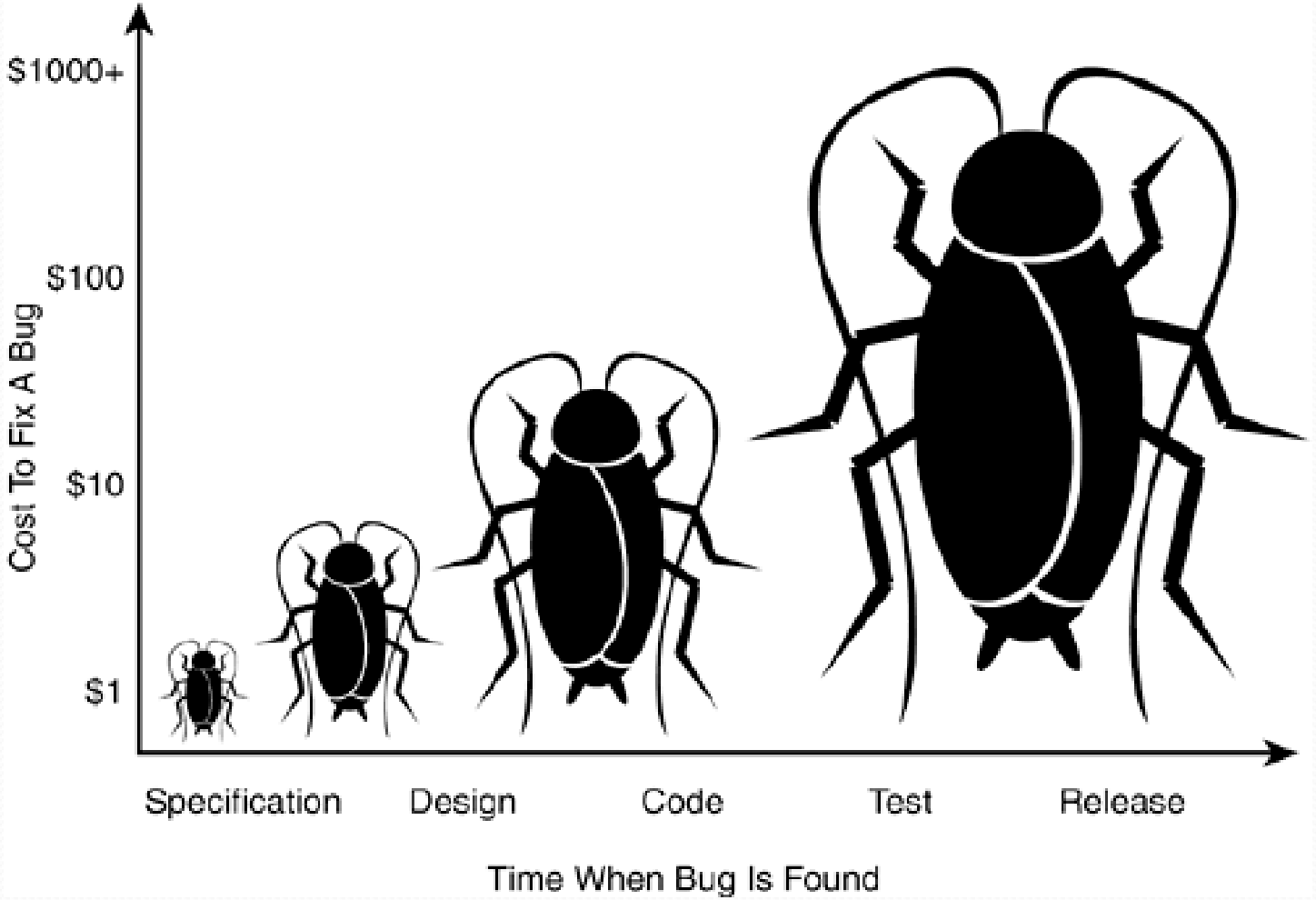
COMP 3711

(OOA and OOD)

