

SEPM ASSIGNMENT 2

Done By

Harikrishnan V

CS6A

Roll No : 27

PRINCIPLES OF FUNDAMENTAL CODING

The principles that guide the coding task are closely aligned with

- Programming style
- Programming languages
- Programming methods

Fundamental coding principles are the following :-

- (i) Preparation principles
- (ii) Coding principles
- (iii) Validation principles

Preparation Principles

Before writing a line of code, we have to ensure the following :-

- Understand the problem.
- Understand basic design principles & concepts.
- Pick a programming language.
- Select a programming environment.

Coding Principles

As the coding proceeds following things are ensured:-

- Check your algorithms.
- Select data structures.
- Understand the software architecture & create interfaces that are consistent with it.
- Keep conditional logic as simple as possible.
- Create nested loops in a way that makes them easily testable.
- Select meaningful variable names & follow other local coding standards.
- Write code that is self-documenting.
- Create a visual layout (eg., indentation & blank lines) that aids understanding.

Validation Principles

After completing the coding, following things are conducted:-

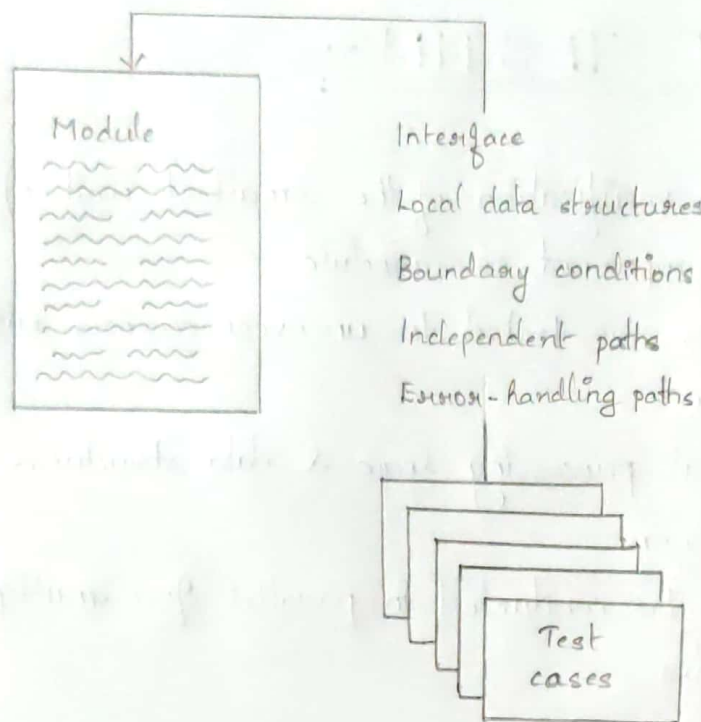
- Conduct a code walkthrough when appropriate.
- Perform unit tests & correct errors you've uncovered.
- Refactor the code.

UNIT TESTING

- Unit testing focuses on verification of the smallest unit of software design, the software component or module.
- Important control paths are tested to uncover errors within the boundary of the module.
- It focuses on the internal processing logic & data structures within the boundaries of a component.
- This type of testing can be conducted in parallel for multiple components or modules.

Tests Involved in Unit Testing

- The module interface is tested to ensure that information properly flows into & out of the program unit under test.
- Local data structures are examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution.
- All independent paths through the control structure are exercised to ensure that all statements in a module have been executed at least once.
- Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.
- And finally, all error-handling paths are tested.



Common Errors Found in Unit Testing

- Incorrect arithmetic precedence
- Mixed mode operations
- Incorrect initialization
- Inaccuracy
- Incorrect symbolic representation of an expression

Boundary Testing

- Boundary testing is one of the most important unit testing tasks.
- Software often fails at its boundaries.
- That is, errors often occur when the n th element of an n -dimensional array is processed.

Anti-Bugging

- A good design anticipates error conditions & establishes error-handling paths to reroute or cleanly terminate processing when an error does occur.

Unit Test Procedures

- The design of unit tests can occur after source code has been generated.
- A review of design information provides guidance for establishing test cases.
- Each test case is coupled with a set of expected results.
- Driver & stub modules have to be developed for each unit test.

