


1     return false;
    }
```

Example of useless Code



```
    if (genericInterfaces.length > 0) {
      for (Type genericInterface : genericInterfaces) {
```

Mutation Testing - How to start

Mutation Testing - How to start

you need jUnit - to generate the reference

Mutation Testing - How to start

- you need jUnit - to generate the reference
  - add the pitest-plugin to the build section

Mutation Testing - How to start

- you need jUnit - to generate the reference
  - add the pitest-plugin to the build section
    - configure the plugin

Mutation Testing - How to start

- you need jUnit - to generate the reference
  - add the pitest-plugin to the build section
    - configure the plugin
      - generate the reference -> clean , install

Mutation Testing - How to start

- you need jUnit - to generate the reference
  - add the pitest-plugin to the build section
    - configure the plugin
      - generate the reference -> clean , install
        - run **pitest: mutationCoverage**

Mutation Testing - How to start

you need jUnit - to generate the reference

add the pitest-plugin to the build section

configure the plugin

generate the reference -> clean , install

run **pitest: mutationCoverage**

report will be under **target/pit-reports**

Mutation Testing - How to start

pom.xml - example - build

```
<plugin>
  <groupId>org.pitest</groupId>
  <artifactId>pitest-maven</artifactId>
  <configuration>
    <outputFormats>
      <outputFormat>XML</outputFormat>
      <outputFormat>HTML</outputFormat>
    </outputFormats>
    <targetClasses>
      <param>org.rapidpm.*</param>
    </targetClasses>
    <targetTests>
      <param>org.rapidpm.*</param>
      <param>junit.org.rapidpm.*</param>
    </targetTests>
  </configuration>
</plugin>
```

Mutation Testing - How to start

pom.xml - example - reporting

```
<reporting>
  <plugins>
    <plugin>
      <groupId>org.pitest</groupId>
      <artifactId>pitest-maven</artifactId>
      <reportSets>
        <reportSet>
          <reports>
            <report>report</report>
          </reports>
        </reportSet>
      </reportSets>
    </plugin>
  </plugins>
</reporting>
```

Mutation Testing - practical usage

@SvenRuppert

Mutation Testing - practical usage

Start with some tests

Mutation Testing - practical usage

- Start with some tests
    - generate the pitest report

Mutation Testing - practical usage

- Start with some tests
  - generate the pitest report
    - write more tests to kill mutations

Mutation Testing - practical usage

- Start with some tests
  - generate the pitest report
    - write more tests to kill mutations
      - if you have time, eliminate useless tests

Mutation Testing - practical usage

- Start with some tests
  - generate the pitest report
    - write more tests to kill mutations
      - if you have time, eliminate useless tests
