Software testing

Miro Bezjak

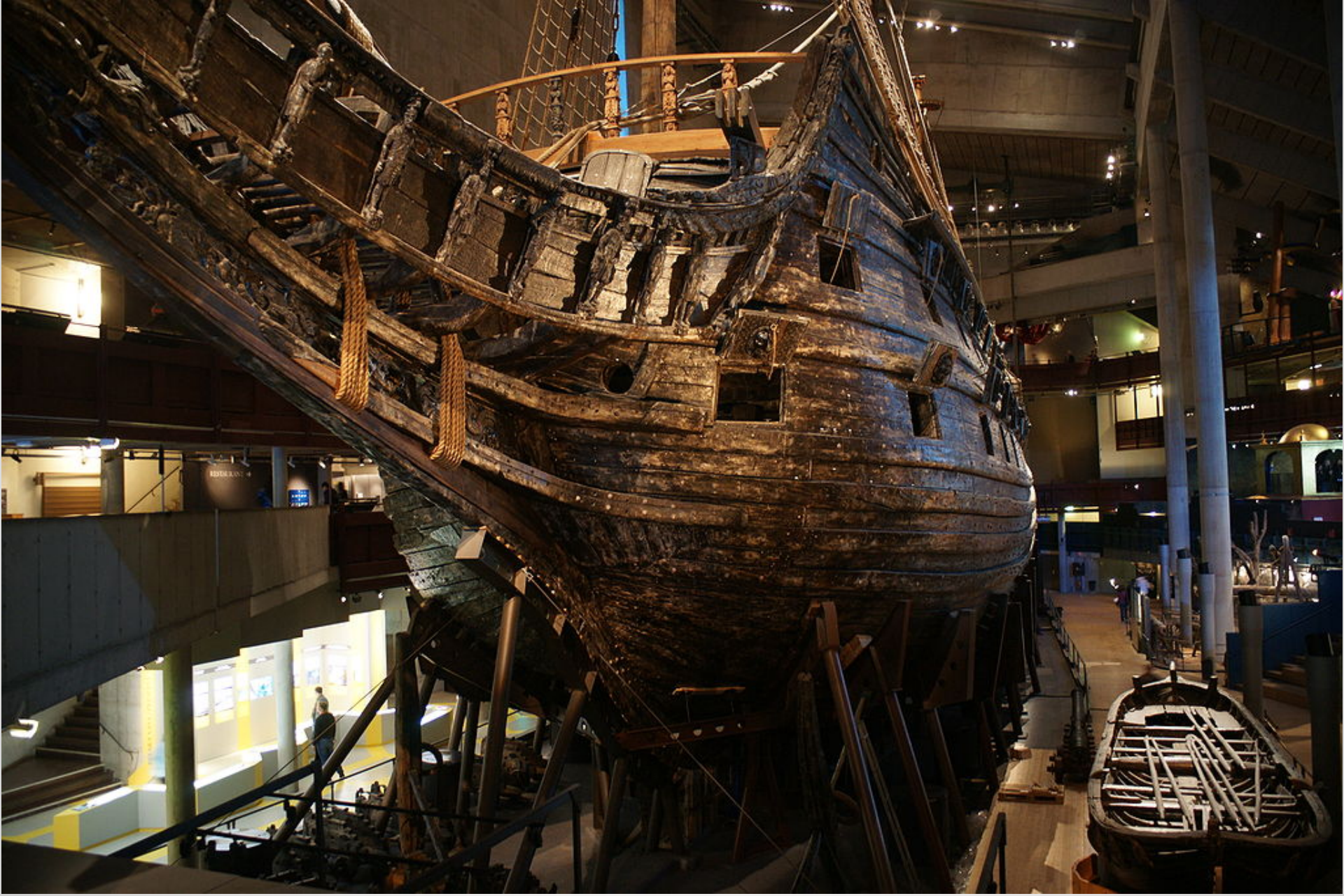
2012.

Testing?

credit: Brady Holt

credit: David Hermeyer and Sam

credit: Shakeelgilgity



[1] http://www.it.uu.se/edu/course/homepage/acsd/vt10/SE3.pdf
[2] http://faculty.up.edu/lulay/failure/vasacasestudy.pdf

credit: JavierKohen

Software testing?

Main.java

```java
package opjj;

public class Main {

    public static void main(String[] args) {
        StackCalculator calc = new StackCalculator();

        System.out.println("Initial stack = " + calc.stack);
        calc.push(41);
        System.out.println("push 41; stack = " + calc.stack);
        calc.push(9);
        System.out.println("push 9; stack = " + calc.stack);

        calc.add();
        System.out.println("add; stack = " + calc.stack);
        System.out.println("result = " + calc.result());
        System.out.println("size = " + calc.size());
    }

}
```

Console

```
<terminated> Main (1) [Java Application] /usr/lib/jvm/java-7-openjdk/bin/java (Jun 12, 2012 10:09:31 PM)
Initial stack = []
push 41; stack = [41]
push 9; stack = [41, 9]
add; stack = [50]
result = 50
size = 1
```

Debug - stack-calculator/src/opjj/Main.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Java Debug

Debug

- Main (1) [Java Application]
  - opjj.Main at localhost:54694
    - Thread [main] (Suspended (breakpoint at line 14 in Main))
      - Main.main(String[]) line: 14
  - /usr/lib/jvm/java-7-openjdk/bin/java (Jun 12, 2012 10:16:15 PM)

Variables | **Breakpoints**

Name	Value
args	String[0] (id=15)
calc	StackCalculator (id=18)
stack	Stack<E> (id=20)
capacityIncrement	0
elementCount	2
elementData	Object[10] (id=31)
modCount	2

