Test Coverage & Adequacy

- How much testing is enough?
- When to stop testing
- Test data selection criteria
- Test data adequacy criteria
  - Stopping rule
  - Degree of adequacy
- Test coverage criteria
- Objective measurement of test quality

Preliminaries

- Test data selection
  - What test cases
- Test data adequacy criteria
  - When to stop testing
- Examples
  - Statement Coverage
  - Branch coverage
  - Def-use coverage
  - Path coverage–

Goodenough & Gerhart [‘75]

- What is a software test adequacy criterion
  - Predicate that defines “what properties of a program must be exercised to constitute a thorough test”, i.e., one whose successful execution implies no errors in a tested program

Uses of test adequacy

- Objectives of testing
- In terms that can be measured
  - Fobxample branch coveragecy

Categories of Criteria

- **Specification based**
  - **All-combination criterion**
    - choices

Others

- Random testing
- Statistical testing
- Interface based

Classification according to

Structural Testing

- **Program-based structural testing**
  - **Control-flow based adequacy criteria**
    - Statement coverage
    - Branch coverage
    - Path coverage
      - Length-i path coverage
    - Cyclomatic number criterion
      - Set of v independent paths, where $v = e - n + 1$
    - Multiple condition coverage
      - All possible combinations of truth values of predicates
  - **Data-flow based adequacy criteria**

Structural Testing

Fault-based Adequacy

-

Properties of Criteria

- Program-based
- To recognize a good adequacy criteria
- And to discard poor choices
- Objective, well-defined properties

1. Applicability Property

- For every program, there exists an adequate test set
- Every program must be adequately testable

Criteria

Exhaustive test set

- If all representable points of the specification's domain have been tested
  - Set of all inputs for which the program *should* produce the desired output
- Exhaustive test set is surely adequate
  - No matter what criterion is used
- There *can*

3. Monotonicity

- Once a program has been adequately tested, running some additional test cases cannot cause the program to be deemed inadequately tested
- If T is adequate for P, and T $\subseteq$ T' then T' is adequate for P
- "Stop when we find less than 50 errors per 1000 hoursm0.1i
- Note
  - An exhaustive test set is

4. Inadequate empty set

- If no testing has been performed, then the program cannot be considered adequately tested
- The empty set is not an adequate test set for any program

Program Equivalence

- P ≡ Q
  - P is equivalent to Q
- For x (input vector) in the specification’s domain
- P(x) = Q(x)
  - Results of P and Q on every x are same

5. Antiextentionality

- There are programs P and Q, such that $P \equiv Q$, and a test set T is adequate for P but T is not adequate for Q
- Remember
  - Program-based
- Semantic equivalence of two programs does not necessarily imply that they be tested the same way
- Program-based testing should consider the implementation, not the functions computed

Syntactic Closeness

- **Two programs have the *<u>same shape</u>***
  - **If one can be transformed into another by applying the following transformations, any number of times**
    - Replace relational operator

6. General Multiple Change

- There are programs P and Q, which are the same shape, and a test set T is adequate for P but T is not adequate for Q
-