        - do it one by one

Mutation Testing - practical usage

@SvenRuppert

Start with some tests
- generate the pitest report
  - write more tests to kill mutations
    - if you have time, eliminate useless tests
      - do it one by one

Mutation 001 **Survived**

Mutation 002 **Survived**

Mutation 003 **Survived**

Mutation 004 **Survived**

Mutation Testing - practical usage

@SvenRuppert

Start with some tests
- generate the pitest report
  - write more tests to kill mutations
    - if you have time, eliminate useless tests
      - do it one by one

Mutation 001 **Survived** ····▶ **Killed**

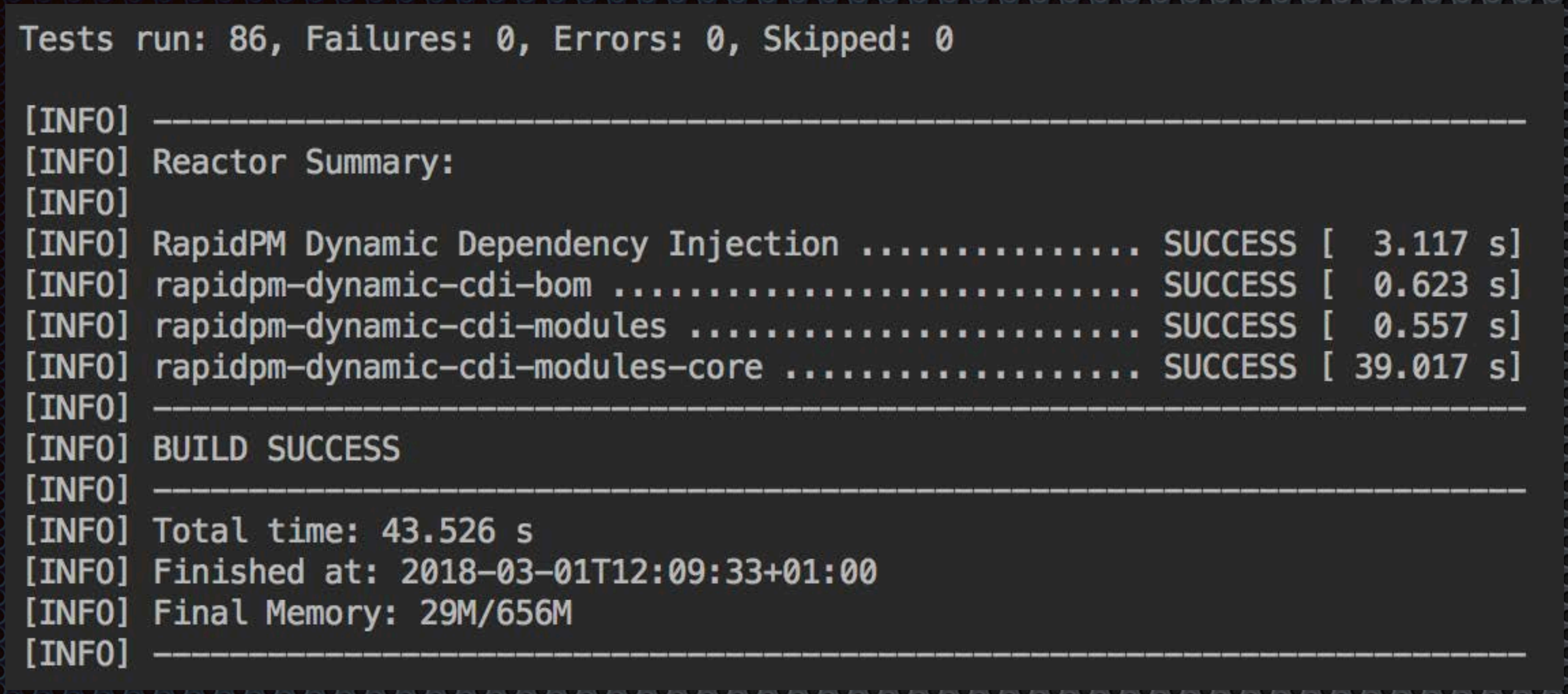
Mutation 002 **Survived** ····▶ **Killed**

Mutation 003 **Survived** ····▶ **Killed**

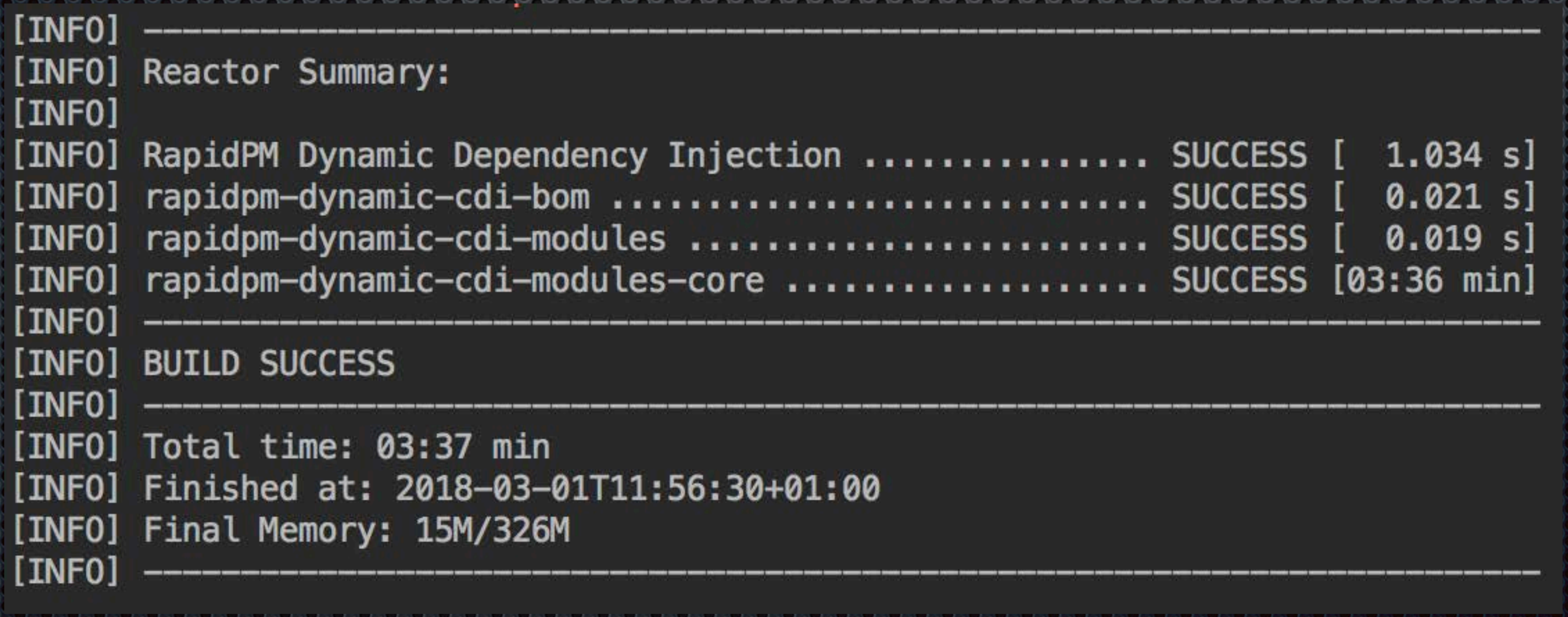
Mutation 004 **Survived** ····▶ **Killed**

Mutation Testing - Real Code Examples

@SvenRuppert



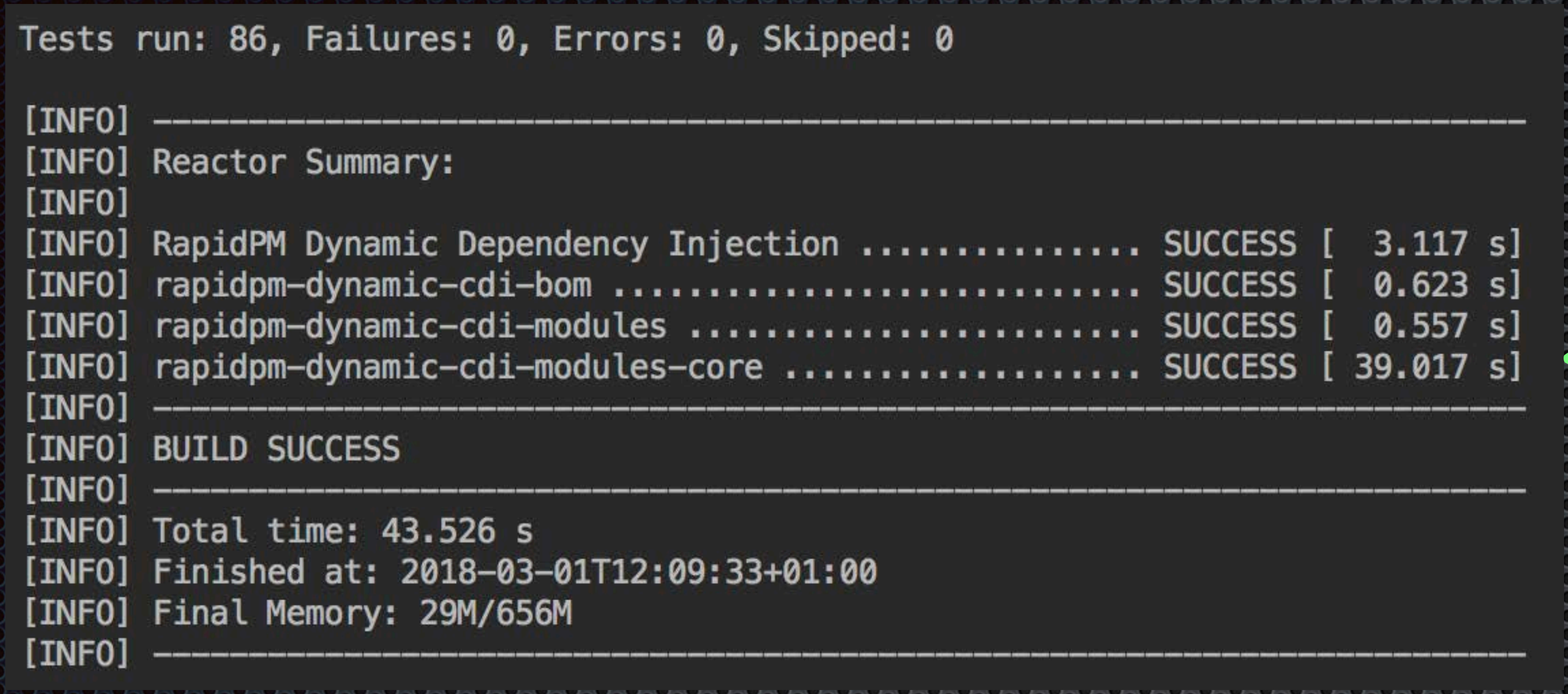
```
Tests run: 86, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  3.117 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.623 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.557 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [ 39.017 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 43.526 s
[INFO] Finished at: 2018-03-01T12:09:33+01:00
[INFO] Final Memory: 29M/656M
[INFO] ------------------------------------------------------------------------
```



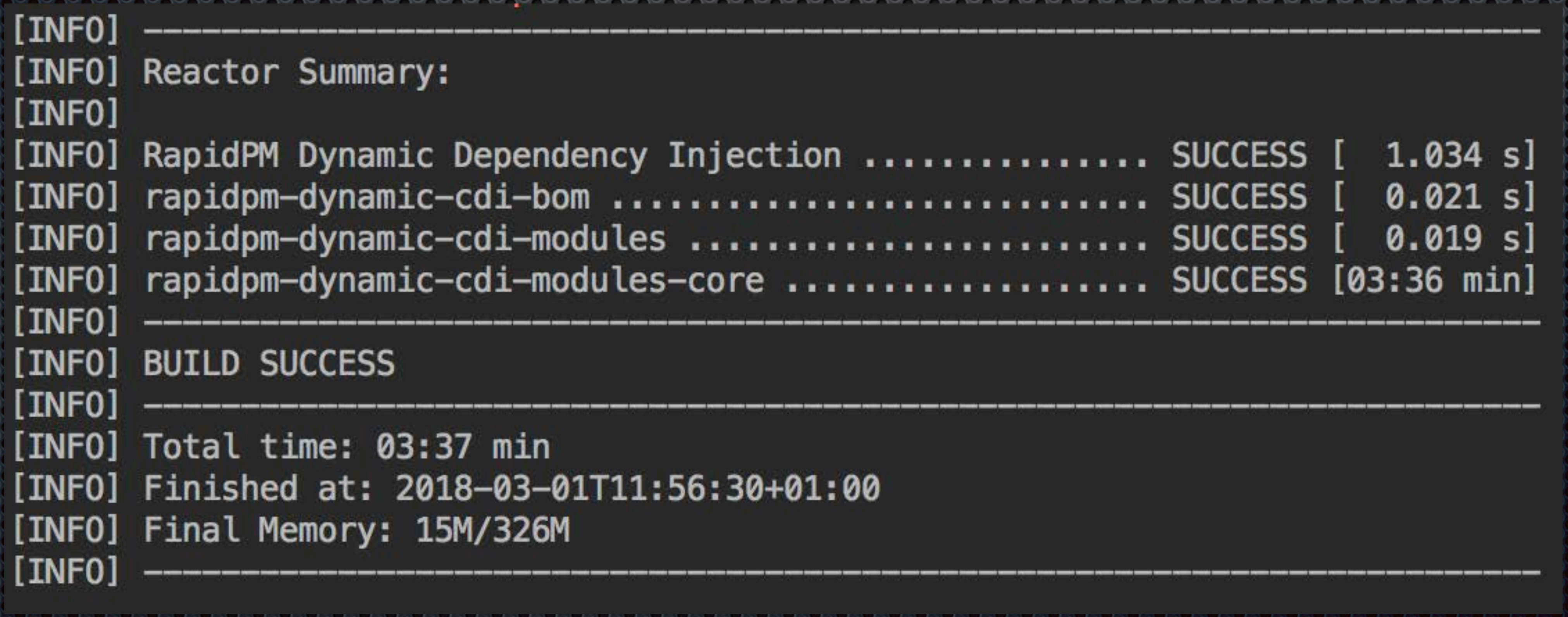
