LDTS 2022/2023

Rui Maranhão

Today

- Verification and Validation
  - Reviews and Inspections
- Software Testing
  - Coverage
    - Statement, branch, and path coverage
    - Equivalence partitioning
    - Boundary analysis

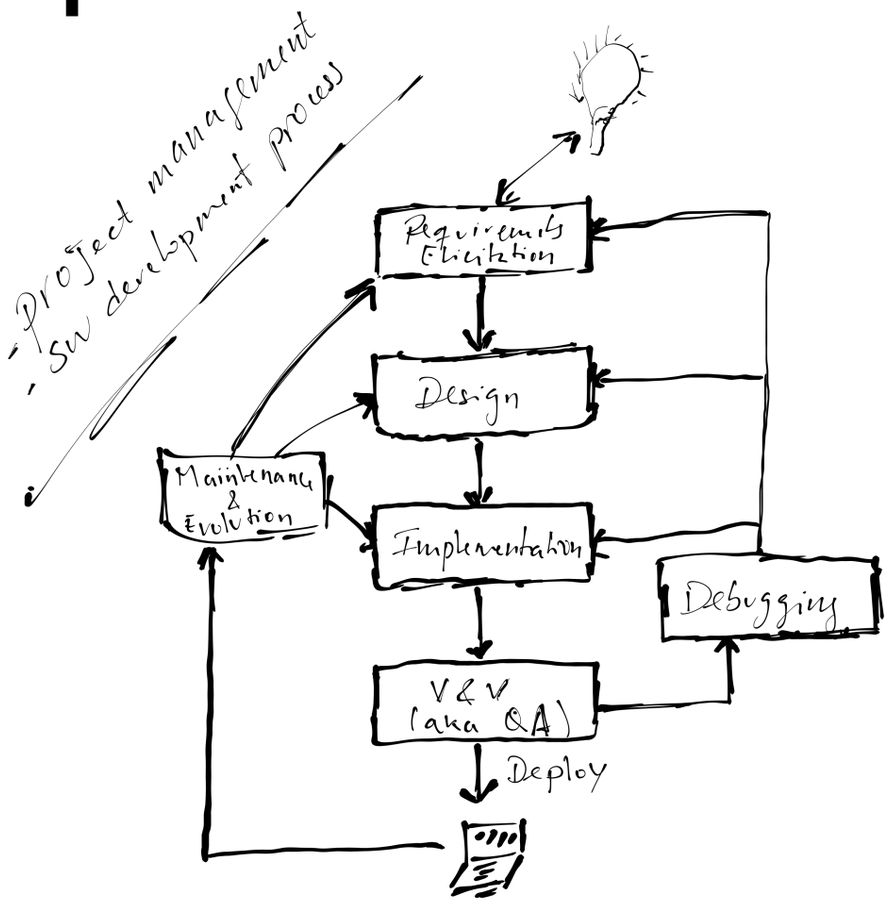
Reviews and Inspections

https://google.github.io/styleguide/javaguide.html

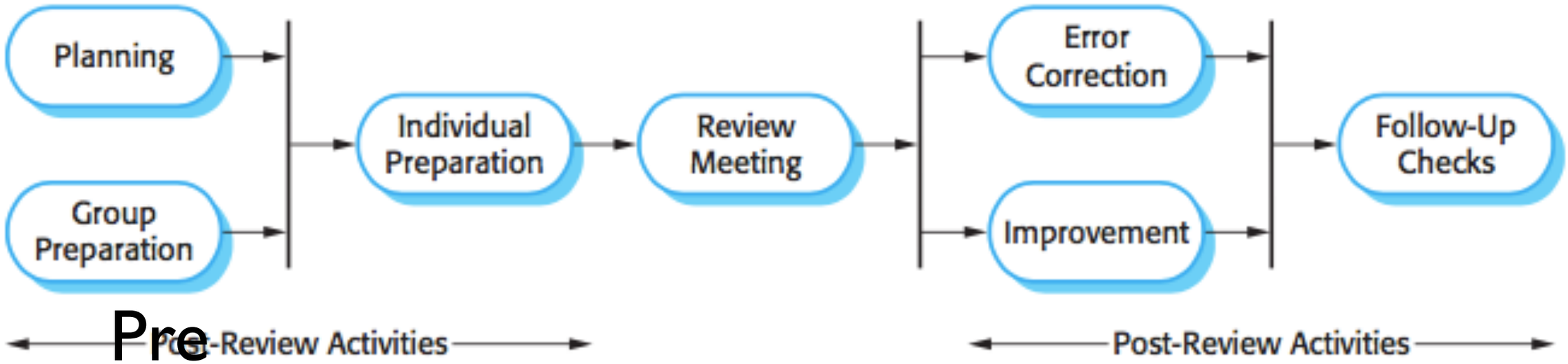
Review the quality of the description

conformance with standards

consistency and completeness



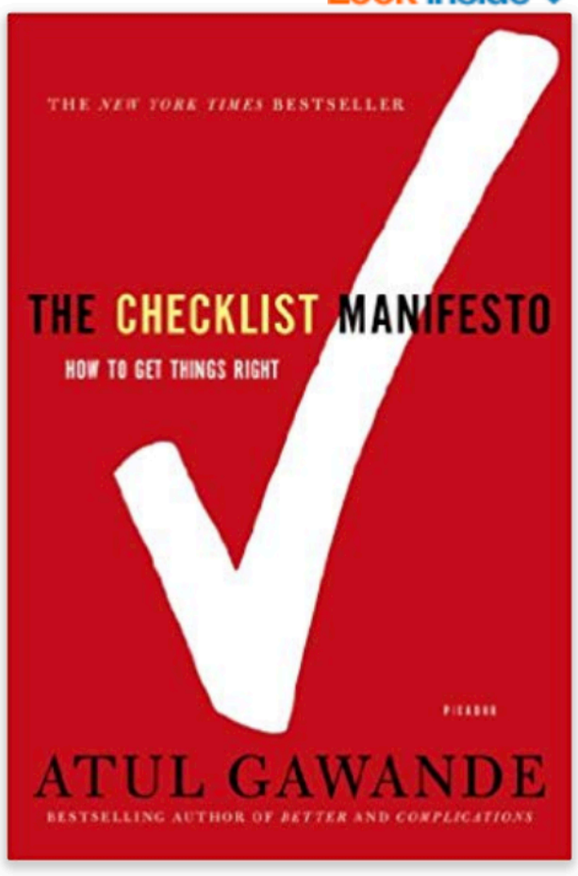
The review process



Code inspections

use a checklist of common errors

are all input variables used…



The Checklist Manifesto: How to Get Things Right Paperback – January 4, 2011

by Atul Gawande (Author)

★★★★☆ 1,410 customer reviews

#1 Best Seller in Health Care Administration

› See all 33 formats and editions

Kindle	Audiobook	Hardcover	Paperback	Audio CD
$9.78	$0.00	$17.45	**$11.98**	$23.08
Read with Our **Free App**	Free with your Audible trial	168 Used from $3.04 82 New from $5.00 2 Collectible from $94.00	171 Used from $2.98 98 New from $9.25 4 Collectible from $8.99	17 Used from $1 37 New from $19

In his latest bestseller, Atul Gawande shows what the simple idea of the checklist reveals about the complexity of our lives and how we can deal with it.

The modern world has given us stupendous know-how. Yet avoidable failures continue to plague us in

http://atulgawande.com/book/the-checklist-manifesto/

Fault class	Inspection check
