Software quality control techniques

- **Fault avoidance**: prevents errors before the system is released.
  - reviews, inspections, walkthroughs, development methodologies, testing, verification
- **Fault tolerance**: enables the system to recover from (some classes of) errors by itself.
  - rollbacks, redundancy, mirroring

Verification & Validation

- **Verification** is the process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.
  - Are we building the product right?
  - Testing, Reviews

- **Validation** is the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.
  - Are we building the right product?
  - Comparison against requirements

Terminology

- **Mistake** – a human action that produces an incorrect result.
- **Fault [or Defect]** – an incorrect step, process, or data definition in a program.
- **Failure** – the inability of a system or component to perform its required function within the specified performance requirement.
- **Error** – the difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

```
int factorial(int a){
    if(a==1)
          return 1;

    return a* factorial(a-1);
}
```

Mistake: Did not consider numbers less than 1

Exception or error output expected

Error

Overflow / crash

Failure

Lack of input validation

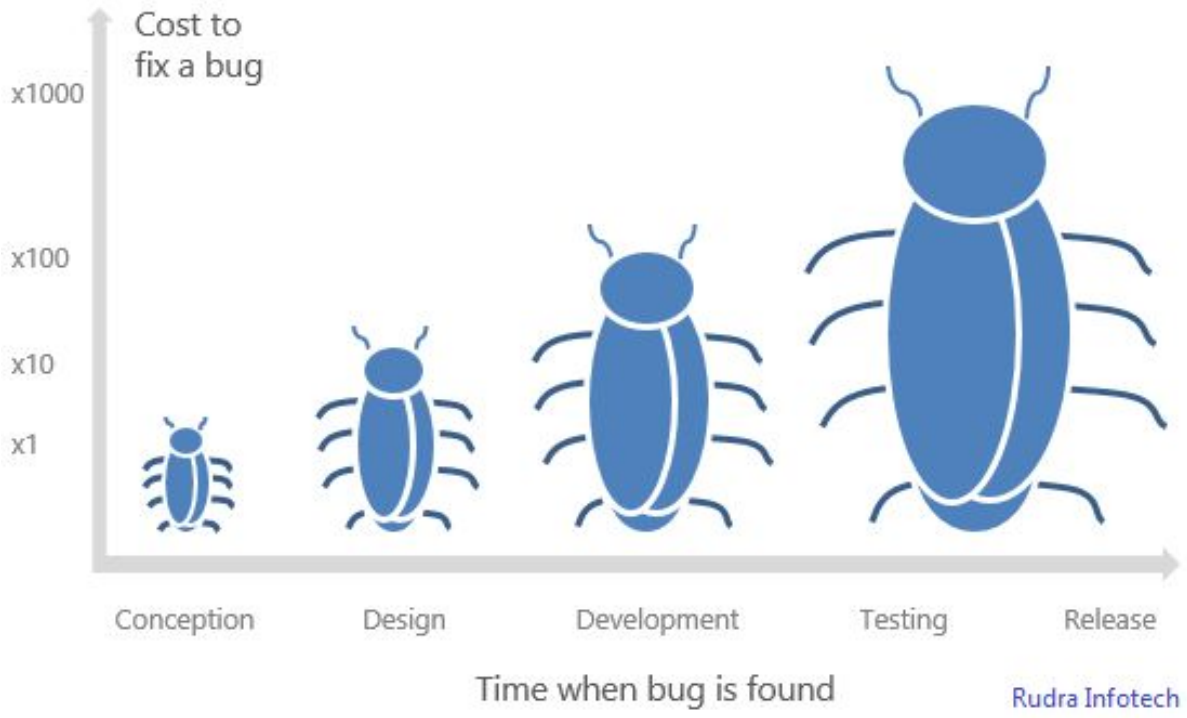
Defect: missing validation

Software Testing

Software testing is the process of analyzing a software item to detect the differences between existing and required conditions (i.e., bugs) and to evaluate the features of the software item.

When should we test software?