[41, 9]

Main.java

```
package opjj;

public class Main {

    public static void main(String[] args) {
        StackCalculator calc = new StackCalculator();

        System.out.println("Initial stack = " + calc.stack);
        calc.push(41);
        System.out.println("push 41; stack = " + calc.stack);
        calc.push(9);
        System.out.println("push 9; stack = " + calc.stack);

        calc.add();
        System.out.println("add; stack = " + calc.stack);
        System.out.println("result = " + calc.result());
        System.out.println("size = " + calc.size());
    }

}
```

Outline

- opjj
- Main
  - main(String[]) : void

1 : 8

VHDLLab 2.5, test

File Edit Tools Help Development

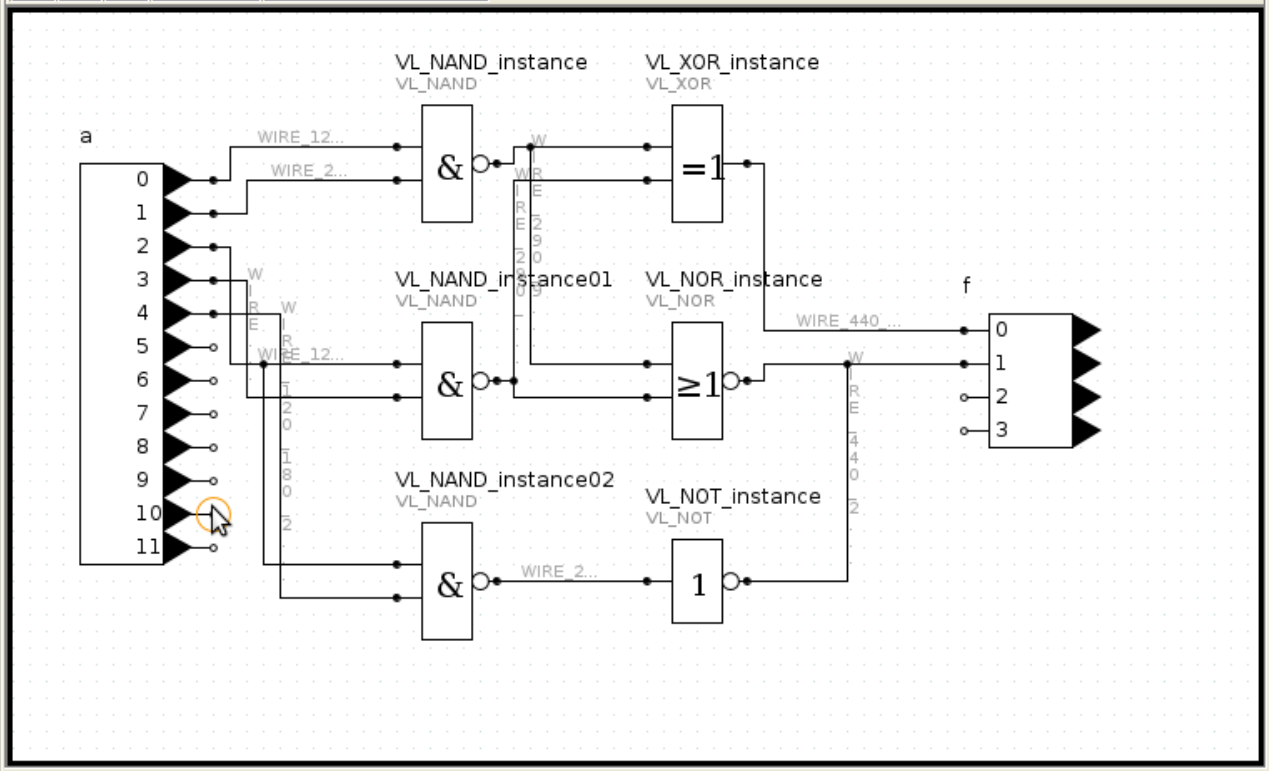
Project Explorer

- adder
  - component_or_tb
    - component_or
      - VL_OR
  - complex

Opened Editors

component_or/adder | component_or:vhdl/adder | component_or_tb/adder | *complex/adder

SNAP ON | SMART CONNECT ON



Selection: complex

Name	Value
Name	complex
Description	

UserComponents | INOUT | BasicGates

Output_vector
Input_scalar
Output_scalar
Input_vector

Log History | Compilation Results | Simulation Results

Log Hist...

