



Code Defenders

A Mutation Testing Game

José Miguel Rojas



Gameplay

Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Gameplay

Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers



Gameplay

Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers



Defenders



Gameplay

Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers



Defenders



Gameplay

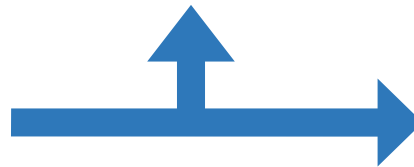
Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Defenders



Gameplay

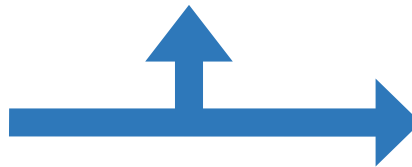
Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders



Gameplay

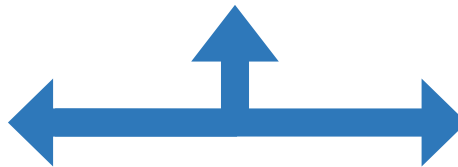
Class Under Test

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders



Gameplay

Class Under Test

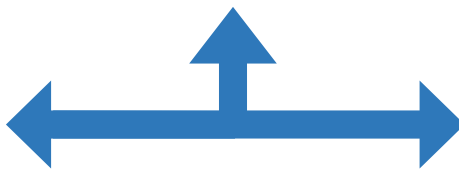
Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders



Gameplay

Class Under Test

Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



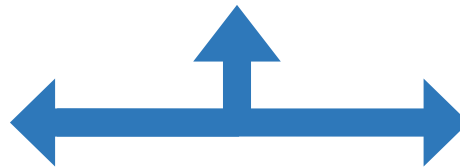
Attackers

Score points for
effective tests

```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders





Gameplay

Class Under Test

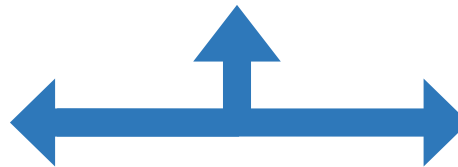
Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Score points for
effective tests

```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders

Equivalent Mutant Duels



Gameplay

Class Under Test

Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

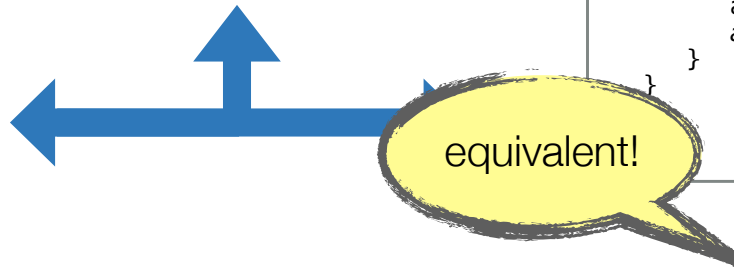
Score points for
effective tests

```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Equivalent Mutant Duels



Gameplay

Class Under Test

Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```

Score points for
effective tests

```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders

no way! here is a
killing test!

equivalent!

Equivalent Mutant Duels



Gameplay

Class Under Test

Score points for
surviving mutants

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x < 0)  
            return x;  
        else  
            return -x;  
    }  
}
```



Attackers

```
public class Arithmetics {  
    public int abs(int x) {  
        if (x >= 0)  
            return x;  
        else  
            return -x;  
    }  
}
```

Score points for
effective tests

```
public class TestArithmetics {  
    @Test  
    public void testAbs() {  
        Arithmetics a;  
        a = new Arithmetics();  
        assertEquals(1, a.abs(-1));  
    }  
}
```



Defenders

no way! here is a
killing test!

equivalent!

Equivalent Mutant Duels



Gameplay

Class Under Test

Score points for
surviving mutants

```
public class Arithmetics {  
  public int abs(int x) {  
    if (x < 0)  
      return x;  
    else  
      return -x;  
  }  
}
```



Attackers

no way! here is a
killing test!

oh no! :(

Score points for
effective tests