```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  1.034 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.021 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.019 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [03:36 min]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 03:37 min
[INFO] Finished at: 2018-03-01T11:56:30+01:00
[INFO] Final Memory: 15M/326M
[INFO] ------------------------------------------------------------------------
```

Mutation Testing - Real Code Examples

@SvenRuppert



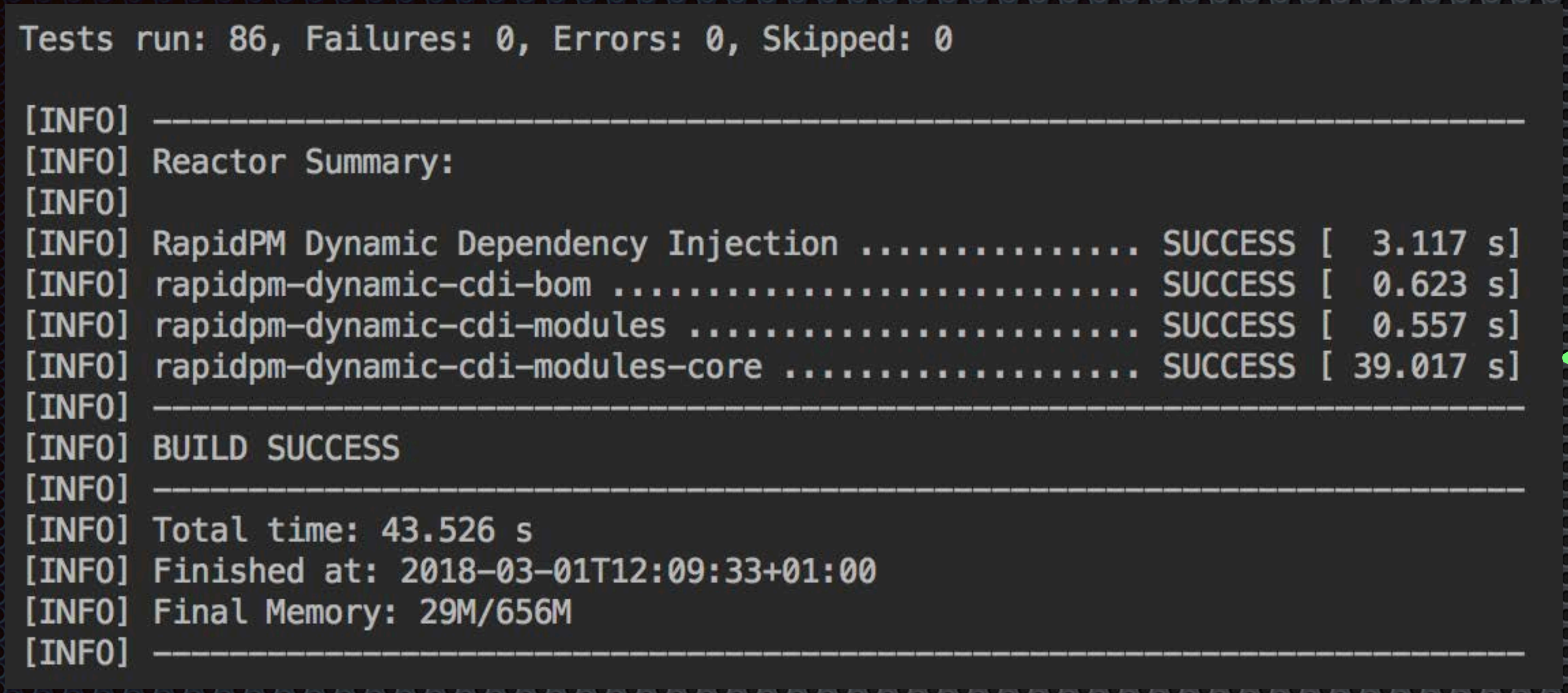
```
Tests run: 86, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  3.117 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.623 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.557 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [ 39.017 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 43.526 s
[INFO] Finished at: 2018-03-01T12:09:33+01:00
[INFO] Final Memory: 29M/656M
[INFO] ------------------------------------------------------------------------
```



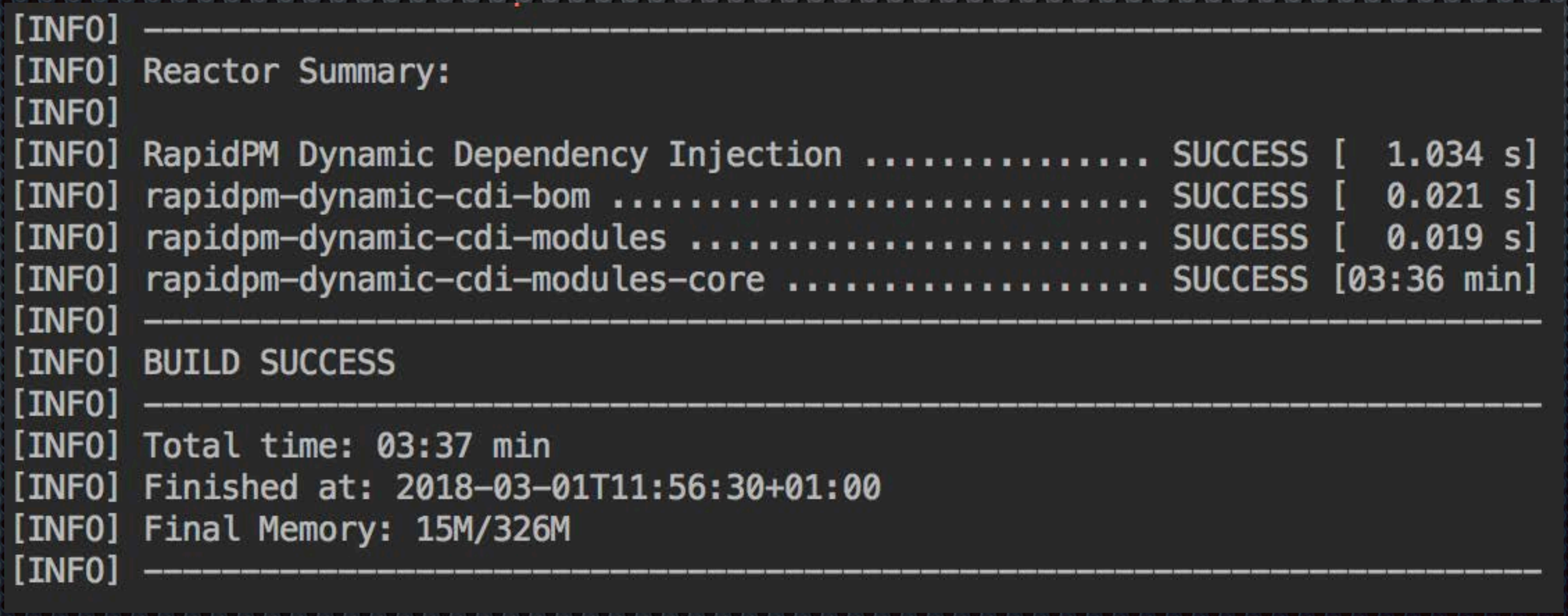
```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  1.034 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.021 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.019 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [03:36 min]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 03:37 min
[INFO] Finished at: 2018-03-01T11:56:30+01:00