Software Testing 5
Finding Bugs and Debugging

Cost of fixing bugs



Finding Bugs

- Requires the right attitude
- Some have more talent in this area than others
- Testers short term goals are often in conflict with other team members short term goals
- Cheapest to find bugs as early as possible
- In some environments, the more bugs the testers find, the lower their popularity

Finding bugs: Attitude

- Testers tend to find what they expect to find
- Testers will find fewer bugs if they expect the program to work correctly
- Testers should attack the program with the attitude that there are bugs to be found and that they are going to find them
  - A search problem or treasure hunt
- Testers should not attack the programmers in any way

Clustering of Bugs

- The more bugs found in a particular piece of code, the higher the probability that more bugs will be found in that piece of code
  - Mistakes tend to be made in bursts
  - Programmers often make the same mistake more than once
  - The first bug may be the tip of the iceberg. Lurking below may be a huge mass of bugs all related to the same fundamental problem

Real Life Examples

- Light does not turn on?
- Laptop cannot perform wireless-connect to internet?

Debugging

- When a test case uncovers an error debugging begins
- Debugging is a consequence of successful testing
- Debugging is more of an art than science as it deals with symptomatic indication of software problem and tries to match to the hidden cause

Debugging

- Goal:
  - To find and correct the cause of a software error
- Outcome of debugging:
  - Cause is found and corrected
  - Cause is not found and corrected
- Approaches:
  - Brute force (when all else failed)
  - Backtracking
  - Cause elimination

Debugging

- Once a bug is found, it must be corrected
- Three simple questions to ask before making correction:
  - Is the cause for the bug reproduced in another part of the program?
  - What “next bug” might be introduced as a result of this fix?
  - What could have been done to prevent this bug in the first place?

Reproducing Bugs

- If a bug cannot be reproduced, it cannot be fixed
- Ideally:
  - Put computer in known state (may require reboot)
  - Execute a series of steps
  - Observe the bug
- Bug found using automated testing tools will normally be easier to reproduce

Problems Reproducing Bugs

- Remember that there are no intermittent software errors
  1. Forgotten details
  2. Bug changes something - replication now impossible
  3. Memory Dependency
  4. Initial state bug
  5. Bug requires corrupted data

Continue…

6. Bug is side effect of another bug
7. Intermittent hardware failure
8. Clock time dependency
9. Resource dependency
10. Special cases in the code
11. Alien intervention

Reproducing Bugs and State

- All of these problems are related to putting the computer into the **state** that it was in before it failed
- Much easier to reproduce this state if:
  - A systematic approach to testing is being used
  - Automatic testing tools that remember or do things in a predefined way are used
  - Program is reasonably stable

Test Design Bugs

- Once each level of software specification has been written, the next step is to design the tests.
- The tests should be designed before or during the construction of the software.
  - If not then it would be too tempting to test the software against what it is observed to do (which is not really testing at all), rather than against what it is specified to do.

F.I.R.S.T.

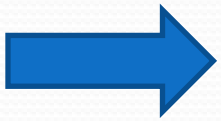
- Clean tests have the following charateristics:
  - Fast
    - Tests should run quickly
  - Independent
    - Tests should not depended on another test
  - Repeatable
    - Tests should be repeatable in any environment (e.g. QA, production, client)
  - Self-Validating
    - Tests should have boolean output (pass/fail)
  - Timely
    - Tests should be written before coding

When does testing end?

- Testing does not end following the conclusion of acceptance testing.
- Software has to be maintained to fix problems that show up during use and to accommodate new functionality.
- Software tests have to be repeated, modified and extended.
- The effort to revise and repeat tests consequently forms a major part of the overall cost of developing and maintaining software.

Two Big Questions

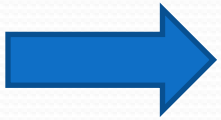
- Are we building the software product "right"?



Verification

Specifications conformance

- Are we building the "right" software product?



Validation

Requirements compliance