



# Testing Oracles

# Testing oracle- definition

*“... the oracle can be a program specification, a table of examples, or simply the programmer’s knowledge of how the program should operate...”*

Howden

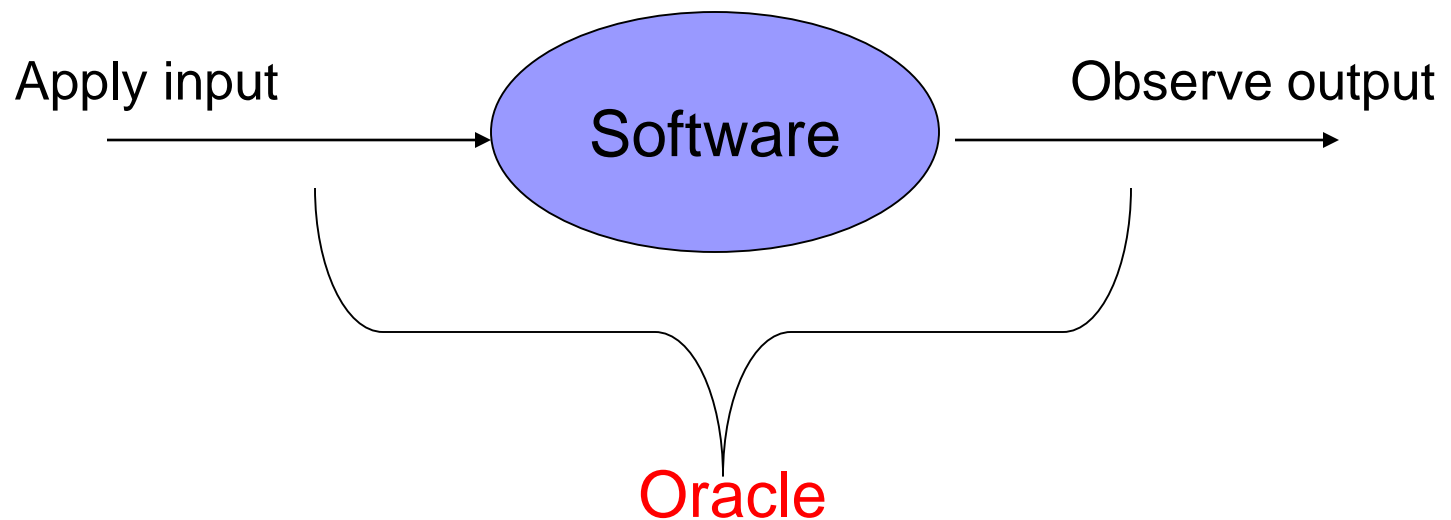
-a mechanism to evaluate the actual results of a test case as pass or no pass.

Two essential components:

- result generator (to produce the expected result for an input)
- comparator to check the actual results

Oracle = generation and comparison mechanism

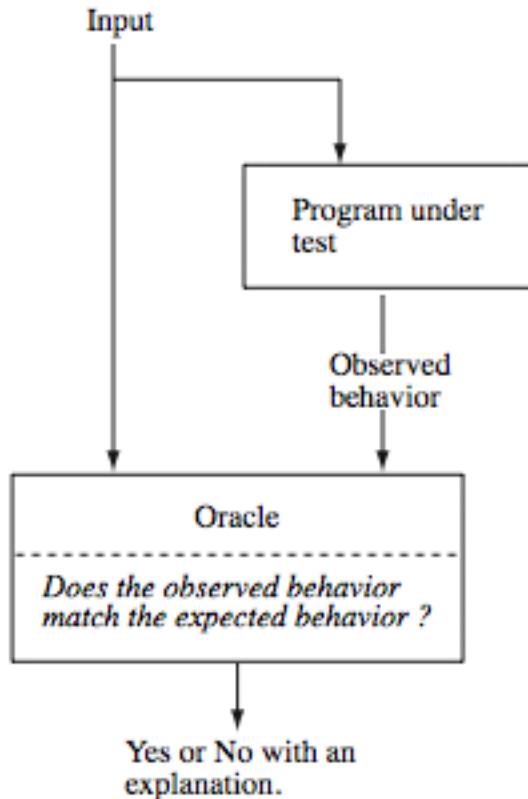
# Testing



Validate the observed output against the expected output

Is the observed output the same as the expected output?