[INFO] Final Memory: 15M/326M
[INFO] ------------------------------------------------------------------------
```

Mutation Testing - Real Code Examples

@SvenRuppert



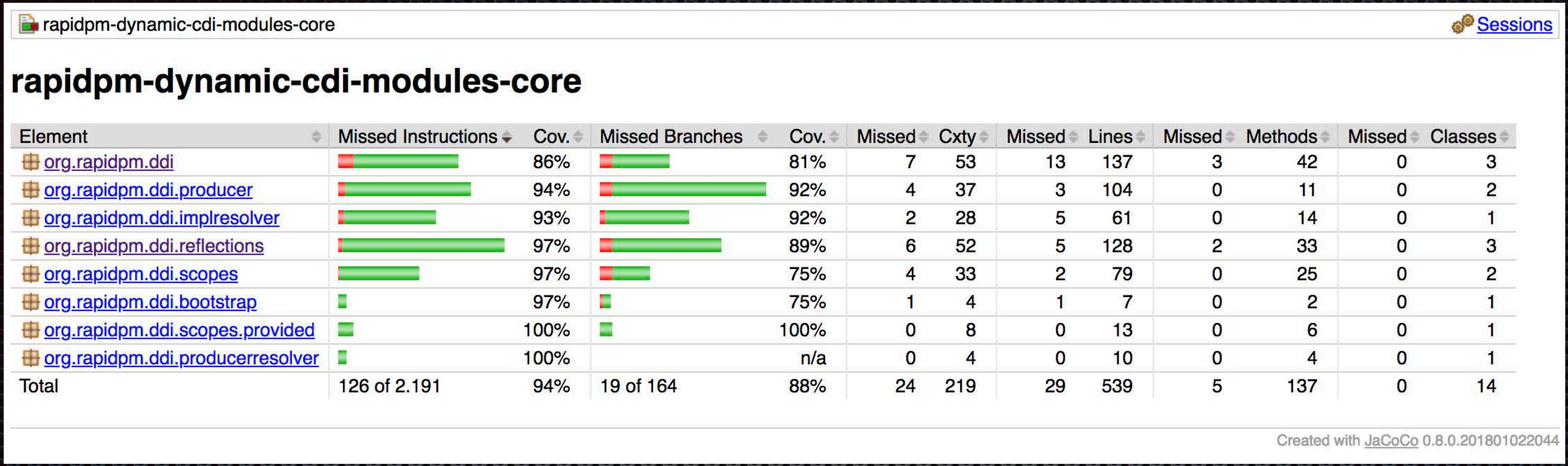
```
Tests run: 86, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  3.117 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.623 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.557 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [ 39.017 s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 43.526 s
[INFO] Finished at: 2018-03-01T12:09:33+01:00
[INFO] Final Memory: 29M/656M
[INFO] ------------------------------------------------------------------------
```



```
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] RapidPM Dynamic Dependency Injection ............... SUCCESS [  1.034 s]
[INFO] rapidpm-dynamic-cdi-bom ............................ SUCCESS [  0.021 s]
[INFO] rapidpm-dynamic-cdi-modules ........................ SUCCESS [  0.019 s]
[INFO] rapidpm-dynamic-cdi-modules-core ................... SUCCESS [03:36 min]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 03:37 min
[INFO] Finished at: 2018-03-01T11:56:30+01:00
[INFO] Final Memory: 15M/326M
[INFO] ------------------------------------------------------------------------
```

Mutation Testing - Real Code Examples

@SvenRuppert

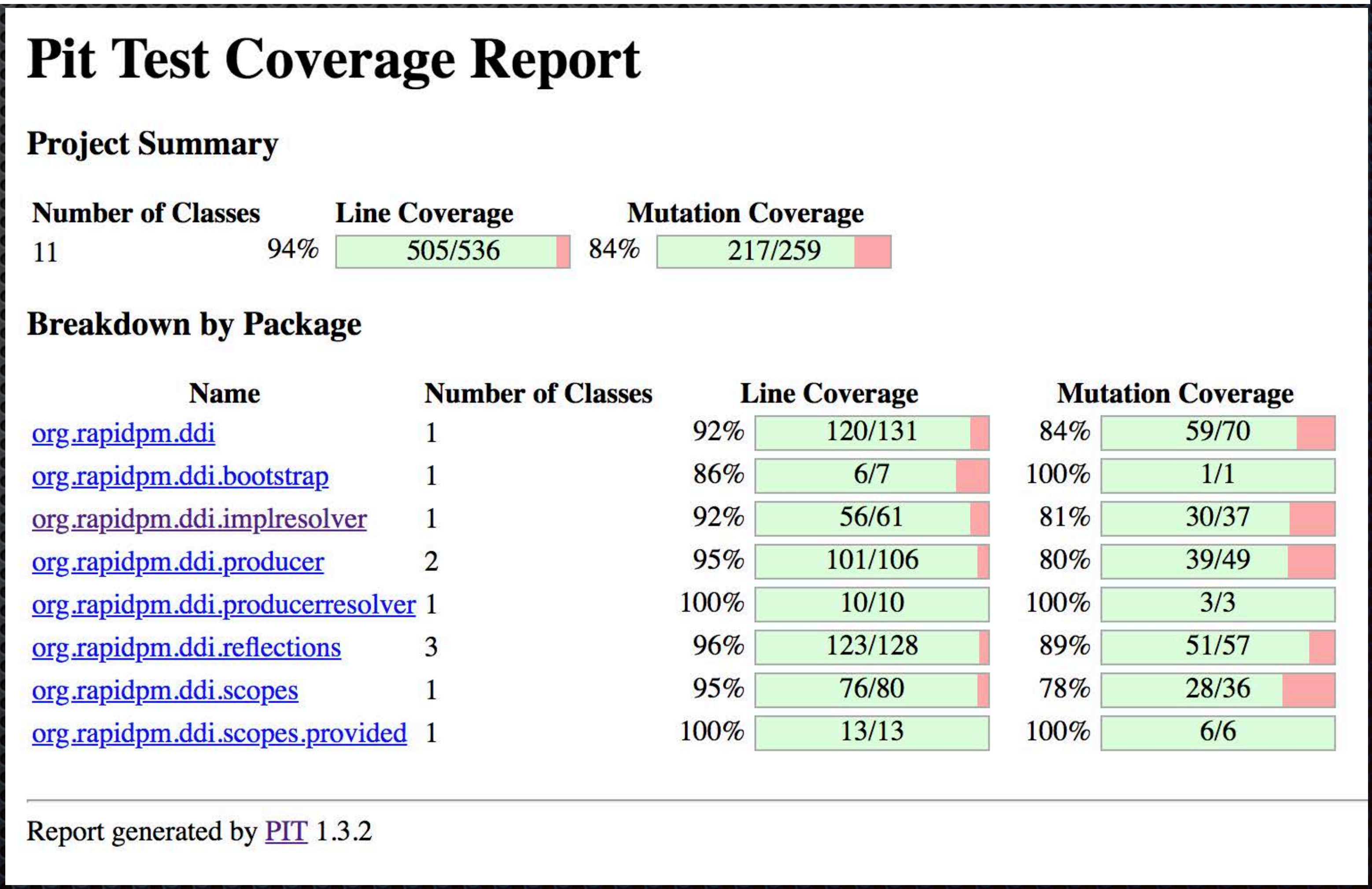


rapidpm-dynamic-cdi-modules-core — Sessions

