

# Programming (ERIM)

## Lecture 6: Test-driven development

Tommi Tervonen

Econometric Institute, Erasmus School of Economics

How do you know your method works?

- Unit testing refers to automated testing of code functionality a "unit" at a time (e.g. method)
- We only test public methods (=first function of a file in Matlab)
- Not tested = doesn't work
- A single unit test tests one functionality, and tests can be grouped to test suites (usually 1 test suite with all tests)

```
function arr = sortFromIndex (array , index)
```

```
function testSortFromIndexMid()  
    arr = [3, 2, 1]  
    arr = sortFromIndex(arr, 2)  
  
    assert(arr(1) == 3)  
    assert(arr(2) == 1)  
    assert(arr(3) == 2)  
end
```

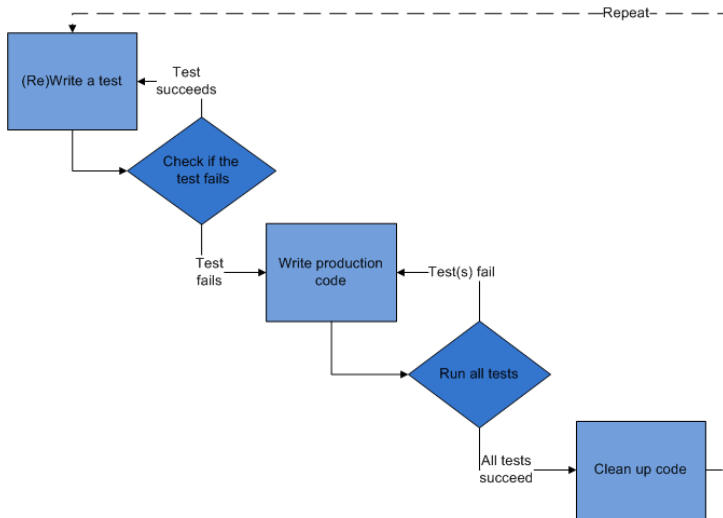
```
function testSortFromIndexAll()  
    arr = [3, 2, 1]  
    arr = sortFromIndex(arr, 1)  
  
    assert(arr(1) == 1)  
    assert(arr(2) == 2)  
    assert(arr(3) == 3)  
end
```

```
testSortFromIndexMid()  
testSortFromIndexAll()
```

# Why test?

- Unit tests document functionality
- Unit tests provide a safety net ("let me change this ... does it break something?")
- More tests = more trust in your code
- Bug = **lack of a test**
- Test-driven development

# Test-driven development



- JUnit (Java)
- testthat (R)
- matlab.unittest (Matlab 2013 onwards)

```
library(testthat)
library(rpm)

test_that("row.dominance", {
  a <- matrix(c(
    1, 1, 1,
    1, 2, 2,
    1, 2, 3), ncol=3, byrow=TRUE)
  b <- matrix(c(
    1, 1, 1,
    2, 2, 2,
    1, 2, 3), ncol=3, byrow=TRUE)
  dom <- row.dominance(a, b)
  expect_identical(dom, array(c(1, 1, 0)))
})
```