```
public class TestArithmetics {  
  @Test  
  public void testAbs() {  
    Arithmetics a;  
    a = new Arithmetics();  
    assertEquals(1, a.abs(-1));  
  }  
}
```



Defenders

equivalent!

Equivalent Mutant Duels



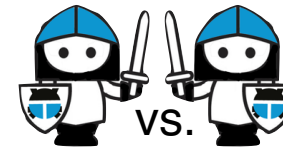
Play Modes



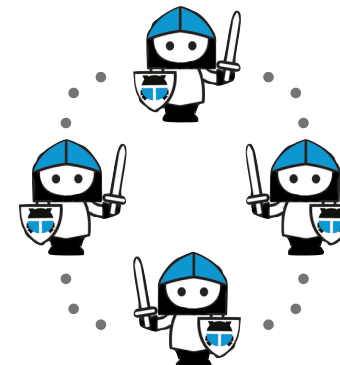
Single-player

VS.

EVASUITE / MAJOR



Two-player



Multi-player

Game 3686 (Player) x +

code-defenders.org/meleegame?gameid=3686

Code Defenders Multiplayer Puzzles shefStud

Game #3686 Mele Player 43min Scoreboard Timeline Gradle Export Feedback Editor Mode: default Chat

Create a mutant here Enemy Coverage Reset Attack Write a new JUnit test here Clone previous test Defend

```
33 public boolean isFull() {
34     return numRiders == capacity;
35 }
36
37 public void addRiders(int numEntering) {
38     if (numRiders + numEntering <= capacity) {
39         numRiders = numRiders + numEntering;
40     } else {
41         numRiders = capacity;
42     }
43 }
44
45 public void goUp() {
46     if (currentFloor < topFloor)
47         currentFloor++;
48 }
49
50 public void goDown() {
51     if (currentFloor > 0)
52         currentFloor--;
53 }
54
55 public void call(int floor) {
56     if (floor >= 0 && floor <= topFloor) {
57         while (floor != currentFloor) {
58             if (floor > currentFloor)
59                 goUp();
60             else
61                 goDown();
62         }
63     }
64 }
65 }
```

Live Killed Claimed Equivalent Equivalent

Mutant/Test restrictions: Moderate

Existing Mutants All Alive Killed Claimed Equivalent Equivalent

2 All Mutants

- Mutants outside methods
- Lift(int)
- Lift(int, int)
- getTopFloor()
- getCurrentFloor()
- getCapacity()

JUnit Tests

All Tests

- Lift(int)
- Lift(int, int)
- getTopFloor()
- getCurrentFloor()
- getCapacity()
- getNumRiders()



Code Coverage and Mutants

```
37     public void addRiders(int numEntering) {
38         if (numRiders + numEntering <= capacity) {
39             numRiders = numRiders + numEntering;
40         } else {
41             numRiders = capacity;
42         }
43     }
44
45     public void goUp() {
46         if (currentFloor < topFloor)
47             currentFloor++;
48     }
49
50     public void goDown() {
51         if (currentFloor > 0)
52             currentFloor--;
53     }
54
55     public void call(int floor) {
56         if (floor >= 0 && floor <= topFloor) {
57             while (floor != currentFloor) {
58                 if (floor > currentFloor)
59                     goUp();
60                 else
61                     goDown();
```

ayer

er

er



Code Coverage and Mutants

```
37     public void addRiders(int numEntering) {  
38         if (numRiders + numEntering <= capacity) {  
39             numRiders = numRiders + numEntering;  
40         } else {  
41             numRiders = capacity;  
42         }  
43     }
```

```
45     public void goUp() {  
46         if (currentFloor < topFloor)  
47             currentFloor++;  
48     }
```

```
49  
50     public void goDown() {  
51         if (currentFloor > 0)  
52             currentFloor--;  
53     }
```

```
54  
55     public void call(int floor) {  
56         if (floor >= 0 && floor <= topFloor) {  
57             while (floor != currentFloor) {  
58                 if (floor > currentFloor)  
59                     goUp();  
60                 else  
61                     goDown();
```

The greener the line,
the more tests that cover it!

ayer

er



Code Coverage and Mutants

```
37 public void addRiders(int numEntering) {  
38     if (numRiders + numEntering <= capacity) {  
39         numRiders = numRiders + numEntering;  
40     }
```



Mutant markers:



Live



Killed



Claimed Equivalent



Equivalent

```
45 public void goUp() {  
46     if (currentFloor < topFloor)  
47         currentFloor++;  
48 }
```



```
49  
50 public void goDown() {  
51     if (currentFloor > 0)  
52         currentFloor--;  
53 }
```