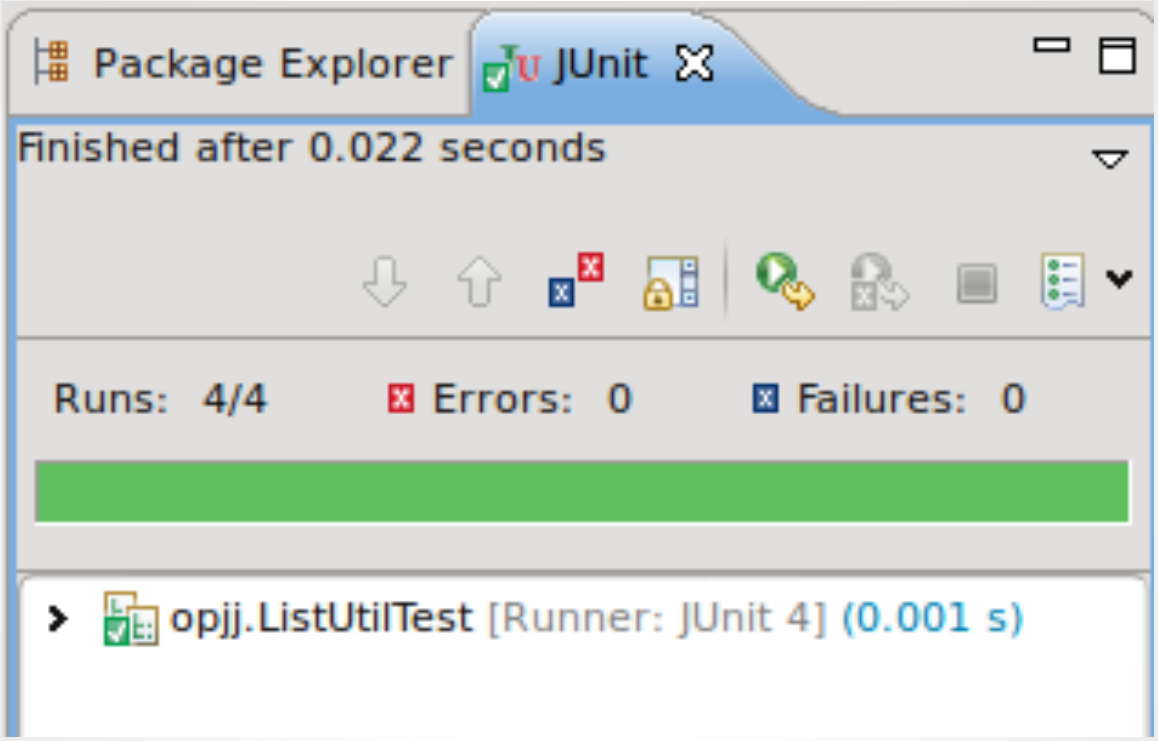
```
00:12:41  Resource component_or in project adder has been compiled
00:12:44  Editor component_or:vhdl/adder has been opened
00:13:03  Editor component_or_tb/adder has been opened
00:13:03  Resource component_or_tb in project adder has been created
00:13:13  Resource component_or_tb in project adder has been saved
00:13:19  Resource component_or_tb in project adder has been simulated
00:15:11  Editor complex/adder has been opened
00:15:11  Resource complex in project adder has been created
```

Resource complex in project adder has been created

Manual testing

- Da li je test prošao?
- Pokretanje svih testova
- Često pokretanje
- Detalji
- ...

Automation is everything


Package Explorer
JUnit
Finished after 0.022 seconds
Runs: 4/4
Errors: 0
Failures: 0
opjj.ListUtilTest [Runner: JUnit 4] (0.001 s)

Test results - Test Summ

file:///tmp/build/reports/tests/index.html

Test Summary

5	0	0.077s
tests	failures	duration

100%
successful

Packages Classes

Package	Tests	Failures	Duration	Success rate
opjj	5	0	0.077s	100%

Generated by Gradle 1.0-rc-3 at Jun 10, 2012 11:06:25 AM

scala : java

File Edit View Bookmarks Settings Help

```
[mbezjak@arch scala]$ sbt test
[info] Loading project definition from /tmp/scala/project
[info] Set current project to reverse-list-scala (in build file:/tmp/scala/)
[info] ListUtilSpec:
[info] reverse of empty list
[info] - should be an empty list
[info] singleton list
[info] - should already be reversed
[info] - should be capable of reversing a list of integers
[info] - should be capable of reversing a list of strings
[info] Passed: : Total 4, Failed 0, Errors 0, Passed 4, Skipped 0
[success] Total time: 1 s, completed Jun 10, 2012 11:31:38 AM
[mbezjak@arch scala]$
```

Automated testing

- Da li je test prošao? DA / NE
- Pokretanje svih testova odjednom
- Često pokretanje
- Detalji samo kada test padne
- Continuous integration
- ...

I dalje moguće

- System.out.println();
- Debugger

Bitno?

Testiranje

"Why is everyone in such a rush? Walk into any bookstore, and you'll see how to *Teach Yourself Java in 7 Days* [...]" [1]

-- Peter Norvig

[1] Teach Yourself Programming in Ten Years

Primjer

List reverse

Java

```java
public class ListUtil {

    public static <T> List<T> reverse(List<T> list) {
        List<T> reversed = new ArrayList<T>(list.size());

        for (int i = list.size() - 1; i >= 0; i--) {
            reversed.add(list.get(i));
        }

        return reversed;
    }

}
```

JUnit

```
public class ListUtilTest {

    @Test
    public void reverseOfEmptyListIsEmptyList() {
        List<Integer> list = new ArrayList<Integer>();
        assertEquals(list, ListUtil.reverse(list));
    }

    @Test
    public void singletonListIsAlreadyReversed() {
        List<Integer> list = Arrays.asList(42);
        assertEquals(list, ListUtil.reverse(list));
    }

    @Test
    public void reverseOfListInteger() {
        List<Integer> list     = Arrays.asList(1, 2, 3, 4, 5);
        List<Integer> expected = Arrays.asList(5, 4, 3, 2, 1);
        assertEquals(expected, ListUtil.reverse(list));
    }

}
```

Package Explorer | JUnit

Finished after 0.036 seconds

Runs: 4/4 Errors: 0 Failures: 3

- opjj.ListUtilTest [Runner: JUnit 4] (0.008 s)
  - reverseOfEmptyListIsEmptyList (0.000 s)
  - singletonListIsAlreadyReversed (0.003 s)
  - reverseOfListInteger (0.002 s)
  - reverseOfListString (0.002 s)

Failure Trace

```
java.lang.AssertionError: expected:<[42]> but was:<[]>
  at opjj.ListUtilTest.singletonListIsAlreadyReversed(ListUtilTest.java:24)
```