7. Antidecomposition

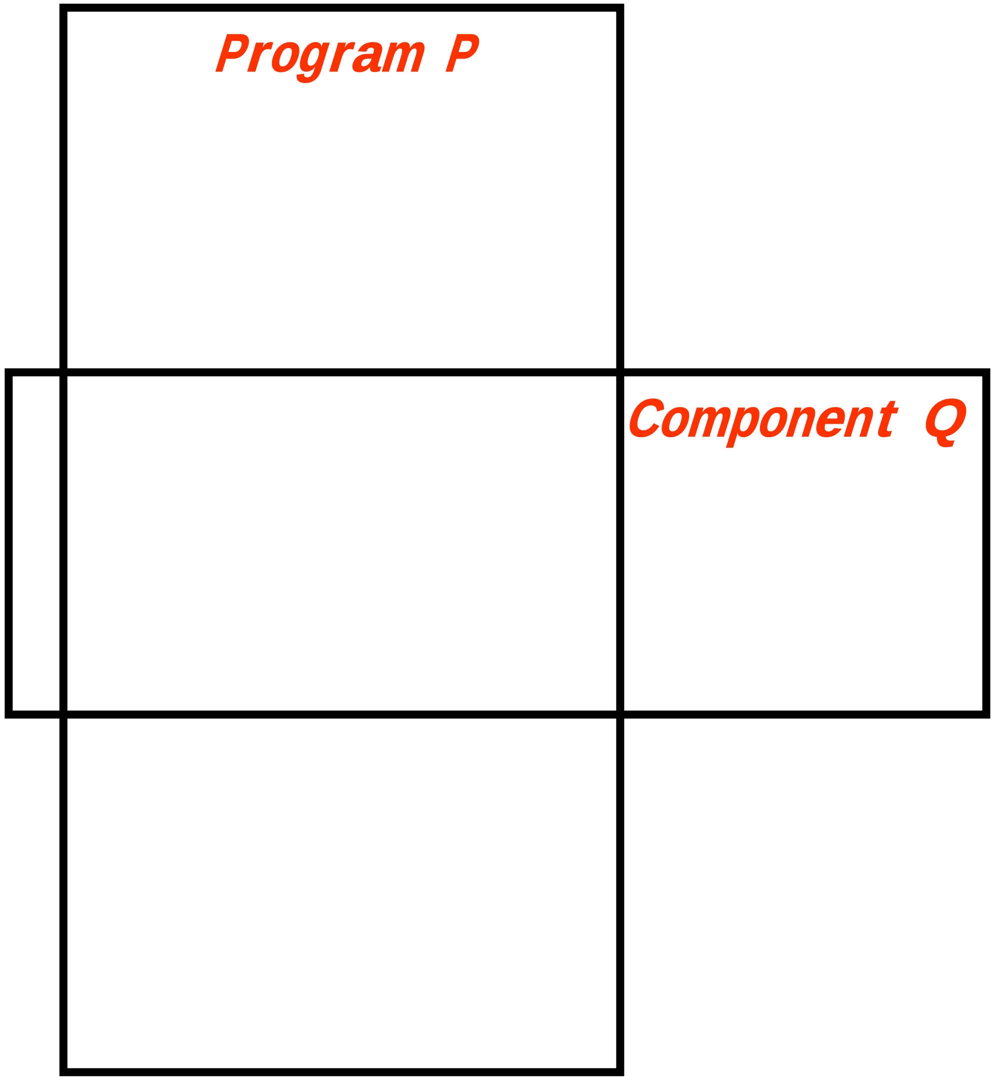
- There exists a program P, and
- component Q,
-