Data faults	• Are all program variables initialized before their values are used? • Have all constants been named? • Should the upper bound of arrays be equal to the size of the array or Size -1? • If character strings are used, is a delimiter explicitly assigned? • Is there any possibility of buffer overflow?
Control faults	• For each conditional statement, is the condition correct? • Is each loop certain to terminate? • Are compound statements correctly bracketed? • In case statements, are all possible cases accounted for? • If a break is required after each case in case statements, has it been included?
Input/output faults	• Are all input variables used? • Are all output variables assigned a value before they are output? • Can unexpected inputs cause corruption?
Interface faults	• Do all function and method calls have the correct number of parameters? • Do formal and actual parameter types match? • Are the parameters in the right order? • If components access shared memory, do they have the same model of the shared memory structure?
Storage management faults	• If a linked structure is modified, have all links been correctly reassigned? • If dynamic storage is used, has space been allocated correctly? • Is space explicitly deallocated after it is no longer required?
Exception management faults	• Have all possible error conditions been taken into account?

Detect >60% errors

(Fagan, 86)

Use static analysis tools while coding

https://github.com/google/error-prone
https://fbinfer.com/

IntelliJ plugin: SonarQube, SonarLint

Testing

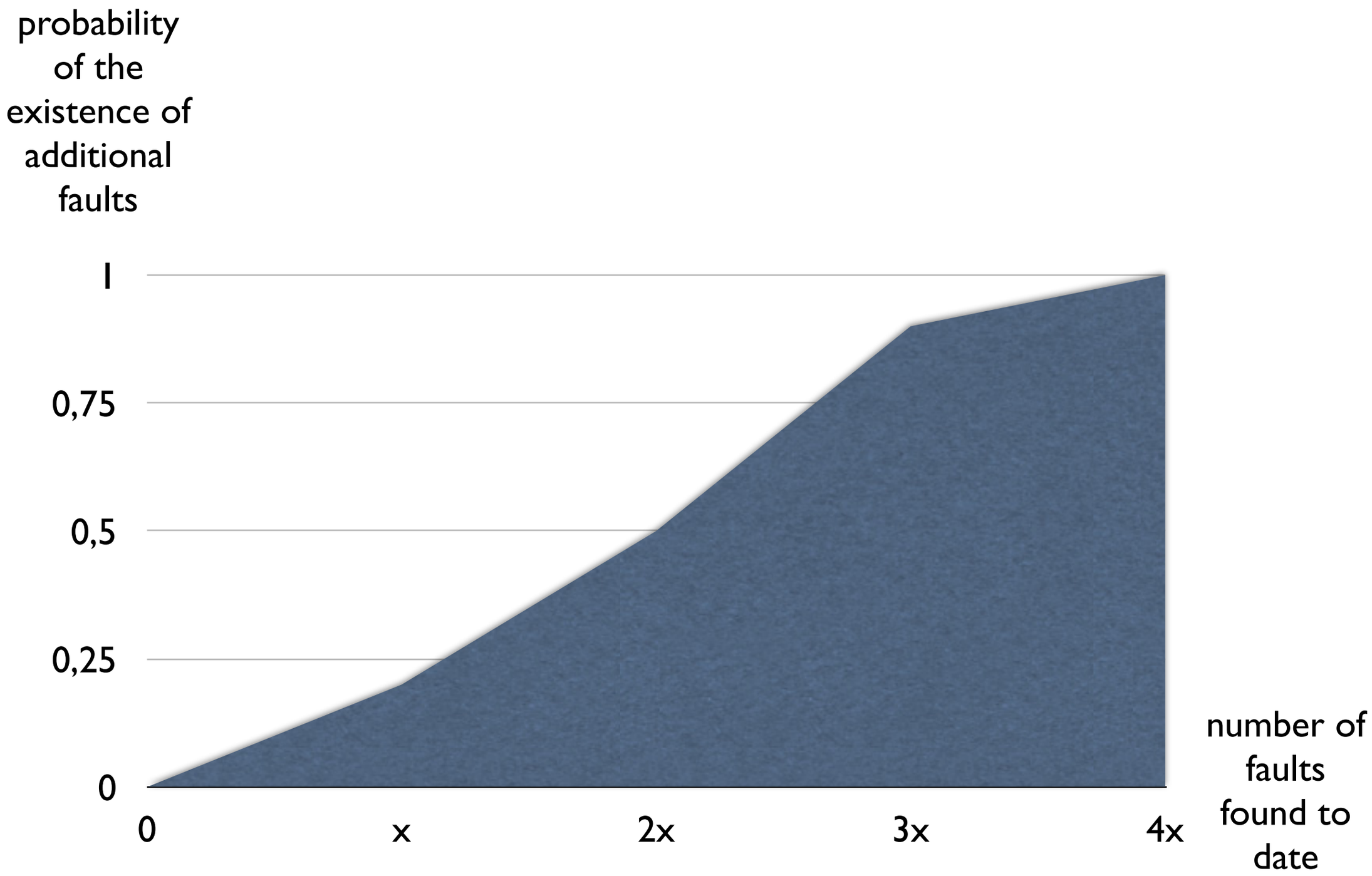
Who is involved in testing ?