Scala

```
object ListUtil {

  def reverse[A](list : List[A]) : List[A] = list match {
    case head :: tail => reverse(tail) :+ head
    case _            => Nil
  }

}
```

ScalaTest

```
class ListUtilSpec extends FlatSpec with ShouldMatchers {

  "reverse of empty list" should "be an empty list" in {
    ListUtil.reverse(List()) should be ('empty)
  }

  "singleton list" should "already be reversed" in {
    val list = List(42)
    ListUtil.reverse(list) should equal (list)
  }

  it should "be capable of reversing a list of integers"  in {
    val list     = (1 to 5).toList
    val reversed = (5 to 1 by -1).toList

    ListUtil.reverse(list) should equal (reversed)
  }

}
```

scala : java

File Edit View Bookmarks Settings Help

```
[mbezjak@arch scala]$ sbt test
[info] Loading project definition from /tmp/scala/project
[info] Set current project to reverse-list-scala (in build file:/tmp/scala/)
[info] ListUtilSpec:
[info] reverse of empty list
[info] - should be an empty list
[info] singleton list
[info] - should already be reversed *** FAILED ***
[info]   List() did not equal List(42) (ListUtilSpec.scala:14)
[info] - should be capable of reversing a list of integers *** FAILED ***
[info]   List() did not equal List(5, 4, 3, 2, 1) (ListUtilSpec.scala:21)
[info] - should be capable of reversing a list of strings *** FAILED ***
[info]   List() did not equal List(baz, bar, foo) (ListUtilSpec.scala:28)
[error] Failed: : Total 4, Failed 3, Errors 0, Passed 1, Skipped 0
[error] Failed tests:
[error]         opjj.ListUtilSpec
[error] {file:/tmp/scala/}default-199b54/test:test: Tests unsuccessful
[error] Total time: 1 s, completed Jun 15, 2012 7:43:48 PM
[mbezjak@arch scala]$
```

Groovy

```
class ListUtil {

    static List reverse(List list) {
        list ? list[-1..0] : []
    }

}
```

Spock

```
class ListUtilSpec extends Specification {

    @Unroll
    def "reverse of #list should be #reversed"() {
        expect:
        reversed == ListUtil.reverse(list)

        where:
        list                    | reversed
        []                      | []
        [1]                     | [1]
        ['foo']                 | ['foo']
        [1, 2, 3, 4, 5]         | [5, 4, 3, 2, 1]
        ['foo', 'bar', 'baz'] | ['baz', 'bar', 'foo']
    }

}
```

Test results - Class opjj.L

file:///tmp/build/reports/tests/opjj.ListUtilSpec.html#reverse of [1, 2, 3, 4, 5] should be [5, 4, 3, 2, 1]

Class opjj.ListUtilSpec

all > opjj > ListUtilSpec

5	4	0.280s
tests	failures	duration

20%
successful

Failed tests Tests

reverse of [1, 2, 3, 4, 5] should be [5, 4, 3, 2, 1]

```
Condition not satisfied:

reversed == ListUtil.reverse(list)
|         |            |        |
|         false        |        [1, 2, 3, 4, 5]
[5, 4, 3, 2, 1]        [4, 3, 2, 1]

        at opjj.ListUtilSpec.reverse of #list should be #reversed(ListUtilSpec.groovy:11)
```

reverse of [1] should be [1]

```
java.lang.ArrayIndexOutOfBoundsException: Negative array index [-2] too large for array size 1
        at opjj.ListUtil.reverse(ListUtil.groovy:6)
        at opjj.ListUtilSpec.reverse of #list should be #reversed(ListUtilSpec.groovy:11)
```

Vrste

- Unit
- Integration
- System
- Acceptance
- Stress
- QuickCheck properties
- ...

http://en.wikipedia.org/wiki/Software_testing

Zašto unit testing?

- Jednostavan
- Brzina izvođenja
- Bolja arhitektura
- Refactoring
- Dokumentacija

xUnit

- SUnit
- JUnit
- CppUnit
- NUnit
- EUnit
- lisp-unit
- JSUnit
- PyUnit
- Test::Unit
- ...