Explanation

T is adequate for P

$t \subseteq T$

T' is not adequate for Q



Explanation

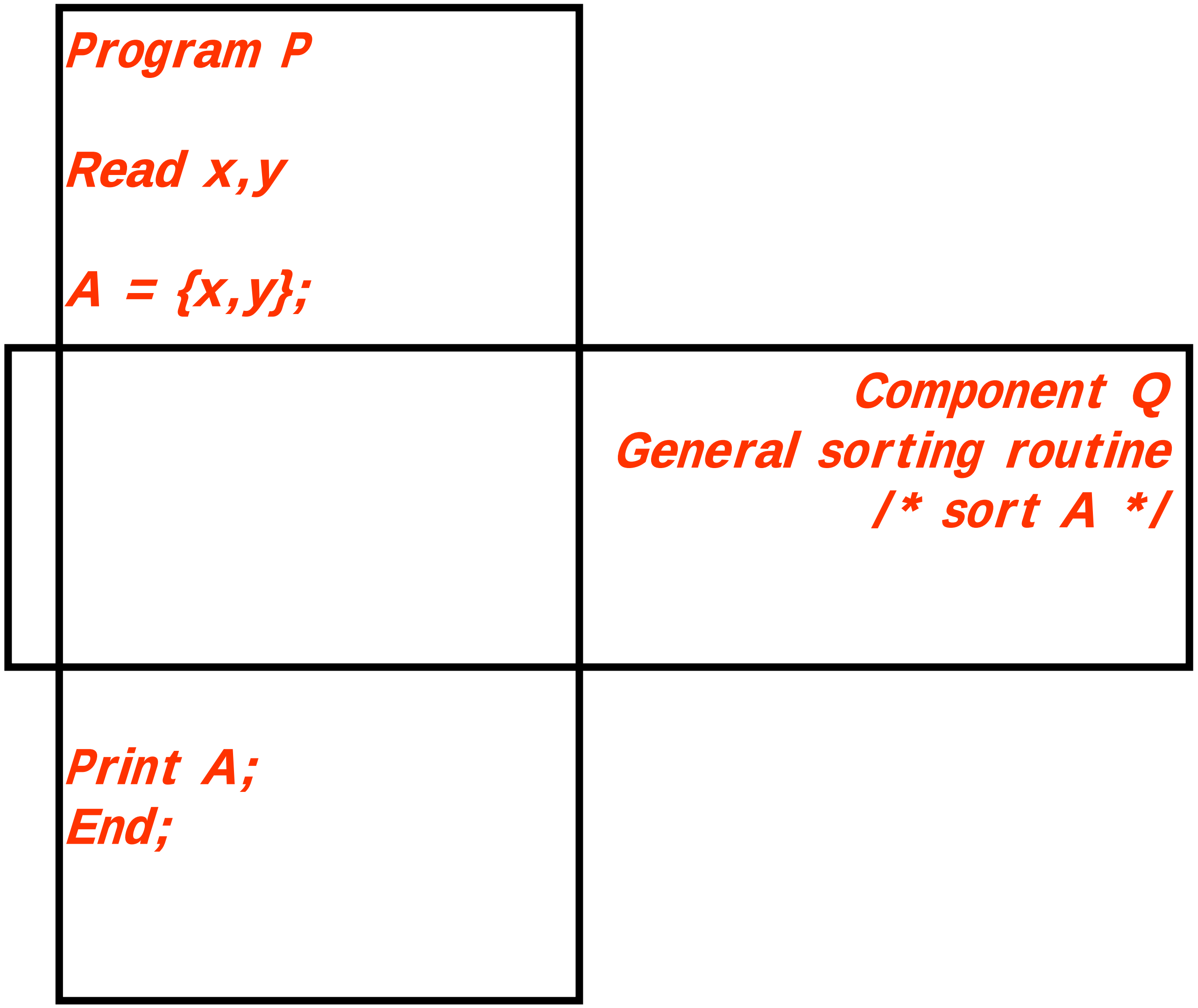
- Although a program has been adequately tested, it does not necessarily imply that each of its component pieces has been properly tested
- A routine that has been adequately

Explanation

T is adequate for P

$t \subseteq T$

T' is not adequate for Q



```
Program P

Read x,y

A = {x,y};
```

```
Component Q
General sorting routine
/* sort A */
```

```
Print A;
End;
```

Criteria

- Statement coverage
- Branch coverage

- Antidecomposition property rules out criteria that do not recognize that the context of a piece of code is importantis important

Criteria

- Statement coverage
- Branch coverage

- Anticomposition property eliminates criteria that do not have provision for testing the interaction of program pieces

Gödel Numbering

- Definition
  - A unique numerical value for each program, such that the program can be algorithmically retrieved from this value
- For a program P with Gödel number p
  - A test set T is Gödel adequate for P if $p \in T$
- Any test set T that contains a program P's Gödel number is adequate for P

Examining Gödel Adequacy

- Gödel adequacy has nothing to do with a program's semantics, syntax or specifications
- Every program will always have an adequate test set of size one
- Does this criterion satisfy all the properties that we have discussed?
- Do you think that this criterion is useful?

Program Renaming

9. Renaming Property

- Let P be a renaming of Q
- Test set T is adequate for P iff T is adequate for Q

- Intuitively, an “inessential” change in a program, such as changing variable names, should not change the test data required to adequately test the program
- Gödel adequacy does not satisfy this property!!

Canonical Representation

- Given a Program P with k variables
  - Obtain its canonical representation by
  -

Gödel-class Numbering

- Definition
  -

Subsumption

- Criteria C_1 subsumes criteria C_2, iff
  - For all programs p being tested with specifications s
  -