- Developers
- QA engineers
- Customers

Advantages and Disadvantages ?

When to stop testing ?

When is “enough”, enough ?


probability of the existence of additional faults
1
0,75
0,5
0,25
0
0
x
2x
3x
4x
number of faults found to date

Fault injection

$$\frac{\text{Detected injected faults}}{\text{Total injected faults}}$$

When time runs out...

Determine the extent of testing

Test strategies

Test against a specification

Operation: pop(stack)

Pre-condition: stack is not empty

which are the test cases?

Black box testing

Operation: pop(stack)

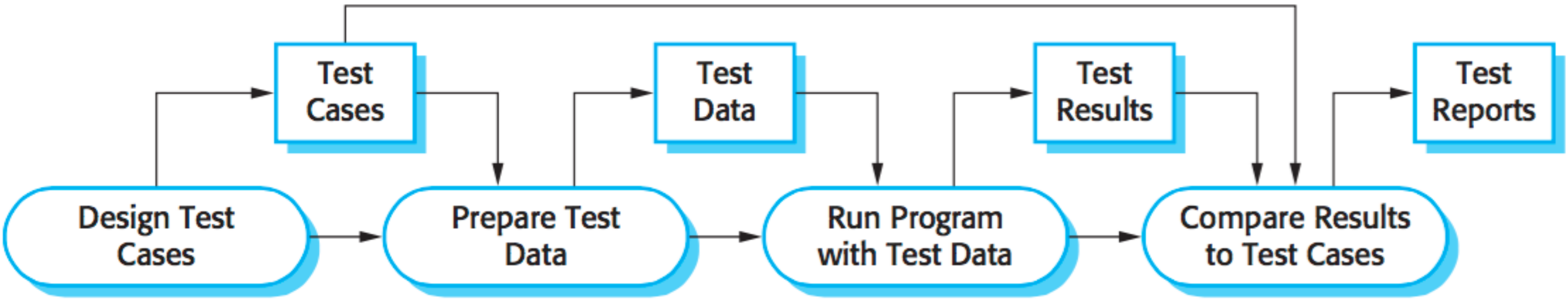
Implementation: stack is implemented by a set of ordered arrays, 1000 elements each

which are the test cases?

Test against a design or implementation

White box testing

Software testing process


Test Cases
Test Data
Test Results
Test Reports
Design Test Cases
Prepare Test Data
Run Program with Test Data
Compare Results to Test Cases

Can we automate it?