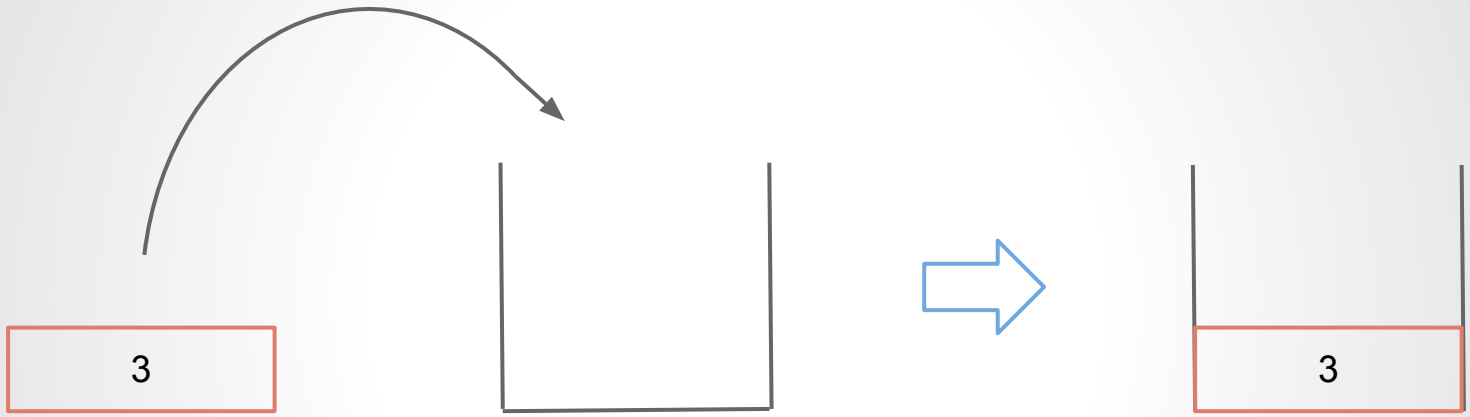
Demo

StackCalculator

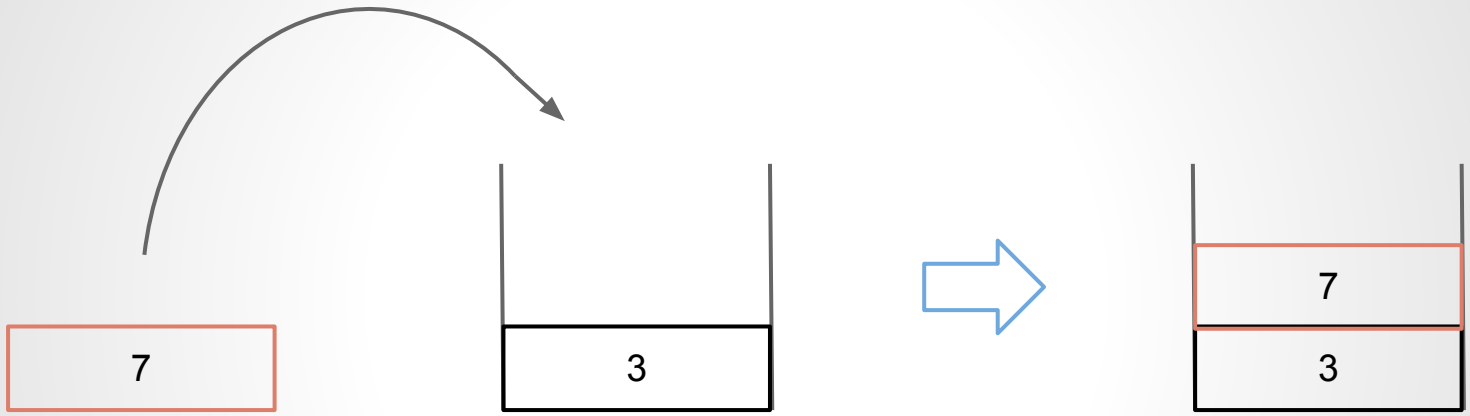
StackCalculator

```
public interface StackCalculator {

    StackCalculator push(int n);
    StackCalculator add();
    int result();
    int size();

}
```