```
54  
55 public void call(int floor) {  
56     if (floor >= 0 && floor <= topFloor) {  
57         while (floor != currentFloor) {  
58             if (floor > currentFloor)  
59                 goUp();  
60             else  
61                 goDown();
```

The greener the line,
the more tests that cover it!


ayer

er



Scoreboard

Scoreboard ×

7  3

Attackers	Mutants	Alive / Killed / Equivalent	Duels Won / Lost / Ongoing	Total Points
user2	4	3 / 1 / 0	0 / 0 / 1	7
Attacking Team	4	3 / 1 / 0	0 / 0 / 1	7
Defenders	Tests	Mutants Killed	Duels Won / Lost / Ongoing	Total Points
user1	2	1	0 / 0 / 1	3
Defending Team	2	1	0 / 0 / 1	3

Close



Today's Practical – Sign up

Go to <https://code-defenders.org/>

Click on **Log in or Sign up** to create an account and sign in

You should then see this page:

The screenshot shows a web browser window with the URL `code-defenders.org/games/overview`. The page has a blue header with the Code Defenders logo, a 'Multiplayer' dropdown menu, a 'Puzzles' button, and a user profile 'shefStud'. The main content area is titled 'My Games' and contains a table with columns: ID, Creator, Class, Players, and Level. A message states 'You are currently not active in any games.' Below this are two buttons: 'Create battleground game' and 'Create melee game'. The section 'Open Battleground Games' follows, with a table showing game details. The first game has ID 135, created by sandya123, with a class link 'Lift', 44 attackers, 41 defenders, and a 'Hard' level. 'Join' buttons are present for both attacker and defender roles.

ID	Creator	Class	Attackers	Defenders	Level
> 135	sandya123	Lift	44 Join	41 Join	Hard



Today's Practical – Puzzles

Next, go to **Puzzles** and complete **as many puzzles as you can in 15 minutes** (puzzle tiles turn green when completed):

Puzzles

Class Under Test

```
1 public class Puzzle1 {
2     public int multiply(int x, int y) {
3         return x * y;
4     }
5 }
```

Write a new JUnit test here

```
1 import org.junit.Test;
2
3 import static org.junit.Assert.*;
4
5 import static org.hamcrest.MatcherAssert.assertThat;
6 import static org.hamcrest.Matchers.*;
7
8 public class PuzzleTest {
9     @Test(timeout = 4000)
10    public void test() throws Throwable {
11        // Your test here
12    }
13 }
```

Existing Mutants

All Mutants

JUnit Tests

All Tests

NEXT PUZZLE:
Beginner, Puzzle 3
Write a mutant which evades all the tests

Beginner

Puzzle 1
Write a mutant which evades all the tests

Puzzle 2
Write a test which kills the remaining mutant

Puzzle 3
Write a mutant which evades all the tests

Puzzle 4
Write a test which kills the remaining mutant

Puzzle 5
Write a mutant which evades all the tests



Today's Practical – Classroom

Next, join the Code Defenders classroom at:

<https://bit.ly/COM3529-24-CodeDefenders>

The screenshot shows a web browser window with the URL `code-defenders.org/classroom?classroomUid=66b2ac48-1343-4708-8df9-bbbc6128296f`. The page header is blue with the Code Defenders logo, navigation buttons for 'Multiplayer' and 'Puzzles', and a user profile 'shefStud'. The main content area is titled 'TUOS COM3529'. A yellow-bordered box contains the text: 'You are viewing this classroom as a guest. If you would like to join the classroom, you can do so here: [Join](#)'. A large green arrow points to the 'Join' button. Below this is a 'Games' section with tabs for 'Active' and 'Archived', a search bar, and a table with columns: Game ID, Game Mode, Your Role, State, and Link. The table is currently empty, displaying the message 'This classroom has no games... yet.' At the bottom, there are sections for 'Members' (with a search bar) and 'Join Settings'.



Today's Practical – Classroom

Once you have joined the classroom, wait to be added to a game and click on the game link when its state changes to **Running: XXmin left**.

The screenshot shows the Code Defenders website interface for a classroom named TUOS COM3529. The browser address bar shows the URL: code-defenders.org/classroom?classroomUid=66b2ac48-1343-4708-8df9-bbbc6128296f. The page has a blue header with the Code Defenders logo, navigation tabs for Multiplayer and Puzzles, and a user profile for shefStud.

TUOS COM3529

Games

Active Archived Search

Game ID	Game Mode	Your Role	State	Link
3686	Melee	Player	Not Started	Link

Members

Search

Name	Role
------	------

Join Settings

Joining is **enabled**.
Visibility is **private**.



Today's Practical – Game on!

Game 3686 (Player)

code-defenders.org/meleegame?gameId=3686

Code Defenders Multiplayer Puzzles shefStud

Game #3686 Melee Player 1h Scoreboard Timeline Gradle Export Feedback Editor Mode: default Chat 0

Create a mutant here Enemy Coverage Reset Attack