rapidpm-dynamic-cdi-modules-core

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.rapidpm.ddi		86%		81%	7	53	13	137	3	42	0	3
org.rapidpm.ddi.producer		94%		92%	4	37	3	104	0	11	0	2
org.rapidpm.ddi.implresolver		93%		92%	2	28	5	61	0	14	0	1
org.rapidpm.ddi.reflections		97%		89%	6	52	5	128	2	33	0	3
org.rapidpm.ddi.scopes		97%		75%	4	33	2	79	0	25	0	2
org.rapidpm.ddi.bootstrap		97%		75%	1	4	1	7	0	2	0	1
org.rapidpm.ddi.scopes.provided		100%		100%	0	8	0	13	0	6	0	1
org.rapidpm.ddi.producerresolver		100%		n/a	0	4	0	10	0	4	0	1
Total	126 of 2.191	94%	19 of 164	88%	24	219	29	539	5	137	0	14

Created with JaCoCo 0.8.0.201801022044



Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
11	94%	505/536	84%	217/259

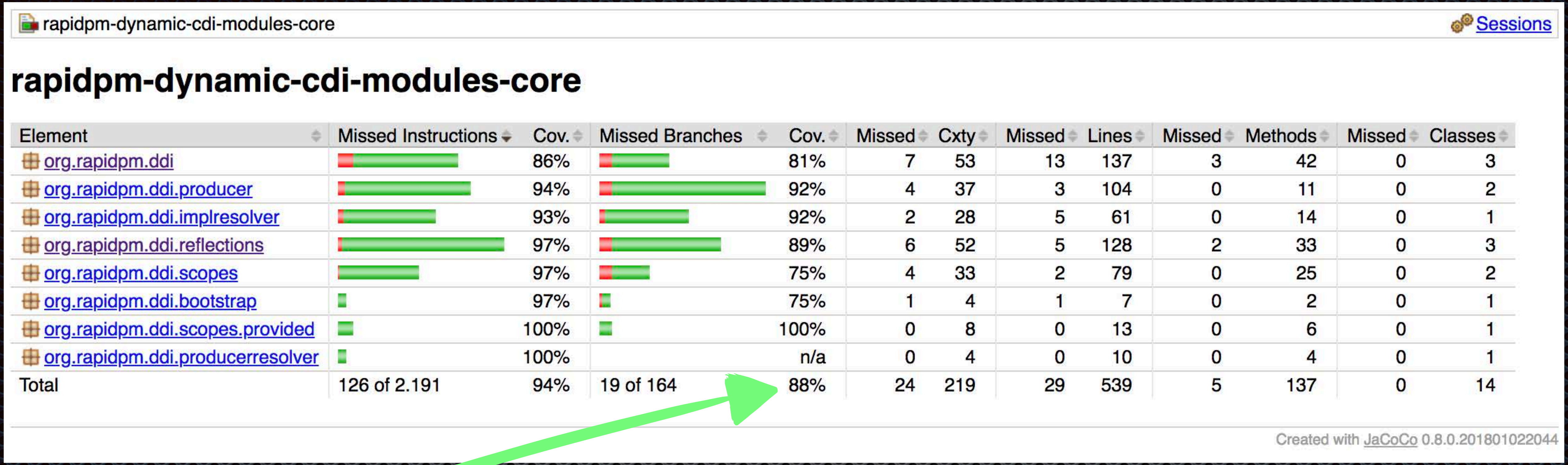
Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.rapidpm.ddi	1	92%	120/131	84%	59/70
org.rapidpm.ddi.bootstrap	1	86%	6/7	100%	1/1
org.rapidpm.ddi.implresolver	1	92%	56/61	81%	30/37
org.rapidpm.ddi.producer	2	95%	101/106	80%	39/49
org.rapidpm.ddi.producerresolver	1	100%	10/10	100%	3/3
org.rapidpm.ddi.reflections	3	96%	123/128	89%	51/57
org.rapidpm.ddi.scopes	1	95%	76/80	78%	28/36
org.rapidpm.ddi.scopes.provided	1	100%	13/13	100%	6/6

Report generated by PIT 1.3.2

Mutation Testing - Real Code Examples

@SvenRuppert

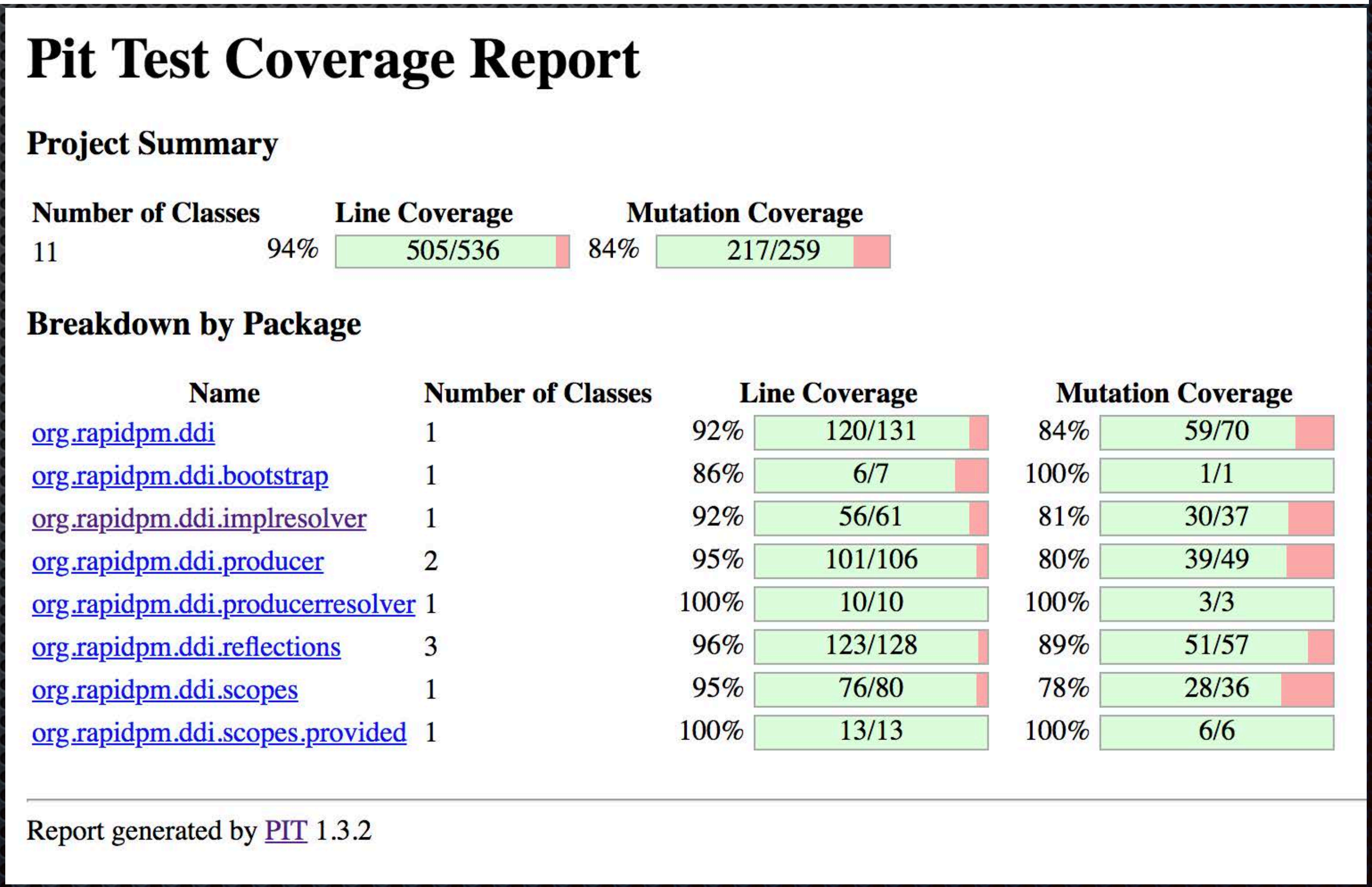


rapidpm-dynamic-cdi-modules-core | Sessions

rapidpm-dynamic-cdi-modules-core