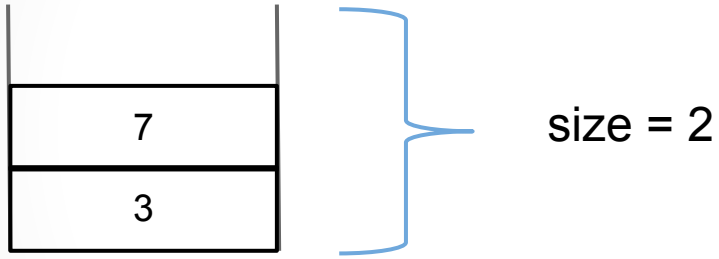
Push 3



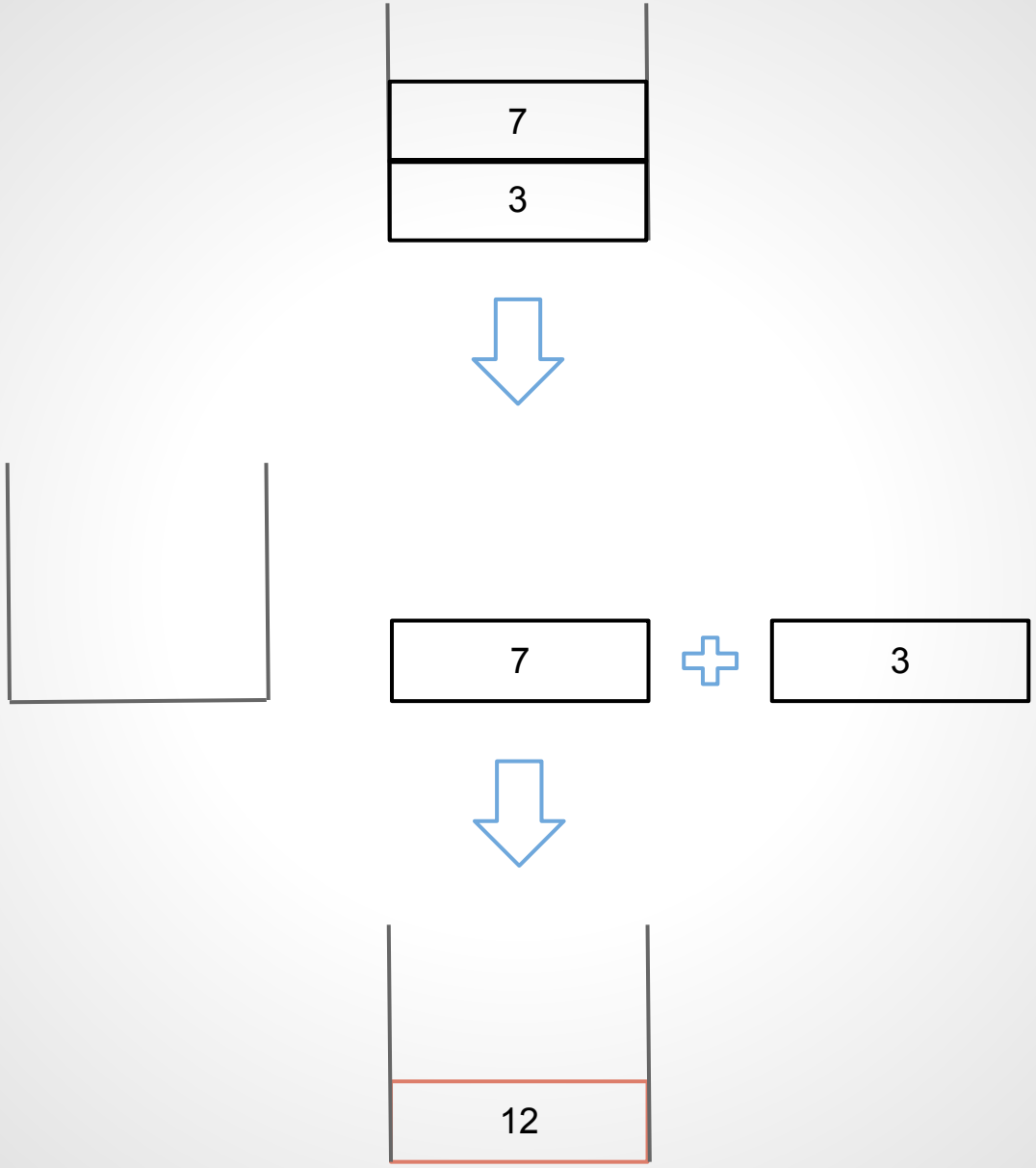
Push 7



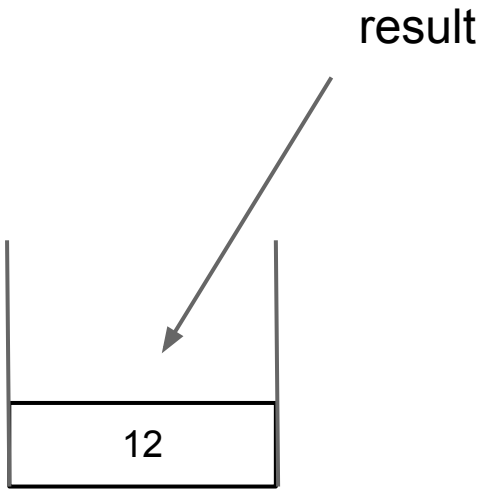
Size



Add



Result



implement().andTest();

JUnit

ClassTest

```
public class SorterTest {

    ...

}
```

@org.junit.Test

```
@Test
public void sortShouldDoThisAndThat() {

    ...
}
```

@org.junit.Test(expect=...)

```
@Test(expect = IllegalArgumentException.class)
public void sortShouldDoThisAndThat() throws Exception
{

    ...
}
```

import static org.junit.Assert.*

```
void assertEquals(String message,
                  Object expected,
                  Object actual);
```

import static org.junit.Assert.*

```
void assertSame(Object exp, Object act);
void assertNotSame(Object exp, Object act);
void assertArrayEquals(Object[] e, Object[] a);
void assertTrue(boolean condition);
void assertFalse(boolean condition);
void assertNotNull(Object object);
void assertNull(Object object);
void fail(String message);
```

@org.junit.{Before,After}

```
@Before
public void setupBeforeEachTest() { ... }

@After
public void cleanupAfterEachTest() { ... }
```

@org.junit.{BeforeClass,AfterClass}

```
@BeforeClass
public static void setupDatabase() { ... }

@AfterClass
public static void closeDatabase() { ... }
```

Ostalo

- @Test(timeout = 5000L)
- @Ignore
- @Rule
- @Theory
- @DataPoint
- assertThat
- ...

Mocks

Primjer

ACME

```java
public interface Employer {
    void accept(JobApplication application);
    void offerJob(int salary);
}
```

```java
public interface JobApplication {
    int getMinimumSalary();
    void jobOffered(int salary);
    void turnedDown();
}
```

```java
public class Acme implements Employer {

    private List<JobApplication> applications = new
                                ArrayList<JobApplication>();

    public void accept(JobApplication application) {
        applications.add(application);
    }

    public void offerJob(int salary) {
        for (JobApplication application : applications) {
            if (salary >= application.getMinimumSalary()) {
                application.jobOffered(salary);
            } else {
                application.turnedDown();
            }
        }
    }

}
```

```java
public class AcmeTest {

    private Employer acme = new Acme();
    private int acmeOffer = 90;

    @Test
    public void acmeShouldOfferAJobToEveryoneThatMeetRequirements ()
{
        JobApplication john = createMock(JobApplication. class);
        JobApplication jane = createMock(JobApplication. class);
        acme.accept(john);
        acme.accept(jane);

        expect(john.getMinimumSalary()).andReturn(100);
        expect(jane.getMinimumSalary()).andReturn(80);
        john.turnedDown();
        jane.jobOffered(acmeOffer);
        replay(john, jane);

        acme.offerJob(acmeOffer);

        verify(john, jane);
    }

}
```

Primjer

Send mail

Kako testirati?

```
class ExceptionMailerService {

    def mailService

    void mail(String description) {
        mailService.sendMail {
          to      'admins@acme.com', 'developers@acme.com'
          from    'sales.app@acme.com'
          subject 'Unrecognized exception'
          body    description
        }
    }

}
```

Kako testirati?

- unit? mock?
- integration?
- system?
- acceptance?