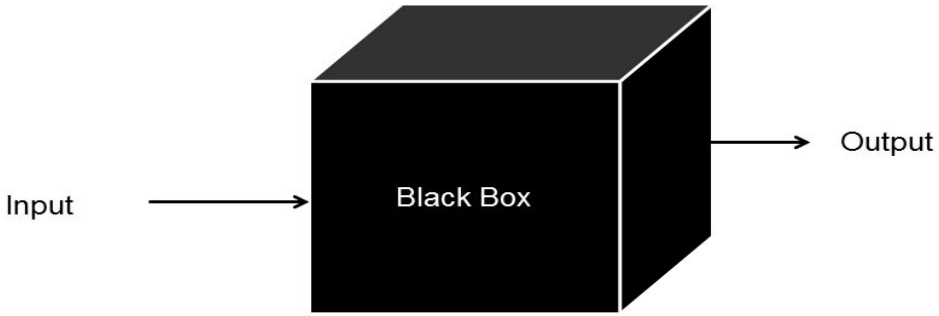
Answer: Throughout the whole development process



Testing goals

- Write test cases to cause failures.
- But, there is no way to guarantee that all faults have been detected.
- Work smart: write as few test cases as possible to cause failures; don't have more than one test cause the same failure

Testing model: Black box testing



- **Black box testing:** ignores the internal mechanism of a system or component and focuses solely on the outputs generated in response to selected inputs and execution conditions.
  - Interface visible, internals unknown
  - You know what it is supposed to do, you design tests that make it do what you think that it should do
  - From the outside, you are testing its functionality against the specs
  - For software this is testing the interface
    - What is input to the system?
    - What you can do from the outside to change the system?
    - What is output from the system?
  - Tests the functionality of the system by observing its external behavior
  - No knowledge of how it goes about meeting the goals

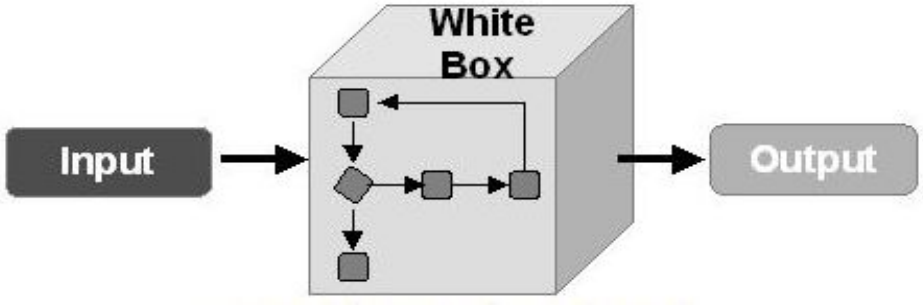
Advantages of Black box testing

- Process is not influenced by component being tested
  - Assumptions embodied in code not propagated to test data.
- Robust with respect to changes in implementation
  - Test data need not be changed when code is changed
- Allows for independent testers
  - Testers need not be familiar with code
- Useful for validation (are we building the right software)

Disadvantages of Black box testing

- It will miss bugs in the implementation that are not covered by the specification
  - Control-flow details
  - Performance optimizations
  - Alternate algorithms for different cases

Testing model: White box testing



- **White box testing** takes into account the internal mechanisms of a system or component.
  - Given knowledge of the internal workings, you thoroughly test what is happening on the inside
  - Close examination of procedural level of detail
  - Logical paths through code are tested
    - Conditionals
    - Loops
    - Branches
  - Status is examined in terms of expected values

Advantages of White box testing

- Finds an important class of boundaries.
- Yields useful test cases.
- Very useful to examine and test important data structures
- Useful for verification (are we building the software right)

Disadvantages of White box testing

- Tests may have the same bugs as implementation!
- Impossible to thoroughly exercise all paths
  - Exhaustive testing grows without bound

Types of testing

- Six types of testing
  - Unit testing
  - Integration testing
  - Functional / System testing
  - Acceptance testing
  - Regression testing
  - Beta testing