# Oracle: Example (1)



How to verify the output of a matrix multiplication?

How to verify the output of a matrix inversion program?

How to verify the output of a sorting procedure?

## Oracle: Example (2)

A tester assuming the role of an oracle and thus serving as human oracle.

How to verify the output of a matrix multiplication?

Hand calculation: the tester might input two matrices and check if the output of the program matches the results of hand calculation.

Oracles can also be programs. For example, one might use a matrix multiplication to check if a matrix inversion program has produced the correct result:  $A \times A^{-1} = I$

How to verify the output of a sorting algorithm?

## Oracle: Example (3)

Calculations of variance

$$s^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}$$

Calculator formula (single pass through data)

$$s^2 = \frac{\sum_{i=1}^n x_i^2 - \frac{1}{n} (\sum_{i=1}^n x_i)^2}{n - 1}$$

Three data: 90,000,001    90,000,002    90,000,0003; correct answer =1

Sum of squares 24,300,001,080,000,014

Second formula: the result is 4/(3-1)=2, incorrect

*Catastrophic cancellation*



# Testing oracle- further considerations

**Oracles could be manual, automated, or partially automated**

**Perfect oracle** – behaviorally equivalent to the implementation  
(defect –free version of software under testing?)

Development of perfect oracle at least as difficult as developing original software

**Main features (development requirements)**

- Fidelity
- Generality (e.g., solved examples oracles)
- Cost



# Testing oracle: Main categories

## **Judging oracles**

### **Prespectification**

- Solved example oracles
- Simulation oracles
- Approximation oracles

### **Gold standard oracles**

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### **Organic**

- Smoke test
- Reversing oracles



# Judging oracles

## Judging oracles

### Prespectification

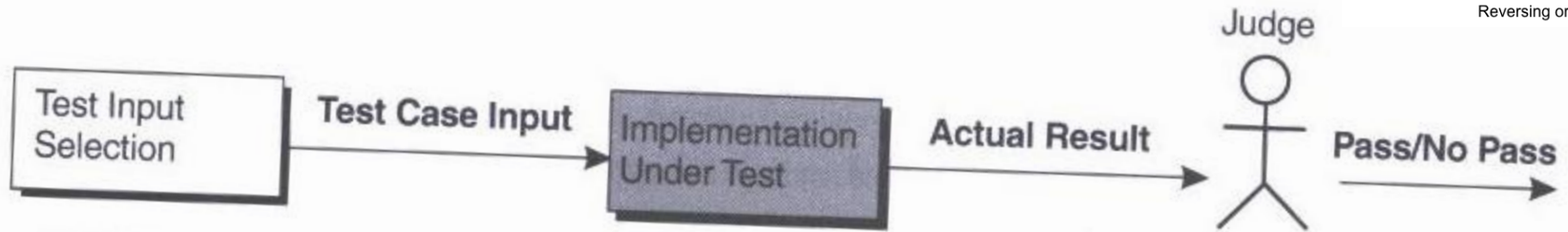
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



## Making pass/no pass decision

Based on analysis, subjective evaluation, or both

**Subjective evaluation** – test cases improvised and results assessed on the fly (ad hoc beta testing). Judgment limited by human abilities; slow, error prone

**Post-test analysis** - test cases do not specify or provide only a general indication of expected results. The results are recorded for each test case and checked

**Expert user** - in case of qualitative evaluation (say, high resolution graphics, audio)

# Solved example oracles

## Judging oracles

### Prespectification

- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles

Expected results are developed by hand

Each test case designed and output is prepared *manually*

The oracle useful at any scope but in general limited to evaluation of simple processing rules with several output variables

