What Is Testing?

- Process of determining whether a task has been correctly carried out [Schach '96]
- Goals of testing
  - Reveal faults
    - Correctness
    - Reliability
    - Usability
    - Robustness
    - Performance

  } Conflicting Goals?

Facts About Testing

- Question "does program P obey specification S" is undecidable!
- Every testing technique embodies some compromise between accuracy and computational cost
- Facts
  - Inaccuracy is not a limitation of the technique
  - It is theoretically impossible to devise a completely accurate technique
  - Every practical technique must sacrifice accuracy in some way

Cost/benefit

- Testing takes more than 50% of

Execution-based Testing

- Generating and *executing* test cases on the software
- Types of execution-based testing
  - Testing to specifications
    - Black-box testing
  - Testing to code
    - Glass-box (white-box) testing
  - Remember: difference is in generating test

White-box Testing

- **Example**

```
INPUT-FROM-USER(x);
If (x <= 300) {
        INPUT-FROM-FILE(BALANCE);
        If (x <= BALANCE)
                GiveMoney x;
        else Print “You don’t have $x in your account!!”}
else
        Print “You cannot withdraw more than $300”;
```

Testing Approaches

- Top-down
- Bottom-up
- Big bang

- Unit testing
- Integration testing
- Stubs
- System testing

Glossary

- **Exception (IEEE)**
  - An event that causes suspension of normal program operation. Types include addressing exception, data exception, operation exception, overflow exception, protection exception, underflow exception
- **Anomaly (IEEE)**
  - Anything observed in the documentation or operation of software that deviates from expectations based on previously verified software products or reference documents

Structural Testing

- **Coverage-based testing**
  - Test cases to satisfy statement coverage
  - Or branch coverage, etc
- **Complexity-based testing**
  - Cyclomatic complexity
    - Graph representation
    - Find the basis set
    - # Of braches + 1

Mutation Testing

-

Example

- VCR command-line software
- Commands
  - Rewind
    - If at the end of tape
  - Play
    - If fully rewound i 1901 74 5704 -3504 re681 Tm1 0190 rg0 Tc0 Tw(–)Tj/T

Preconditions & E

- Rewind
  - Precondition: end_of_tape
  - Effects: end_of_tape
- Play
  - Precondition: end_of_tape
  - Effects: end_of_tape
- Eject
  - Precondition: end_of_tape
  - Effects: has_taphas_06 630.

– Effects1q1 1 1 rg426.521 69576 3

Initial and Goal States

- Initial state
  - end_of_tape & has_tape
- Goal state
  - has_tape
- Plan?
  - Play
  - Eject

Iterative Relaxation

- Key idea
  - Path-oriented testing
  -


w=u

Test Coverage & Adequacy

- How much testing is enough?
- When to stop testing
- Test data selection criteria
-

Categories of Criteria

Classification according to underlying testing approach

- **Structural testing**
  - **Coverage of a particular set of elements in the structure of the program**
- **Fault-based testing**
  - **Some measurement of the fault detecting ability of test sets**
- **Error-based testing**
  - **Check on some error-prone points**

Structural Testing

- **Program-based structural testing**
  - **Control-flow based adequacy criteria**
    - Statement coverage
    - Branch coverage
    - Path coverage
      - Length-i path coverage
    - Multiple condition coverage
      - All possible combinations of truth values of predicates
  - **Data-flow based adequacy criteria**

Structural Testing

- **Data-flow based adequacy criteria**
  - All definitions criterion
    - Each definition to some *reachable*