TDD

Test-driven development

TDD

- Način razvijanja softwarea
- Test First Design + refactoring

Ciklus

1. Dodaj test
2. Pokreni sve testove i vidi da je novi pao
3. Napravi malu promjenu
4. Pokreni sve testove i vidi da su svi prošli
5. Refactor
6. Ponovi

Primjer

Fibonacci

Fibonacci

```
fib(0) = 0
fib(1) = 1
fib(n) = fib(n-1) + fib(n-2)
```

implementUsingTDD();

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return 0;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return 0;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return n;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return n;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return n <= 1 ? n : 1;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
        assertEquals("f(3) == 2", 2, Fibonacci.calc(3));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        return n <= 1 ? n : 1;
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
        assertEquals("f(3) == 2", 2, Fibonacci.calc(3));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        if (n <= 1) {
            return n;
        } else if (n == 2) {
            return 1;
        } else {
            return 2;
        }
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
        assertEquals("f(3) == 2", 2, Fibonacci.calc(3));
        assertEquals("f(4) == 3", 3, Fibonacci.calc(4));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        if (n <= 1) {
            return n;
        } else if (n == 2) {
            return 1;
        } else {
            return 2;
        }
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
        assertEquals("f(3) == 2", 2, Fibonacci.calc(3));
        assertEquals("f(4) == 3", 3, Fibonacci.calc(4));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        if (n <= 1) {
            return n;
        } else if (n == 2) {
            return 1;
        } else {
            return n == 3 ? 2 : 3;
        }
    }
}
```

```java
public class FibonacciTest {
    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0", 0, Fibonacci.calc(0));
        assertEquals("f(1) == 1", 1, Fibonacci.calc(1));
        assertEquals("f(2) == 1", 1, Fibonacci.calc(2));
        assertEquals("f(3) == 2", 2, Fibonacci.calc(3));
        assertEquals("f(4) == 3", 3, Fibonacci.calc(4));
    }
}
```

```java
public class Fibonacci {
    public static int calc(int n) {
        if (n <= 1)
            return n;

        return calc(n - 1) + calc(n - 2);
    }
}
```

Test

```
public class FibonacciTest {

    @Test
    public void fibonacciOfN() {
        assertEquals("f(0) == 0",  0,  Fibonacci.calc(0));
        assertEquals("f(1) == 1",  1,  Fibonacci.calc(1));
        assertEquals("f(2) == 1",  1,  Fibonacci.calc(2));
        assertEquals("f(3) == 2",  2,  Fibonacci.calc(3));
        assertEquals("f(4) == 3",  3,  Fibonacci.calc(4));
        assertEquals("f(5) == 5",  5,  Fibonacci.calc(5));
        assertEquals("f(6) == 8",  8,  Fibonacci.calc(6));
        assertEquals("f(7) == 13", 13, Fibonacci.calc(7));
        assertEquals("f(8) == 21", 21, Fibonacci.calc(8));
        assertEquals("f(9) == 34", 34, Fibonacci.calc(9));
    }

}
```

Implementation

```
public class Fibonacci {

    public static int calc(int n) {
        if (n <= 1)
            return n;

        return calc(n - 1) + calc(n - 2);
    }

}
```

Demo

Slovčani iznos

Slovčani iznos

14,13 kn (slovima: ...)

```
public interface MoneyToTextConverter {

    String convert(BigDecimal money);