JUnit/Spock

part of it

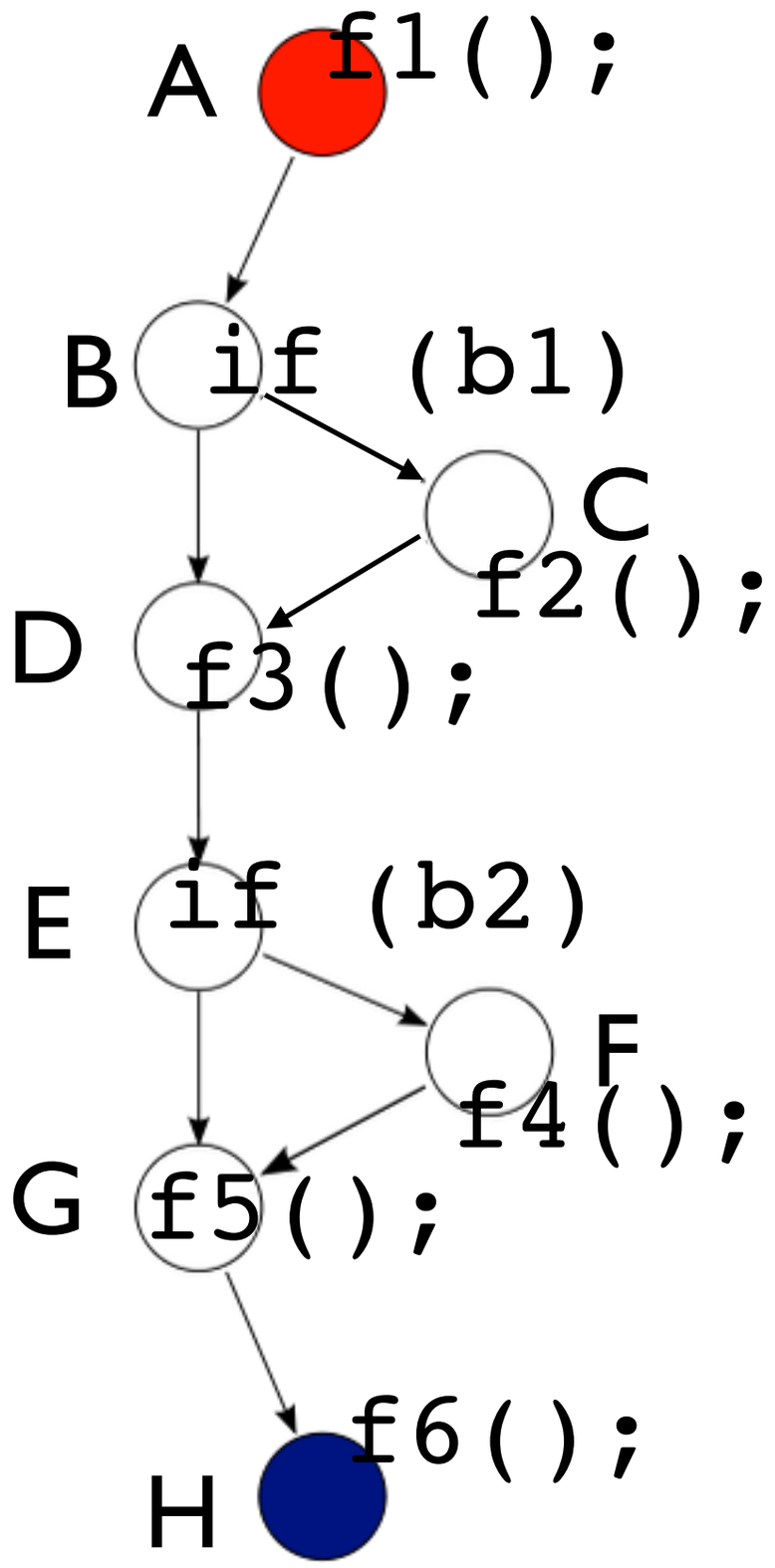
Test coverage

- white box testing
  - statement coverage
  - branch coverage
  - path coverage
- black box testing
  - equivalence partitioning
  - boundary value analysis

White-box Coverage

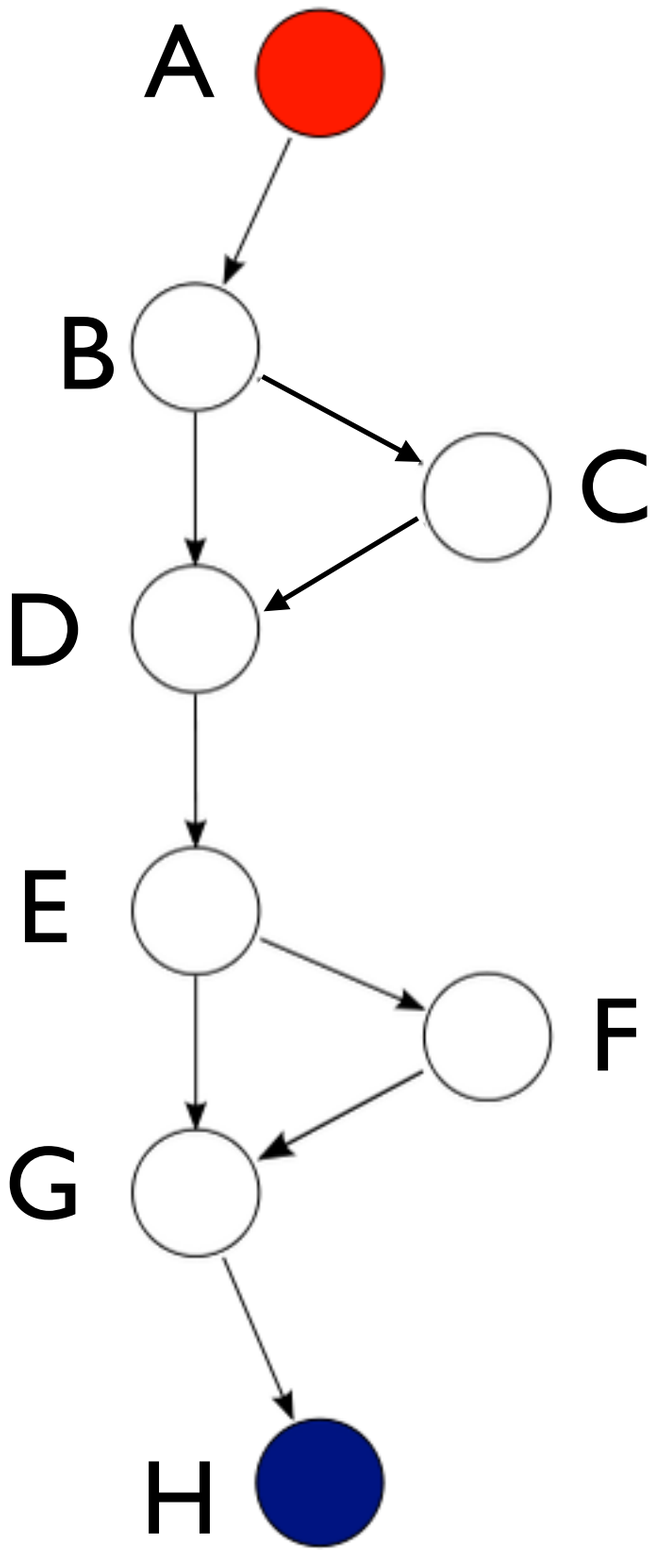
Code-aware

```
f1();
if (b1)
  f2();
f3();
if (b2)
  f4();
f5();
f6();
```