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.rapidpm.ddi		86%		81%	7	53	13	137	3	42	0	3
org.rapidpm.ddi.producer		94%		92%	4	37	3	104	0	11	0	2
org.rapidpm.ddi.implresolver		93%		92%	2	28	5	61	0	14	0	1
org.rapidpm.ddi.reflections		97%		89%	6	52	5	128	2	33	0	3
org.rapidpm.ddi.scopes		97%		75%	4	33	2	79	0	25	0	2
org.rapidpm.ddi.bootstrap		97%		75%	1	4	1	7	0	2	0	1
org.rapidpm.ddi.scopes.provided		100%		100%	0	8	0	13	0	6	0	1
org.rapidpm.ddi.producerresolver		100%		n/a	0	4	0	10	0	4	0	1
Total	126 of 2.191	94%	19 of 164	88%	24	219	29	539	5	137	0	14

Created with JaCoCo 0.8.0.201801022044



Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
11	94%	505/536	84%	217/259

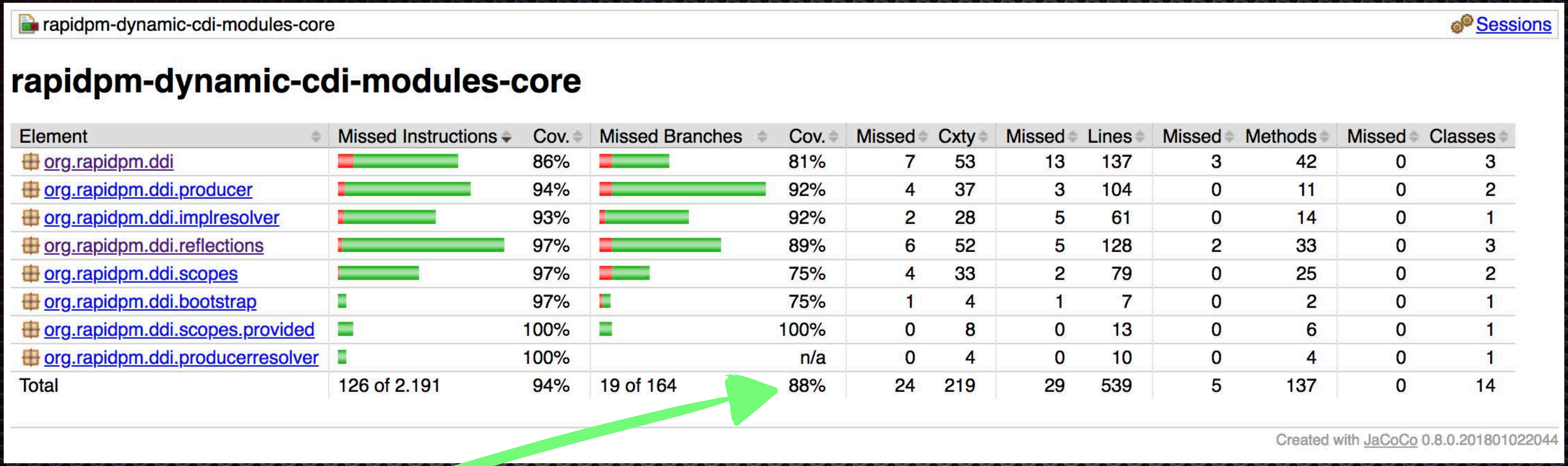
Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.rapidpm.ddi	1	92%	120/131	84%	59/70
org.rapidpm.ddi.bootstrap	1	86%	6/7	100%	1/1
org.rapidpm.ddi.implresolver	1	92%	56/61	81%	30/37
org.rapidpm.ddi.producer	2	95%	101/106	80%	39/49
org.rapidpm.ddi.producerresolver	1	100%	10/10	100%	3/3
org.rapidpm.ddi.reflections	3	96%	123/128	89%	51/57
org.rapidpm.ddi.scopes	1	95%	76/80	78%	28/36
org.rapidpm.ddi.scopes.provided	1	100%	13/13	100%	6/6

Report generated by PIT 1.3.2

Mutation Testing - Real Code Examples

@SvenRuppert



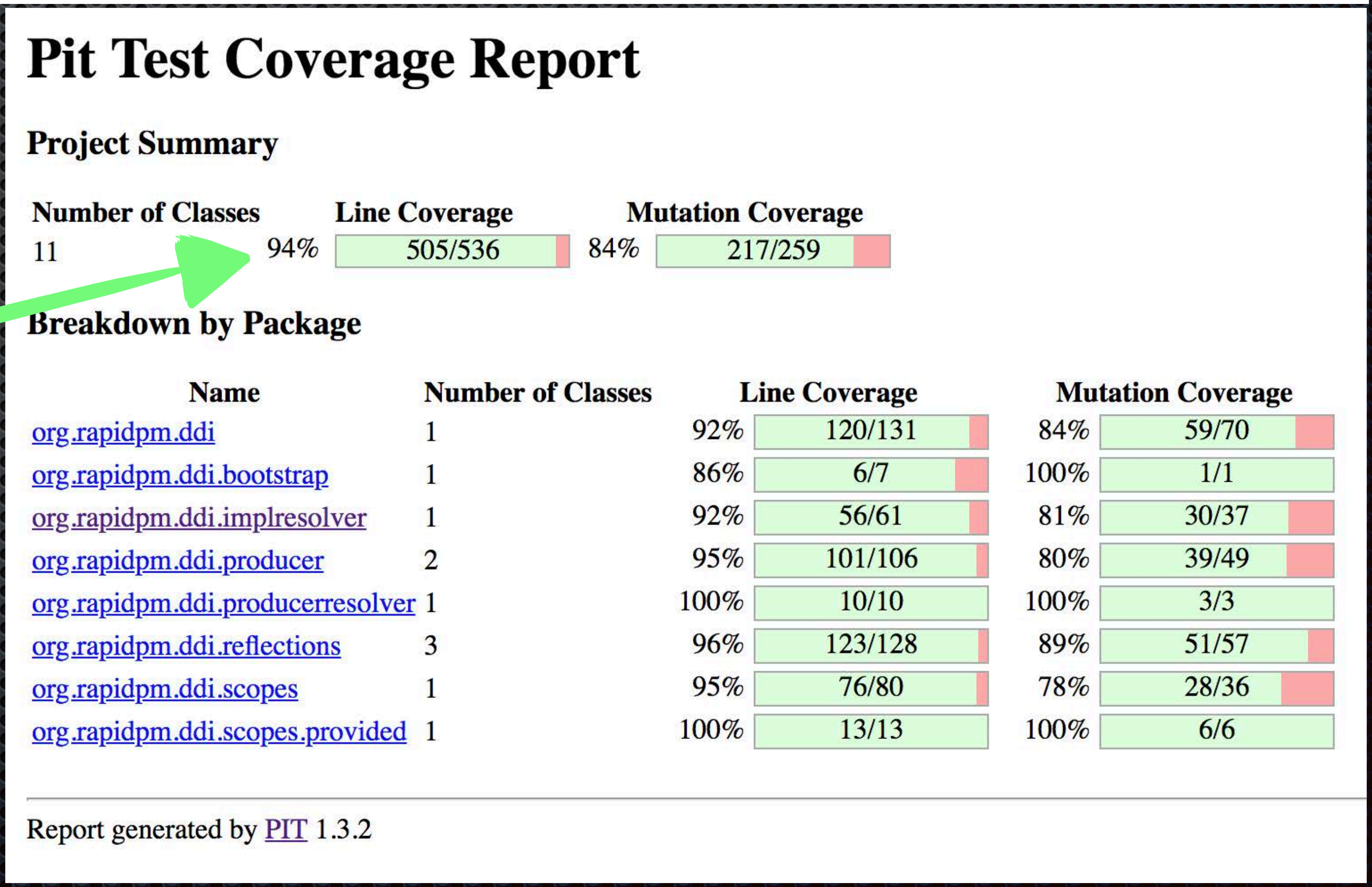
rapidpm-dynamic-cdi-modules-core

Sessions

rapidpm-dynamic-cdi-modules-core

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.rapidpm.ddi		86%		81%	7	53	13	137	3	42	0	3