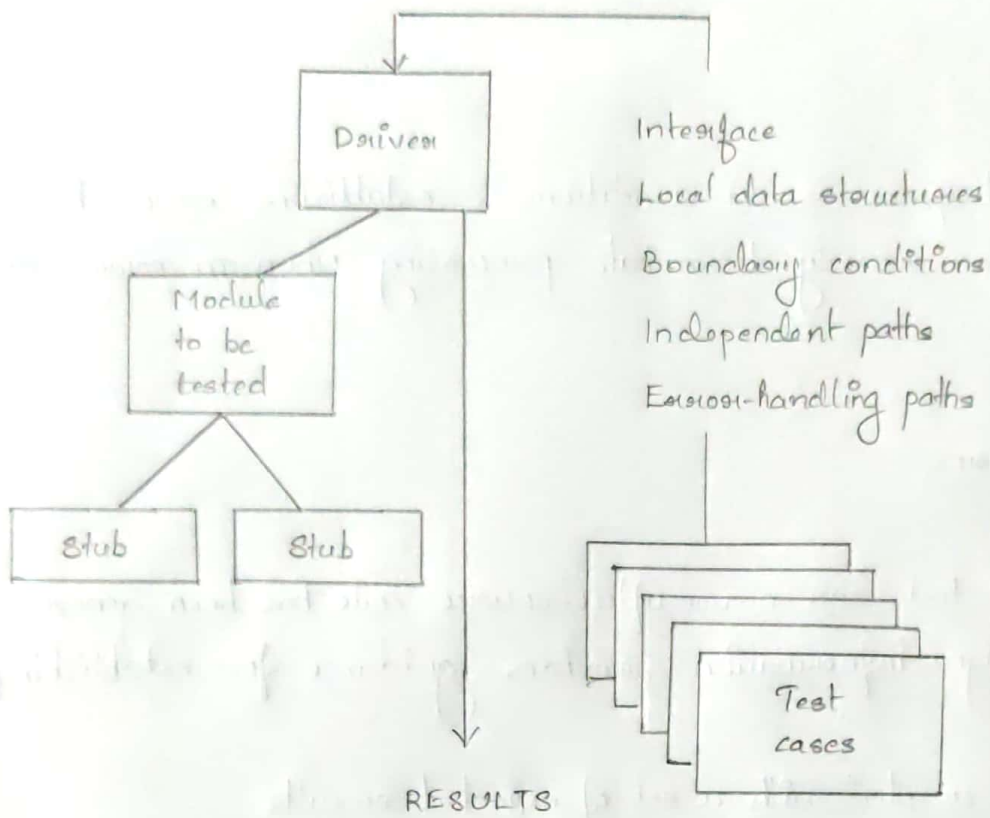
Driver & Stub Modules

• Driver

→ Driver is a dummy "main program" that accepts test-case data, & passes such data to the component to be tested, & prints relevant results.

• Stubs

→ A stub or "dummy subprogram" uses the subordinate module's interface, does minimal data manipulation, prints verification of entry, & returns control to the module undergoing testing.



BLACK BOX TESTING VS WHITE BOX TESTING

Black Box Testing

- It is a way of software testing in which the internal structure or the program or the code is hidden & nothing is known about it.
- It is mostly done by software testers.
- No knowledge of implementation is needed.
- It can be referred as outer or external software testing.
- It is functional test of the software.
- This testing can be initiated on the basis of requirement specifications document.
- No knowledge of programming is required.
- It is the behavior testing of the software.
- It is applicable to the highest levels of testing of software.
- It is also called closed testing.
- It is least time consuming.
- It is not suitable or preferred for algorithm testing.

White Box Testing

- It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
- It is mostly done by software developers.
- Knowledge of implementation is needed.
- It is the inner or the internal software testing.
- It is structural test of the software.
- This type of testing of software is started after detail design document.
- It is mandatory to have knowledge of programming.
- It is the logic testing of the software.
- It is generally applicable to the lower levels of software testing.
- It is also called as clear box testing.
- It is most time consuming.
- It is suitable for algorithm testing.
- Data domains along with inner or

- Can be done by trial & error ways & methods
- Eg: search something on google by using keywords

Integral boundaries can be better tested.

- Eg: search something on google by using keywords

- Eg: By input to check & verify loops