A
f1();
B
if (b1)
C
f2();
D
f3();
E
if (b2)
F
f4();
G
f5();
f6();
H

Statement Coverage

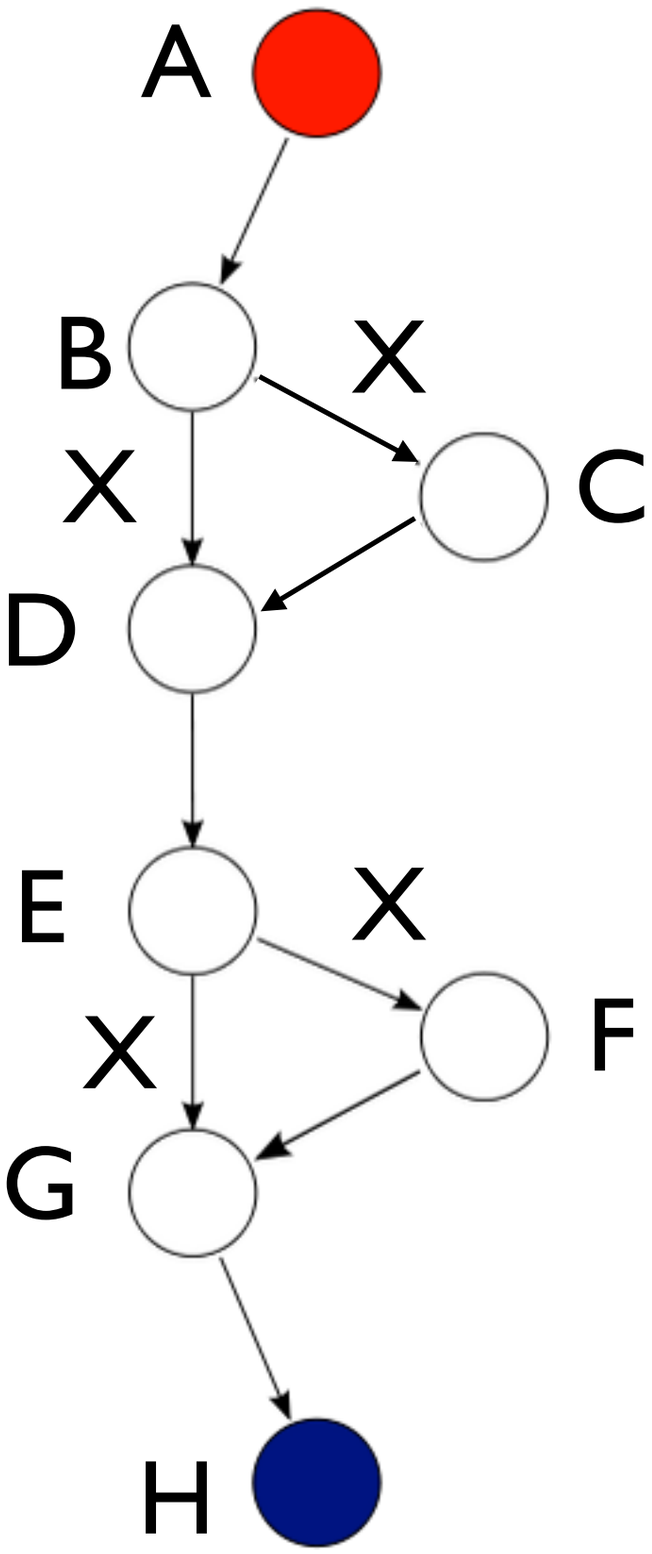
statements are executed at least once



A B C D E F G H

Branch Coverage

every condition is evaluated to true and false



A B C D E F G H

A B D E G H

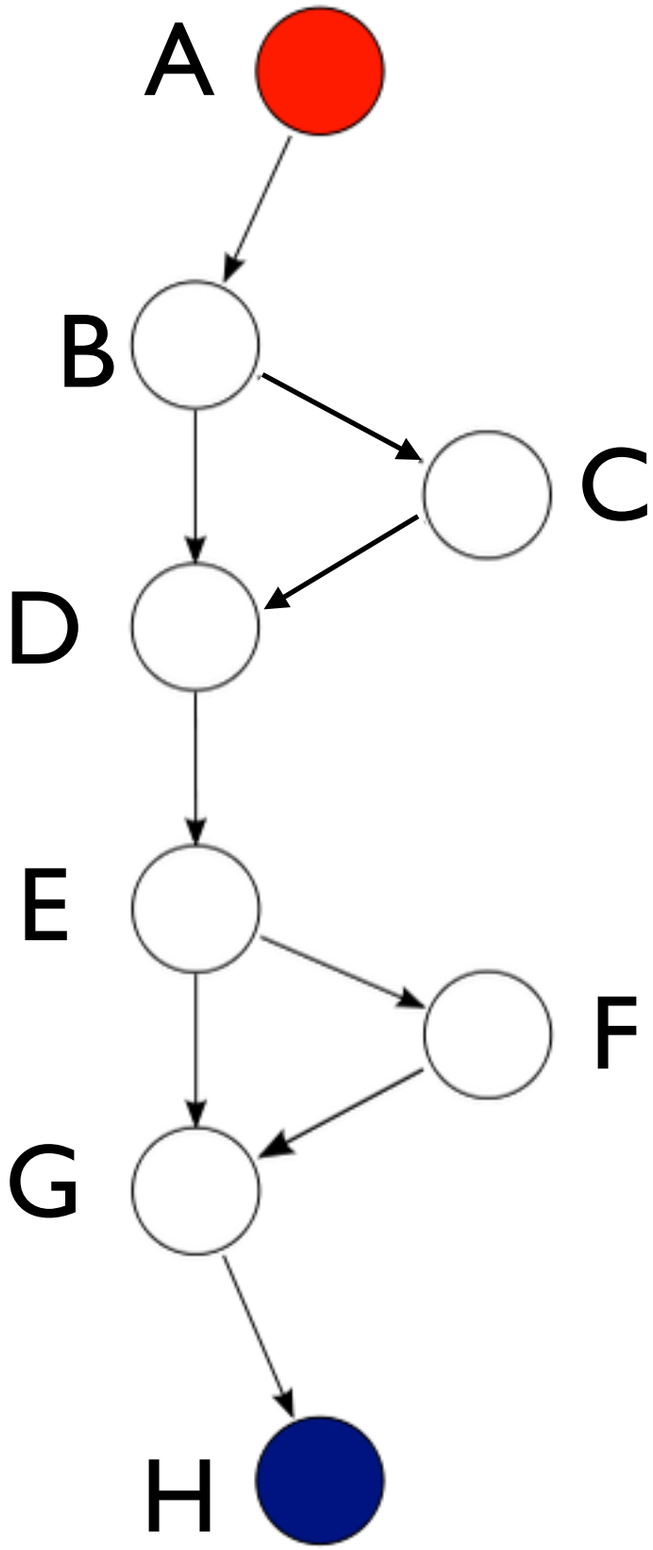
Path Coverage

every independent path is executed

an independent program path is one
that traverses at least one new edge in the flow graph