}
```

Demo

- 0 - 999,999
- bez deklinacija
- bez lipa

Bez deklinacije

1	jedna kuna
2	dvije kuna
42	četrdeset dvije kuna
128	jedna stotina dvadeset osam kuna
256	dvije stotina pedeset šest kuna
1024	jedna tisuća dvadeset četiri kuna

implement();

Have test?

http://tinyurl.com/opjj-mtext

implementWithTests();

Zaključak

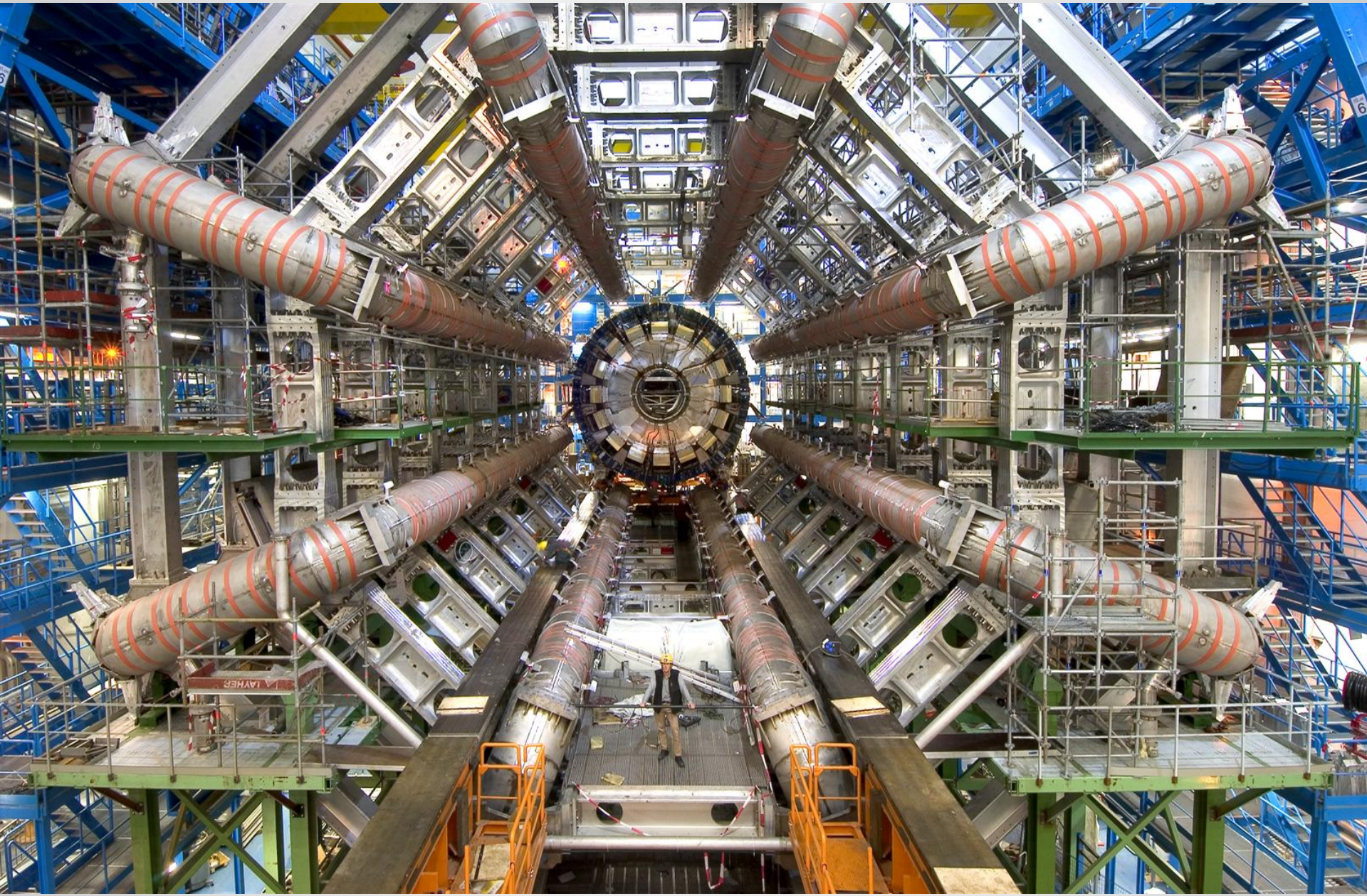
"Based on the software developer and user surveys, the national annual costs of an inadequate infrastructure for software testing is estimated to range from $22.2 to $59.5 billion." [1]

-- NIST

[1] p. 15, The Economic Impacts of Inadequate Infrastructure for Software Testing, NIST, 2002, PDF

Ime	# of tests
Groovy	5.000+
Hibernate	5.500+
Springframework	10.000+
pypy	23.000+
Firefox	? 141.000+

http://bamboo.ci.codehaus.org/browse/GROOVY-GROOVY18JDK16QUAL/latestSuccessful
http://hudson.jboss.org/hudson/view/hibernate/job/hibernate-core-master/lastCompletedBuild/testReport/
https://build.springsource.org/browse/SPR-B31X/latestSuccessful
http://buildbot.pypy.org/summary?builder=own-linux-x86-32
https://tbpl.mozilla.org/php/getParsedLog.php?id=12100401&tree=Firefox



credit: CERN

credit: pizzodisevo

Dobra strana

- Da li je test prošao? DA / NE
- Pokretanje svih testova odjednom
- Često pokretanje
- Detalji samo kada test padne
- Continuous integration
- Bolja arhitektura
- ...

Domaća zadaća

github

github.com/mbezjak/opjj-testing

DZ #1

```
factorial(0) = 1
factorial(n) = n * factorial(n-1)
```

DZ #2

Slovčani iznos

- milijun
- lipe
- deklinacija

Ispravna deklinacija

1,00	jedna kuna i nula lipa
2,00	dvije kune i nula lipa
42,00	četrdeset dvije kune i nula lipa
256,00	dvije stotine pedeset šest kuna i nula lipa
2003,00	dvije tisuće tri kune i nula lipa
15,03	petnaest kuna i tri lipe

DZ #3

```
public interface StackCalculator {

    StackCalculator minus();
    StackCalculator multiply();
    StackCalculator divide(); // integer

}
```

DZ #4

Reverse Polish notation calculator

- operacije: + - * /
- brojevi: [0-9]
- exception
- hint: StackCalculator; ~~mock~~

```
public interface ReversePolishCalculator {

    int calc(String input);

}
```

DZ #4

```
"12+"   == 3  // 2 + 1
"57*"   == 35 // 7 * 5
"34/"   == 1  // 4 / 3
"123++" == 6  // (3 + 2) + 1

"123+" // error
"1"    // ok
```

DZ #5

Caesar cipher

- ASCII
- rotiranje samo malih slova [a-z]
- svi ostali znakovi ostaju isti

```
public interface CaesarCipher {

    String encode(String input, int shift);

}
```

Shift 1

a	b	c	...	z

b	c	d	...	a

DZ #5

Dešifrirati

```
EncodedMessages.MESSAGE_1
EncodedMessages.MESSAGE_2
```

Reference

Wikipedia

- Software_bug
- Software_testing
- Software_development_process
- Test-driven_development
- XUnit

JUnit

- Homepage: junit.org
- Javadoc:

  http://junit.sourceforge.net/javadoc/index.html?org/junit/Assert.html

Članak / knjiga

- Miško Hevery, Writing Testable Code
- Scott W. Ambler, Introduction to Test Driven Development (TDD)
- Kent Beck, Test Driven Development: By Example

Zgodno

- Greg Wilson, What We Actually Know About Software Development, and Why We Believe It's True
- Bret Victor, Inventing on Principle
- Billy Hoffman, JavaScript: The Evil Parts
- Bryan O'Sullivan, Running a Startup on Haskell

?