Used since the dawn of software development

# Solved example oracles

## Judging oracles

### Prespectification

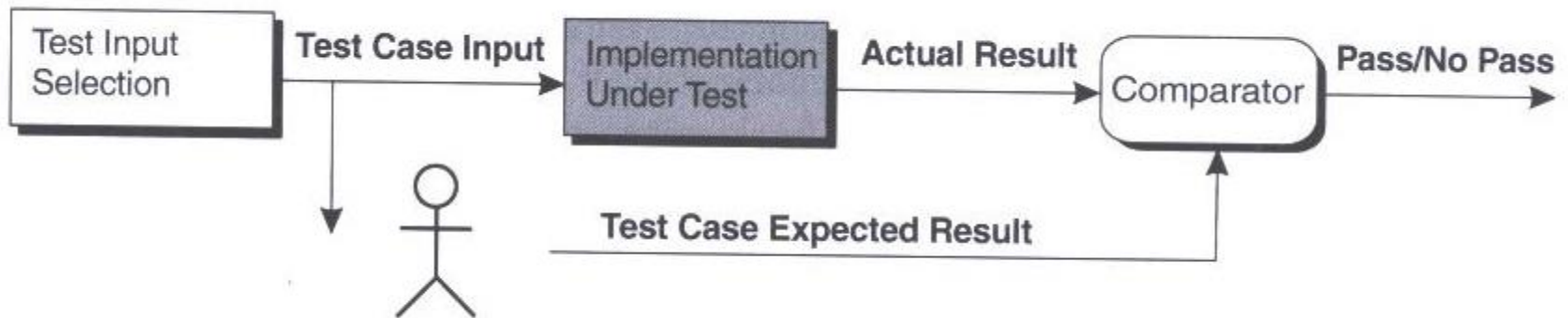
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



# Simulation oracles

## Judging oracles

### Prespectification

- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles

Expected results are produced using a reduced, simplified or prototype implementation of the system

A simulator should be designed so that its development and verification are easier than the original system (say, computing and search → use a spreadsheet)

# Simulation oracles

## Judging oracles

### Prespectification

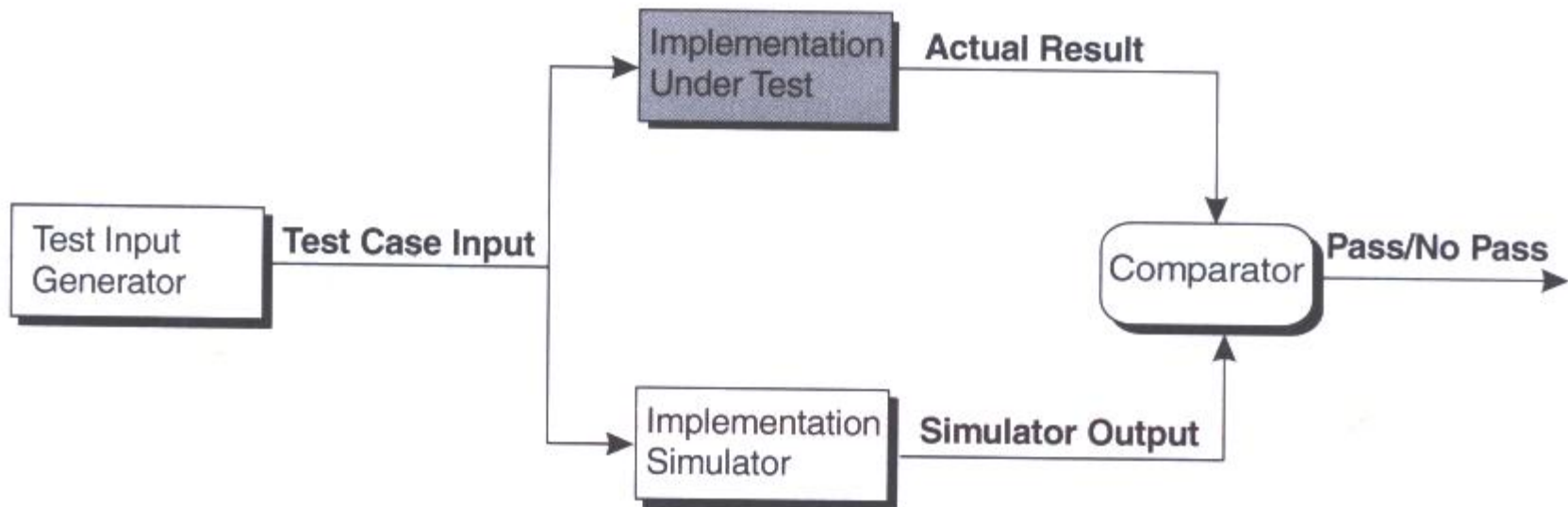
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