Iterative in the construction of paths:

- set all decisions to true
- set first decision to false
- set next decision to false
- stop when last decision is set to false



(all true) A B C D E F G H

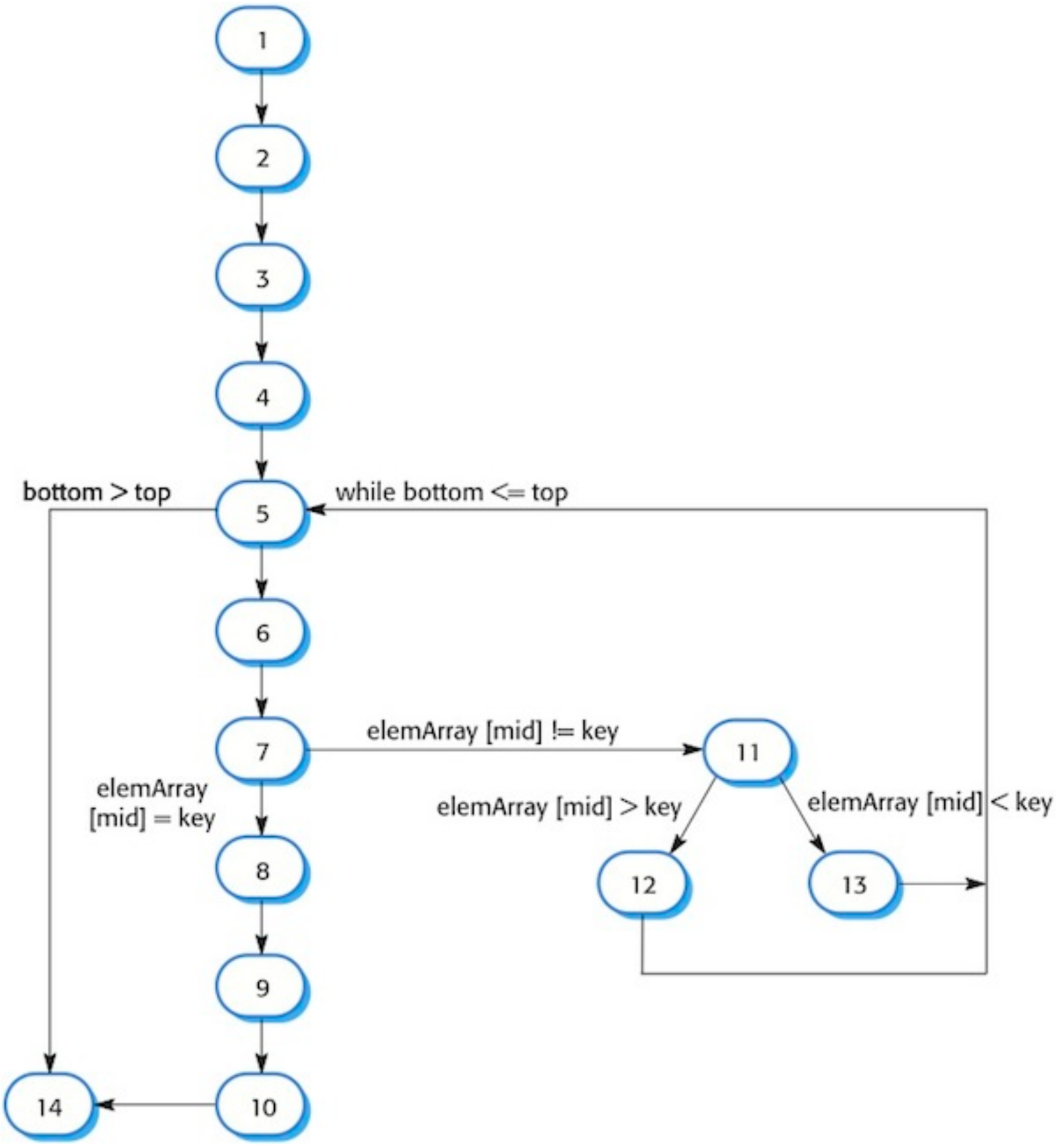
(first false) A B D E F G H

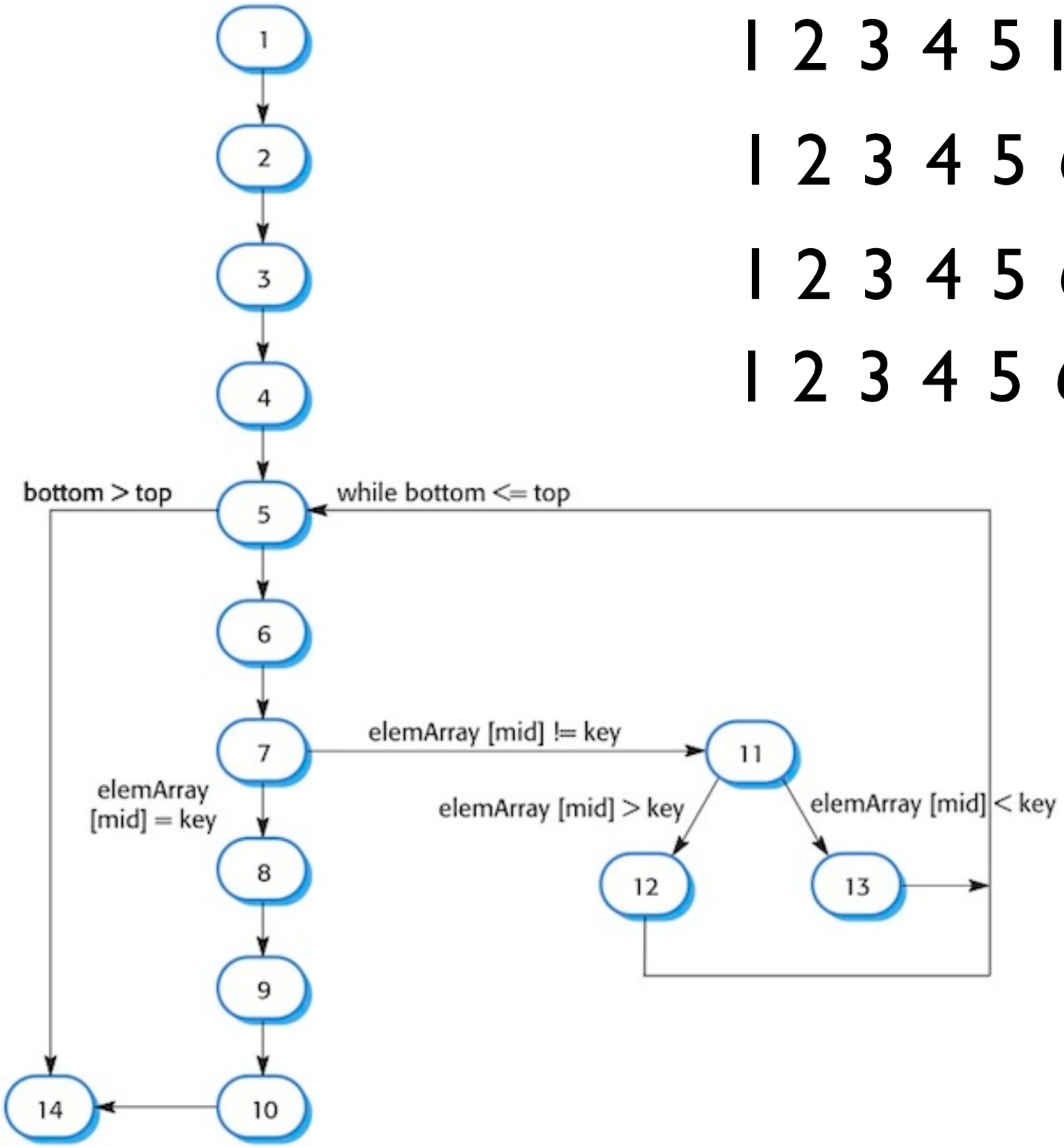
(second false) A B C D E G H

Cyclomatic Complexity

number of independent paths

number of decisions + 1


1
2
3
4
bottom > top
while bottom <= top
5
6
7
elemArray [mid] != key
11
elemArray
[mid] = key
elemArray [mid] > key
elemArray [mid] < key
8
12
13
9
14
10



1 2 3 4 5 14

1 2 3 4 5 6 7 8 9 10 14

1 2 3 4 5 6 7 11 12 5 ...

1 2 3 4 5 6 7 11 13 5 ...

Condition Coverage

if a and b then …

decision vs condition

Jacoco / codecov

```
    public void processBooking() {
        if (this.cancelled) {
            return;
        }

        try {
            this.references.addAll(HotelInterface.bulkBooking(this.number, this.arrival, this.departure));
            this.numberOfHotelExceptions = 0;
            this.numberOfRemoteErrors = 0;
            return;
        } catch (HotelException he) {
            this.numberOfHotelExceptions++;
            if (this.numberOfHotelExceptions == MAX_HOTEL_EXCEPTIONS) {
                this.cancelled = true;
            }
            this.numberOfRemoteErrors = 0;
