Testing the Implementation

- Verification & Validation phase
  - Verify - to meet specified requirements
  - Valid - true; test for correctness of software
- Verification techniques
  - acceptance tests
- Validation techniques
  - code walk-throughs
  - proofs of correctness

1

Types of Testing

- Unit Testing
  - Done by the developer
  - Tests routines (functions or modules) in isolation (other dependencies may be "stubbed" or "dummied up"
  - Unit Test Plan indicates what is to be tested and records the results for each unit test

2

Types of Testing (cont.)

- Integration Testing
  - Done by another group, outside of the control of the development staff
  - Brings subsystems together to be tested as a whole. Subsystems can be testing in isolation and then the entire system is built and tested as it would be used in the field.
  - Tests designed to catch things that might "fall between the cracks" on a large project

3

Types of Testing (cont.)

- Regression Testing
  - Used on "upwardly compatible systems"
  - Test cases which ran on the previous releases of the system are re-run, with their results being compared to the results of the previously released system
  - Designed to catch errors when features are modified when going to a new version of the system

4

Types of Testing (cont.)

- Acceptance Testing
  - The system is put to its first real-world test
  - The customer (if the work was done for hire) is involved in running the system in a trial field test to determine the fitness of the system
  - The customer's old system should be run in parallel with the new system (since this is a test, and things could fail!) to compare results

5

Types of Testing (cont.)

- Alpha (α) Testing
  - Term used to describe the in-house testing that goes on in product development
  - includes:
    - Unit Test
    - Integration Test
    - Regression Test
    - Any other in-house tests required by the development organization

6

Copyright © 1998, Jeffrey Blessing

Types of Testing (cont.)

- Beta (β) Testing
  - Term used to identify the first tests done outside the company that developed the system
  - Obviously, follows the alpha testing phase.
  - Many organizations publish beta software on the web, with the hopes of getting useful feedback from an informed user community

7

Types of Testing (cont.)

- Coverage Testing
  - Of all the possible paths of flow-of-control through a program, how many are exercised by a specific test?
  - For every pass through a code block, how many data dependencies are there which affect the correctness of the result?
  - Exhaustive testing is not possible! Therefore, what %'age of these conditions are covered?

8

Types of Testing (cont.)

- Boundary Testing
  - Usually used to test data structures (files, arrays, lists, etc.)
  - Often, the first time an operation is performed, or the last time an operation is performed, or when an operation works on the first or last item in the structure, special behavior is required.
  - Boundary testing addresses these cases

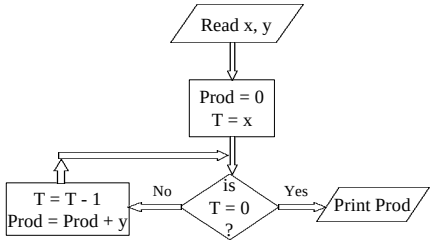
9

Proofs of Correctness

- We can mathematically prove some programs to be correct
- Using a method of "inductive assertions" we can demonstrate this on small, simple programs
- Applying this methodology to larger programs is quite labor intensive
- Do this only for critically important code

10

Multiply 2 ints by Repeated +



11

Proof Method

- Assertions are needed to prove paths correct
- Minimally we need:
  - Input assertion (I)
  - Output assertion (O)
  - Each loop must be *cut* by at least one assertion (the "loop invariant" assertion) (L)
- By induction, if I → L, and L → O, then its true that I → O

12

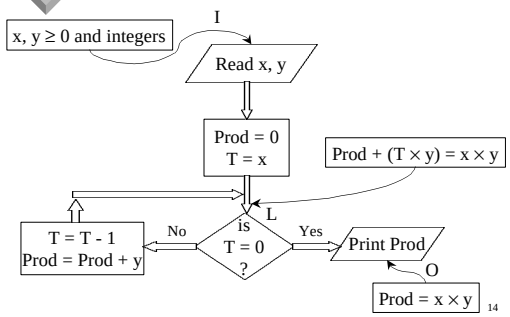
Copyright © 1998, Jeffrey Blessing

Assertions

- Input Assertion
  - → x, y ≥ 0 and integers
- Output Assertion
  - → Prod = x × y
- Loop Invariant Assertion
  - → Prod + (T × y) = x × y
- Add assertions to the flowchart

13

Multiply 2 ints by Repeated +



14

Three Paths of Execution Possible

I → L; L → O; L → L´

- I → L
  - → What happens going from I to L?
  - → Prod = 0 and T = x
  - → Substitute this into L: (Prod + (T × y) = x × y)
  - → 0 + (x × y) = x × y (True!)
  - → This will be true for all non-negative integers
  - → So the path I → L is correct!

15

Three Paths of Execution Possible

I → L; L → O; L → L´

- L → O
  - → T = 0
  - → Substitute this into L:
  - → Prod + (T × y) = x × y
  - → Prod = x × y
  - → This is the output assertion!
  - → Thus, L → O

16

Three Paths of Execution Possible

I → L; L → O; L → L´

- L → L´
  - → Prod´ = Prod + y; T´ = T – 1
  - → Prod = Prod´ – y; T = T´ + 1
  - → Substitute this into L:
  - → Prod + (T × y) = x × y
  - → (Prod´ – y) + ((T´ + 1) × y) = x × y
  - → Prod´ – y + T´y + y = x × y
  - → Prod´ + T´y = x × y (which is L´)!

17

Proofs of Correctness

- As an exercise, try to perform a proof of correctness on a similar program
  - – exponentiation by repeated multiplication
  - – division by repeated subtraction
- The tough part is coming up with the correct Loop Invariant assertion!

18

Copyright © 1998, Jeffrey Blessing