Test Oracles

- Discussion
  - Aut49.5(6(maAut49.i)8)9.Orae D1-5.ay(n)]TJ/TT4 1 T-f0.80

Purpose of Test Oracle

- Sequential systems
  - Check functionality
-

Parts of an Oracle

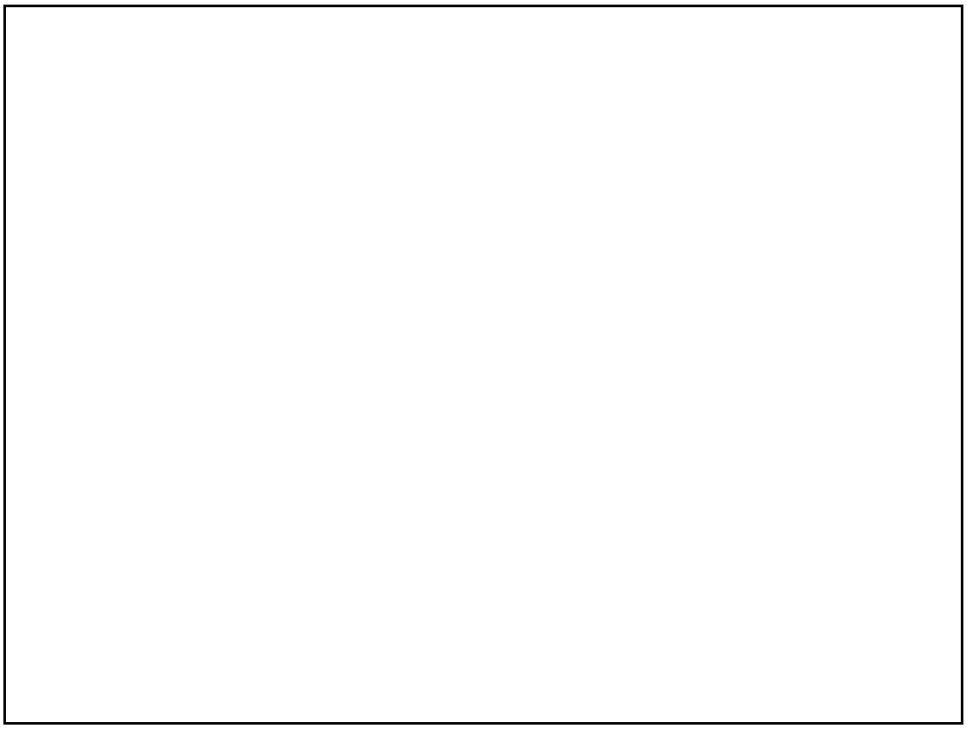
- **Oracle information**
  - **Specifies what constitutes correct behavior**
    - Examples: input/output pairs, embedded assertions
- **Oracle procedure**
  - **Verifies the test execution results with**

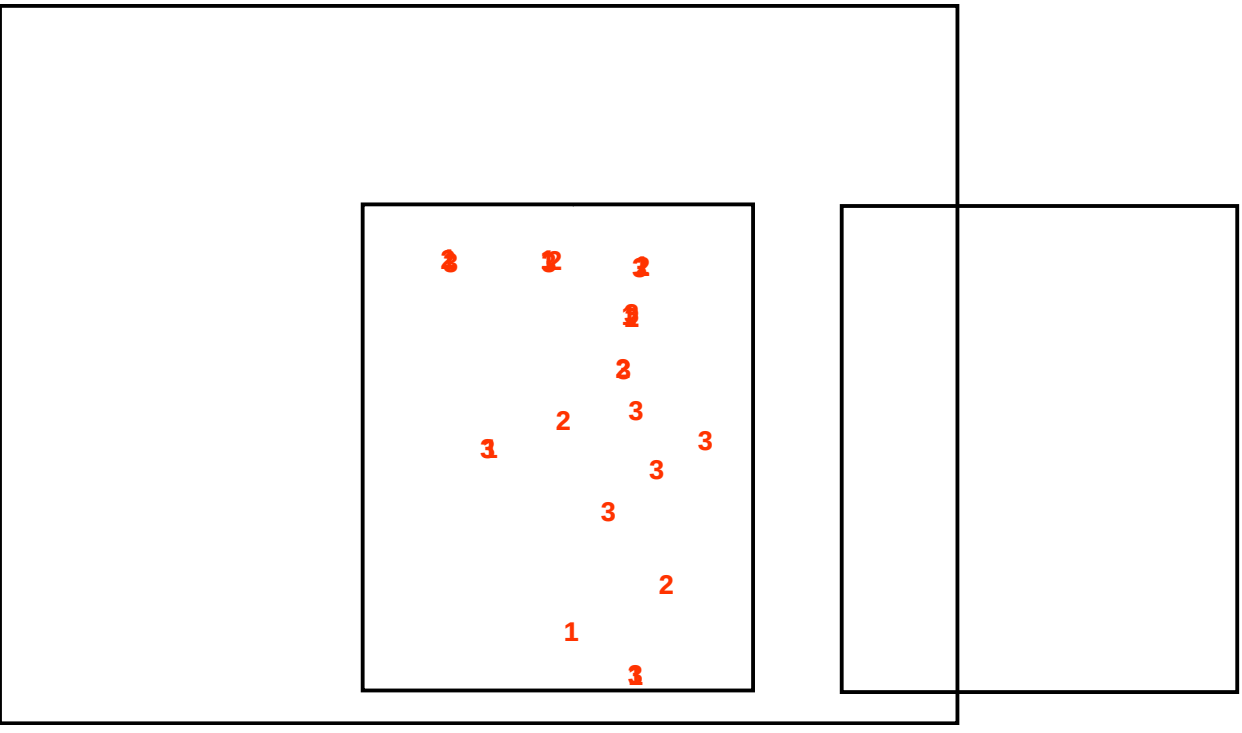
Formal Definition

- Given a program P,
- Its modified version P', and
- A test set T
  - Used previously to test P
- Find a way, making use of T to gain sufficient confidence in the correctness of P'

Selective Retesting

- Tests to rerun
  - Select those tests that will produce different output when run on P'




2
1
2
1
2
1
2
3
2
3
1
3
3
2
1
3

Data-flow Testing

read(x, y)

x := x + 2;

y := 2;

x := x + 2;