            return;
        } catch (RemoteAccessException rae) {
            this.numberOfRemoteErrors++;
            if (this.numberOfRemoteErrors == MAX_REMOTE_ERRORS) {
                this.cancelled = true;
            }
            this.numberOfHotelExceptions = 0;
            return;
        }
    }
```

EclEmma plugin, http://www.eclemma.org

```
    <!-- Coverage -->
    <version.jacoco.maven.plugin>0.8.2</version.jacoco.maven.plugin>
    <coverage.class.ratio>0.15</coverage.class.ratio>
    <coverage.instruction.ratio>0.15</coverage.instruction.ratio>
    <coverage.method.ratio>0.15</coverage.method.ratio>
    <coverage.branch.ratio>0.15</coverage.branch.ratio>
    <coverage.complexity.ratio>0.15</coverage.complexity.ratio>
    <coverage.line.ratio>0.15</coverage.line.ratio>
</properties>
```

http://www.jacoco.org/jacoco/trunk/doc/counters.html

What should be the input values for the test cases?

Black-box Coverage

Consider a method

What is a method specification ?

The method signature

Only ?

for instance,

daysLate has 2^{64} values

Are all 2^{64} values semantically different?

There is some business logic

MAX_FINE_PERIOD

$$0 <= daysLate <= MAX\_FINE\_PERIOD$$

$$daysLate > MAX\_FINE\_PERIOD$$

We need more than the method signature

...from syntax to semantic...