# Approximation oracle

Systems built to solve unsolved problems (with no analytical solutions),  
say weather/economic prediction, geological exploration, mechanical design

No simple techniques for checking the output produced by the system under testing

Some **strategies**

- Reduce the input/output space.** Tests that are simple for which the answer can be checked by hand or simulation

In some systems, a **subject expert** may be able to recognize an output as wrong or Inconsistent

**Constraints on some outputs** (“x should never be greater than y”, etc....)

Determine the **acceptable accuracy** of the results; interpolate

Judging oracles

Prespectification

Solved example oracles  
Simulation oracles  
Approximation oracles

Gold standard oracles

Trusted system oracle  
Parallel test oracle  
Voting oracle

Organic

Smoke test  
Reversing oracles

# Approximation oracle

## Judging oracles

### Prespectification

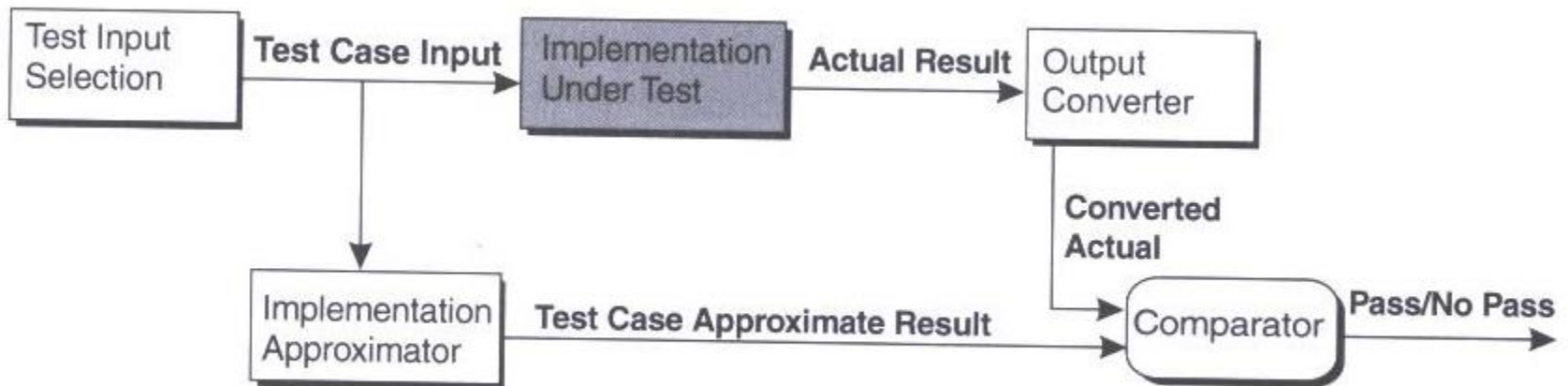
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles

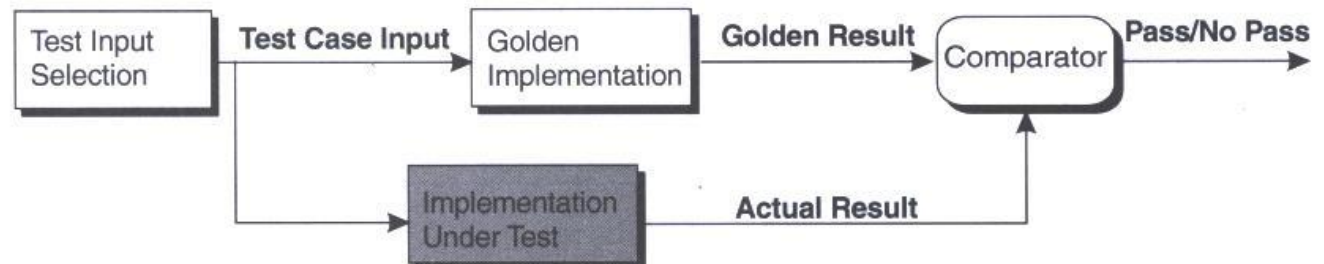


# Gold standard oracles

Gold standard oracles use *one or more versions* of an existing application system to generate expected results

Several alternatives:

- **Trusted system oracle**
- **Parallel test oracle**
- **Voting oracle**





# Gold standard oracles: trusted system oracle

## Judging oracles

### Perspectification

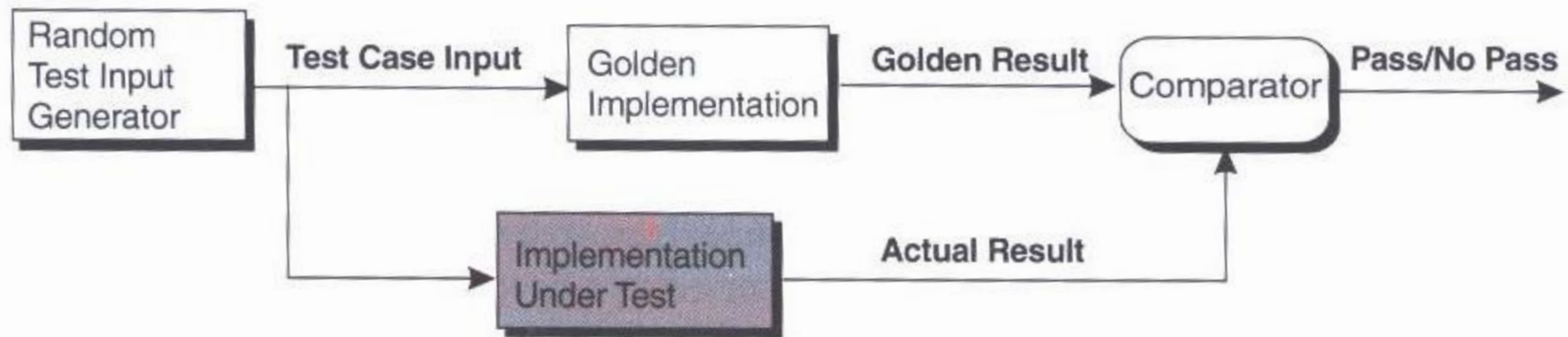
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



Existing system treated as trusted (routinely used with high confidence)  
has to compute many (if not all) of the same functions as the system under testing

Trusted system often present when dealing with legacy systems (to be replaced or ported); may need wrappers for them

Large quantity of test cases

# Gold standard oracles: parallel system oracle

## Judging oracles

### Prespectification

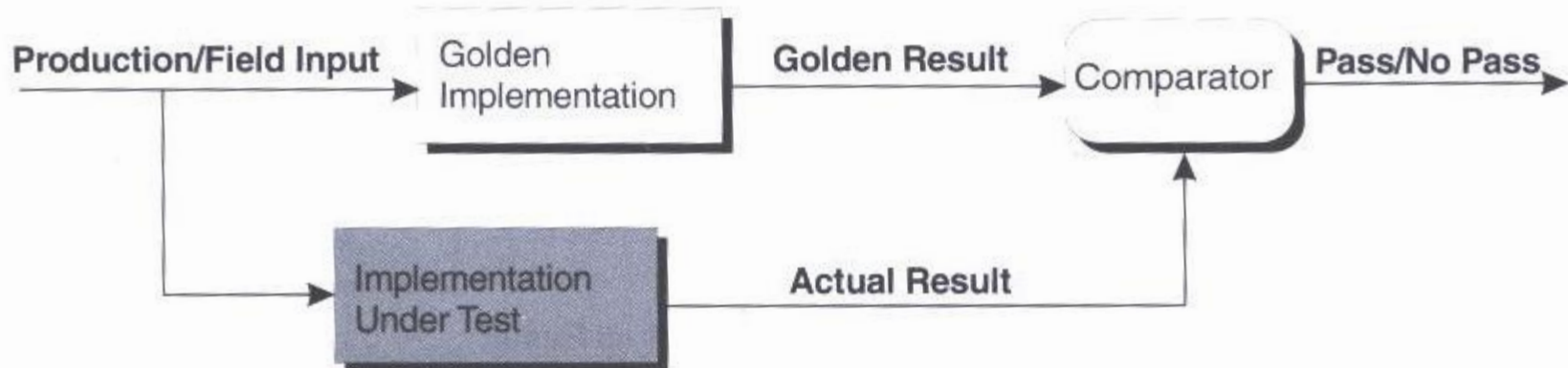
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



The use of the same input streams of data

Run on independent platforms; if not available, run each system in sequence so that there is no interaction