org.rapidpm.ddi.producer		94%		92%	4	37	3	104	0	11	0	2
org.rapidpm.ddi.implresolver		93%		92%	2	28	5	61	0	14	0	1
org.rapidpm.ddi.reflections		97%		89%	6	52	5	128	2	33	0	3
org.rapidpm.ddi.scopes		97%		75%	4	33	2	79	0	25	0	2
org.rapidpm.ddi.bootstrap		97%		75%	1	4	1	7	0	2	0	1
org.rapidpm.ddi.scopes.provided		100%		100%	0	8	0	13	0	6	0	1
org.rapidpm.ddi.producerresolver		100%		n/a	0	4	0	10	0	4	0	1
Total	126 of 2.191	94%	19 of 164	88%	24	219	29	539	5	137	0	14

Created with JaCoCo 0.8.0.201801022044



Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
11	94%	505/536	84%	217/259

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.rapidpm.ddi	1	92%	120/131	84%	59/70
org.rapidpm.ddi.bootstrap	1	86%	6/7	100%	1/1
org.rapidpm.ddi.implresolver	1	92%	56/61	81%	30/37
org.rapidpm.ddi.producer	2	95%	101/106	80%	39/49
org.rapidpm.ddi.producerresolver	1	100%	10/10	100%	3/3
org.rapidpm.ddi.reflections	3	96%	123/128	89%	51/57
org.rapidpm.ddi.scopes	1	95%	76/80	78%	28/36
org.rapidpm.ddi.scopes.provided	1	100%	13/13	100%	6/6

Report generated by PIT 1.3.2

Mutation Testing - Real Code Examples

@SvenRuppert

Summary

If you are interested…

have a look at **GITHUB**

Functional-Reactive-Lib

Dynamic-Dependency-Injection

Vaadin-Developer

or contact me ;-) @SvenRuppert

Summary

If you are interested…

have a look at **GITHUB**

Thank You !!!

Functional-Reactive-Lib

Dynamic-Dependency-Injection

Vaadin-Developer

or contact me ;-) @SvenRuppert