We need a specification

```
if (daysLate <= MAX_FINE_PERIOD)
   fine = daysLate * DAILY_FINE
else
   fine = MAX_FINE
```

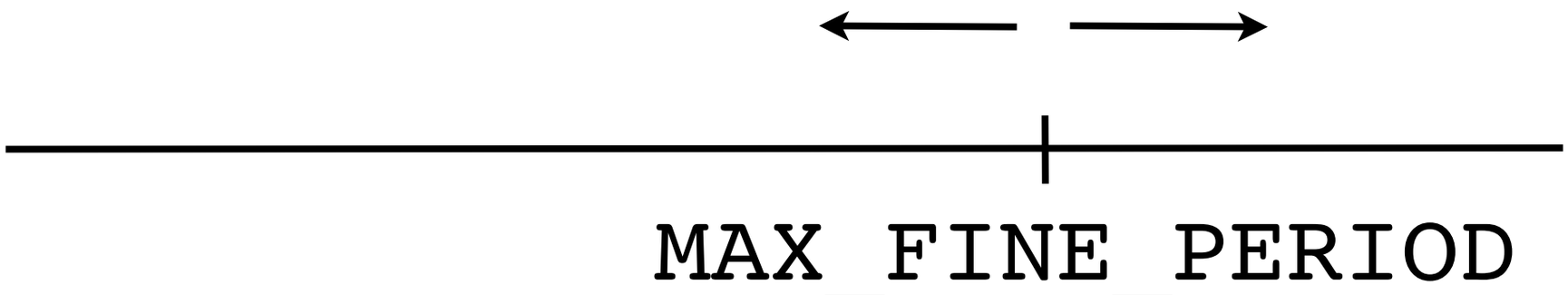
Equivalence partitioning

non overlapping sets that constitute the complete set of possibilities (partitions)

and

the values in each partition are expected to trigger a similar behavior

test values for `daysLate`

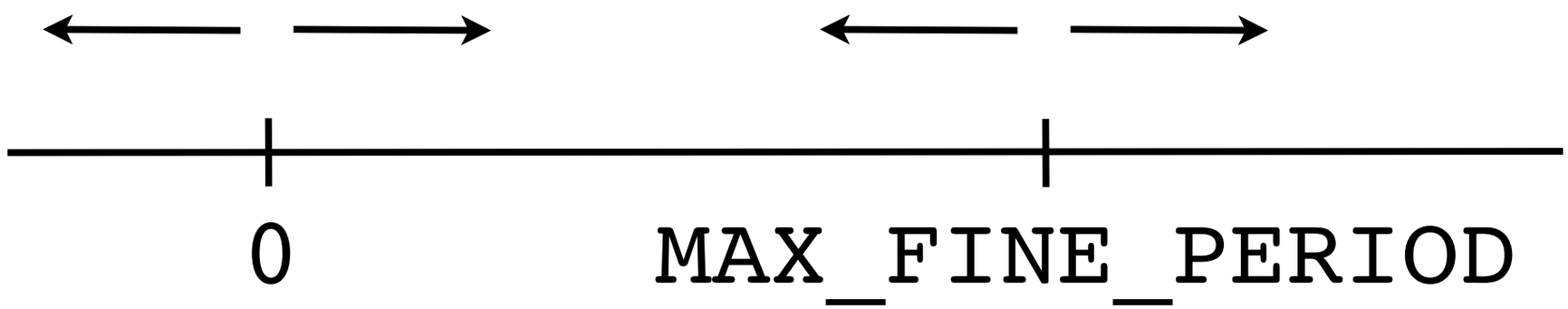


What if

`daysLate < 0`?

is it part of the specification?

test values for `daysLate`



Is it enough to test an element per equivalence partition?

What can happen in the boundaries between equivalence partitions?

Boundary value analysis

Boundary Value Analysis

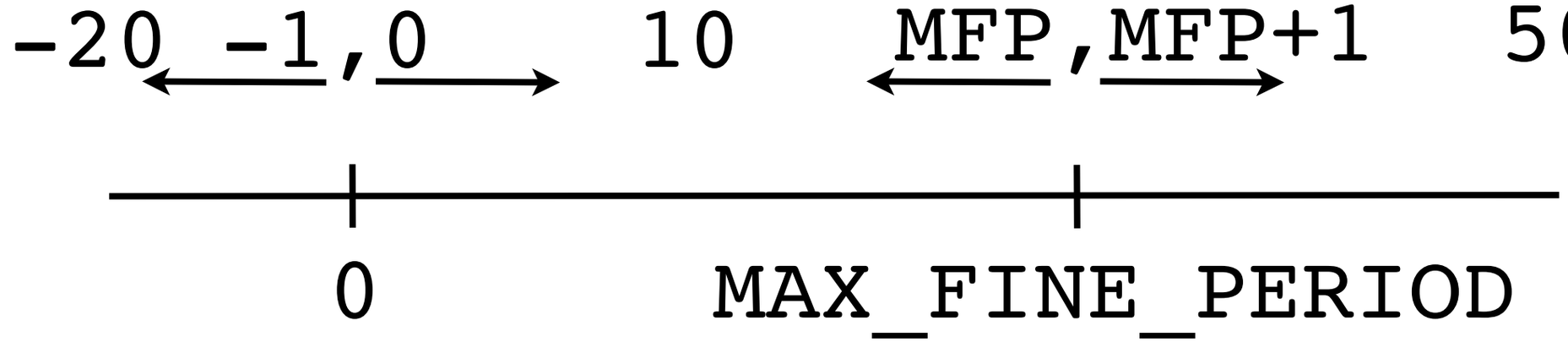
- values strictly within each region
- values at region borders
- include illegal regions

0 <= daysLate <= MAX_FINE_PERIOD

- within range
  - daysLate = (MAX_FINE_PERIOD - 0)÷2
- on boundary
  - daysLate = MAX_FINE_PERIOD
  - daysLate = MAX_FINE_PERIOD + 1
  - daysLate = 0
  - daysLate = -1
- out of range
  - daysLate = MAX_FINE_PERIOD + 50
  - daysLate = 0 - 20

note that some authors propose approaches which test less cases

test values for `daysLate`



equivalence partitioning and boundary analysis

Jacoco to check coverage

In the project

more test cases for implementation is dependent on which are not covered