# Gold standard oracles: voting oracle

## Judging oracles

### Prespectification

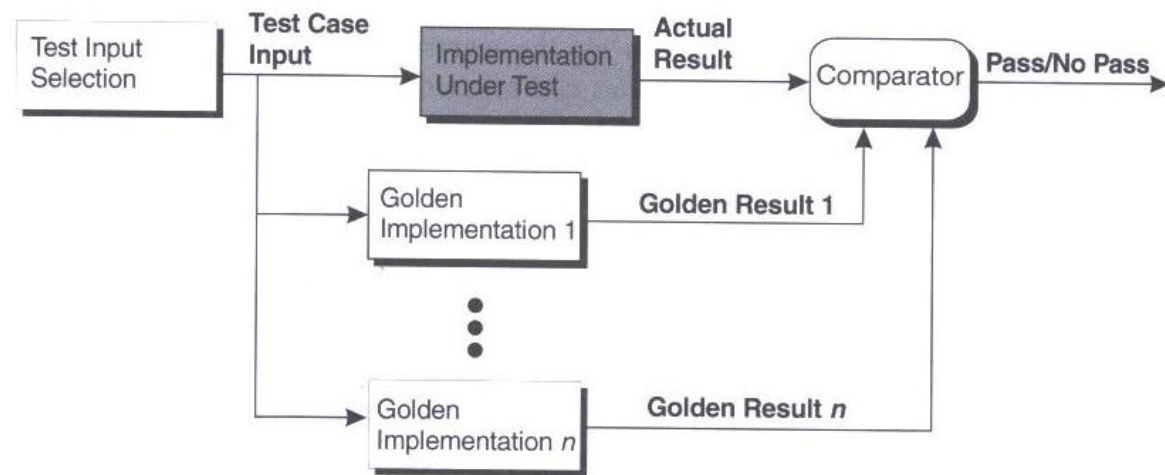
- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles



The use of two or more golden standards

Limited feasibility; need older versions of the trusted system as voters

May need data converters for input and output (the converters should be verified)

# Smoke tests

## Judging oracles

### Prespectification

- Solved example oracles
- Simulation oracles
- Approximation oracles

### Gold standard oracles

- Trusted system oracle
- Parallel test oracle
- Voting oracle

### Organic

- Smoke test
- Reversing oracles

Usage of the basic operability checks of the runtime environment

The test does not require any expected results. Any test case that runs without abnormal termination is considered to have passed

Useful at any scope to establish minimal operability

Limited value

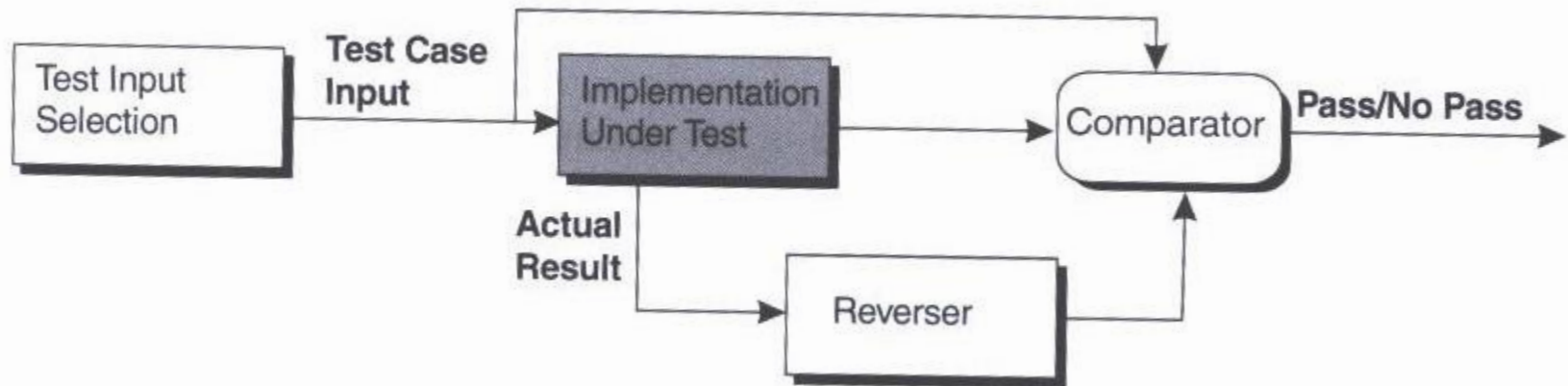
# Reversing oracles

*Some* computations could be reversible [quite unusual]