Unit testing

- Testing of individual hardware or software units or groups of related units
- Done by programmer(s)
- Generally all white box
- Verify that code does what it is intended to do at a very low structural level
- Automation desirable for repeatability

Integration testing

- Testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between them
- Done by programmer as they integrate their code into code base
- Verifies that units work together when they are integrated into a larger code base
  - Just because the components work individually, that does not meant that they all work together when integrated
- Generally white box, maybe some black box
- Automation desirable for repeatability

Non-Functional testing

- Testing conducted on a complete, integrated system to evaluate the system compliance with its specified requirements
- *Stress testing:* evaluating a system beyond the limits of its specification
- *Performance testing*: evaluating the compliance of a system with specified performance requirements.
- *Usability testing:* evaluating the extend to which a user can learn to operation, prepare inputs for, and interpret outputs of a system.
- It is recommended that this be done by external test group
- Mostly black box so that testing is not ‘corrupted’ by too much knowledge

Acceptance testing

- Formal testing conducted to determine whether or not a system satisfies its acceptance criteria (the criteria the system must satisfy to be accepted by a customer) and to enable the customer to determine whether or not to accept the system
- Generally done by customer/customer representative in their environment through the GUI
- Definitely black box

Regression testing

- Regression testing is selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements
- Subset of the original set of test cases.
- Core group of tests re-run often after any significant changes
  - Choose a representative sample of tests that exercise all the existing functionalities
  - Chose additional test cases that are most likely to be affected by the change
- *Smoke test: a* subset of the regression test cases that establish that the system is stable and all major functionality is present and works under “normal” conditions

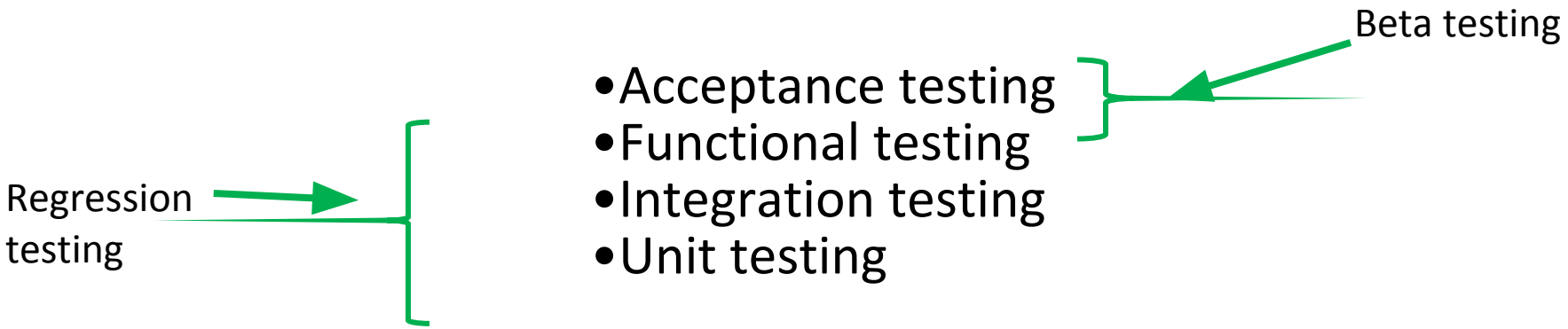
Beta testing

- Organization can offer an advance partial or full version of a software package free to one or more potential users.
- Users use the software as they wish, with the understanding that they will report any errors revealed during usage back to the organization.

- Advantages
  - Identification of unexpected errors
  - Low costs
  - Wider population / environment

- Disadvantages
  - Lack of systematic testing
  - Low quality error reports

Testing Hierarchy



Test plan

- Document describing the scope, approach, resources, and schedule of intended testing activities
- A test plan identifies
  - test items
  - features to be tested
  - Testing tasks
  - who will do the testing
  - any risks requiring contingency plans
- Test throughout the development cycle
- Write the test plan early in the development cycle
- If you wait until the end of the cycle, you might be in a very chaotic, hurried environment