```
1 public class Lift {
2
3     private int topFloor;
4     private int currentFloor = 0; // default
5     private int capacity = 10; // default
6     private int numRiders = 0; // default
7
8     public Lift(int highestFloor) {
9         topFloor = highestFloor;
10    }
11
12    public Lift(int highestFloor, int maxRiders) {
13        this(highestFloor);
14        capacity = maxRiders;
15    }
16
17    public int getTopFloor() {
18        return topFloor;
19    }
20
21    public int getCurrentFloor() {
22        return currentFloor;
23    }
24
25    public int getCapacity() {
26        return capacity;
27    }
28
29    public int getNumRiders() {
30        return numRiders;
31    }
32
33    public boolean isFull() {
34        return numRiders == capacity;
35    }
36 }
```

Live Killed Claimed Equivalent Equivalent

Mutant/Test restrictions: Moderate

Write a new JUnit test here Clone previous test Defend

```
1 import org.junit.Test;
2
3 import static org.junit.Assert.*;
4 import static org.hamcrest.MatcherAssert.assertThat;
5 import static org.hamcrest.Matchers.*;
6
7 public class TestLift {
8     @Test(timeout = 4000)
9     public void test() throws Throwable {
10         // test here!
11     }
12 }
```

Existing Mutants All Alive Killed Claimed Equivalent Equivalent

All Mutants	▼
Mutants outside methods	▼
Lift(int)	▼
Lift(int, int)	▼

JUnit Tests

All Tests	▼
Lift(int)	▼
Lift(int, int)	▼
getTopFloor()	▼

You will play both as **attacker** (creating mutants) and as **defender** (creating tests) at the same time.
Have fun!



Lift Class

```
public class Lift {

    private int topFloor;
    private int currentFloor = 0; // default
    private int capacity = 10;    // default
    private int numRiders = 0;    // default

    public Lift(int highestFloor) {
        topFloor = highestFloor;
    }

    public Lift(int highestFloor, int maxRiders) {
        this(highestFloor);
        capacity = maxRiders;
    }

    public int getTopFloor() {
        return topFloor;
    }

    public int getCurrentFloor() {
        return currentFloor;
    }

    public int getCapacity() {
        return capacity;
    }

    public int getNumRiders() {
        return numRiders;
    }
}
```

```
    public boolean isFull() {
        return numRiders == capacity;
    }

    public void addRiders(int numEntering) {
        if (numRiders + numEntering <= capacity) {
            numRiders = numRiders + numEntering;
        } else {
            numRiders = capacity;
        }
    }

    public void goUp() {
        if (currentFloor < topFloor)
            currentFloor++;
    }

    public void goDown() {
        if (currentFloor > 0)
            currentFloor--;
    }

    public void call(int floor) {
        if (floor >= 0 && floor <= topFloor) {
            while (floor != currentFloor) {
                if (floor > currentFloor)
                    goUp();
                else
                    goDown();
            }
        }
    }
}
```



Example Test for Lift Class

```
@Test
public void testEmptyLift() {
    Lift l = new Lift(5, 10);
    l.call(3);
    assertFalse(l.isFull());
    assertEquals(3, l.getCurrentFloor());
}
```



Stack Class

```
public class Stack<T> {  
    private int capacity = 10;  
    private int pointer = 0;  
    private T[] objects = (T[]) new Object[capacity];  
  
    public void push(T o) {  
        if(pointer >= capacity)  
            throw new IllegalStateException("Stack exceeded capacity!");  
        objects[pointer++] = o;  
    }  
  
    public T pop() {  
        if(pointer <= 0)  
            throw new IllegalStateException("Stack empty");  
        return objects[--pointer];  
    }  
  
    public boolean isEmpty() {  
        return pointer <= 0;  
    }  
}
```



Example Tests for Stack

```
@Test
public void testStackException() {
    Stack<Integer> s = new Stack<Integer>();
    assertTrue(s.isEmpty());
    try {
        s.pop();
        fail("pop on an empty list should fail");
    } catch (IllegalStateException e) {
        // It's all good, exception was expected
    }
}
```

```
@Test
public void testEmptyStack() {
    Stack s = new Stack<String>();
    s.push("Hello");
    assertFalse(s.isEmpty());
}
```