square root function vs. sorted list

If available could be exploited as oracles

No expected results need to be provided



# Reversing oracles: example

```
class Date {  
    // ...  
  
    Date tomorrow(Date aDay)  
        { /* Returns one day after aDay */}  
  
    Date yesterday(Date aDay)  
        { /* Returns one before aDay */}  
  
    Date today()  
        { /* Returns todays date */}  
  
    Date earlier(Date aDay, int diff)  
        { /* Returns the date diff days before aDay */}  
  
    Date later(Date aDay, int diff)  
        { /* Returns the date diff days after aDay */}  
  
    // ...  
}
```

Reversible relationship: the date one day earlier than one day after today is today

**assert (d.earlier(later(d.today(), 1),1)== d.today ());**

# Reversing oracles: reversible relationships

```
earlier( later( today( ),1),1) == today()
```

```
tomorrow(earlier(today(),1)) == today()
```

```
yesterday(later(today(),1)) == today()
```

```
earlier(((today()+1),1) == today()
```

```
earlier((FIRSTDAY + DAYRANGE), DAYRANGE) == FIRSTDAY
```

```
earlier(LASTDAY, DAYRANGE) + DAYRANGE == LASTDAY
```

```
yesterday(FIRSTDAY) == DateException
```

```
tomorrow(LASTDAY) == DateException
```



# Comparators

Automated comparison – a necessity (evaluation by inspection is not practical)

Categories of comparators

- System utilities (comparison of expected and actual results written to files – file comparison utilities)
- Smart comparators – greater control over comparison actions. COTS tools (File-Aid, DB Tester, Xdiff...)
- Application-specific comparators



# Oracles - evaluation

## Judging oracles

### Prespectification

Solved example oracles  
Simulation oracles  
Approximation oracles

### Gold standard oracles

Trusted system oracle  
Parallel test oracle  
Voting oracle

### Organic

Smoke test  
Reversing oracles

Low cost

High cost

narrow range

broad range

narrow range

broad range

High fidelity

•smoke test  
•reversing

•trusted system  
•parallel testing

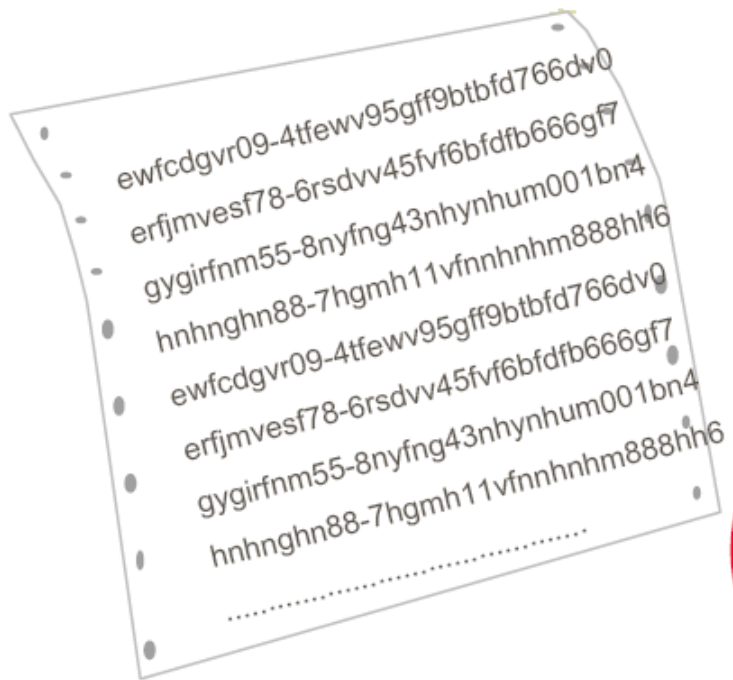
•solved examples  
•voting

Low fidelity

judging

approximation

# Concluding Comment



Test suites may contain hundreds or thousands of test cases



The amount of data to be checked cannot be handled by human beings

Automated oracles are essential!