Introduction to Testing

→ Why focus on software testing?

- The only software defect detection technique that can check the whole system.
  eg - Compiler, processor, network etc.

- Currently, the best way to accurately assess some system behaviours (eg - performance)

- For contract software, necessary for customer to 'accept' the system.

- A reasonably good way of documenting expected system behaviour.

But testing is ALWAYS incomplete.

→ Problem with Software Testing

- It only samples a set of possible behaviour.
- There is no sound basis for extrapolating from tested to untested cases.

→ Overview idea on testing

Scale :

Unit tests : testing individual classes/ functions.

Integration tests : testing packages/subsystems.

System test : testing the entire system.

Process :

Test first : test-driven development (TDD)

- Write the tests before the code
- Write code to pass the test.

Test after :

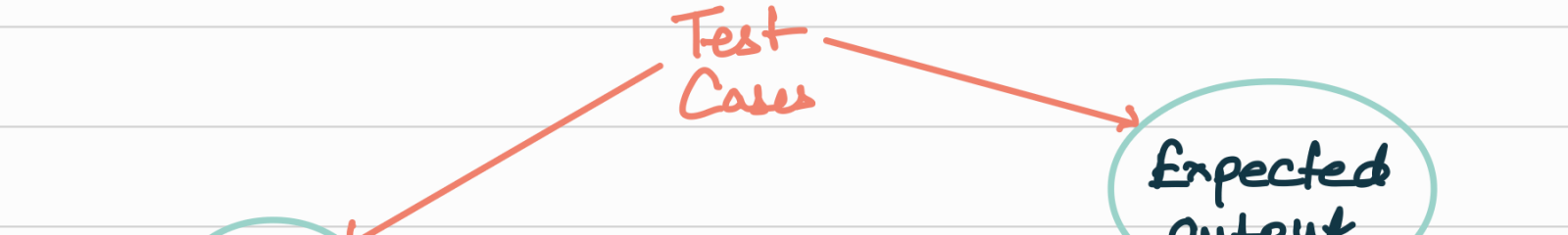
- Check whether the existing code passes test.

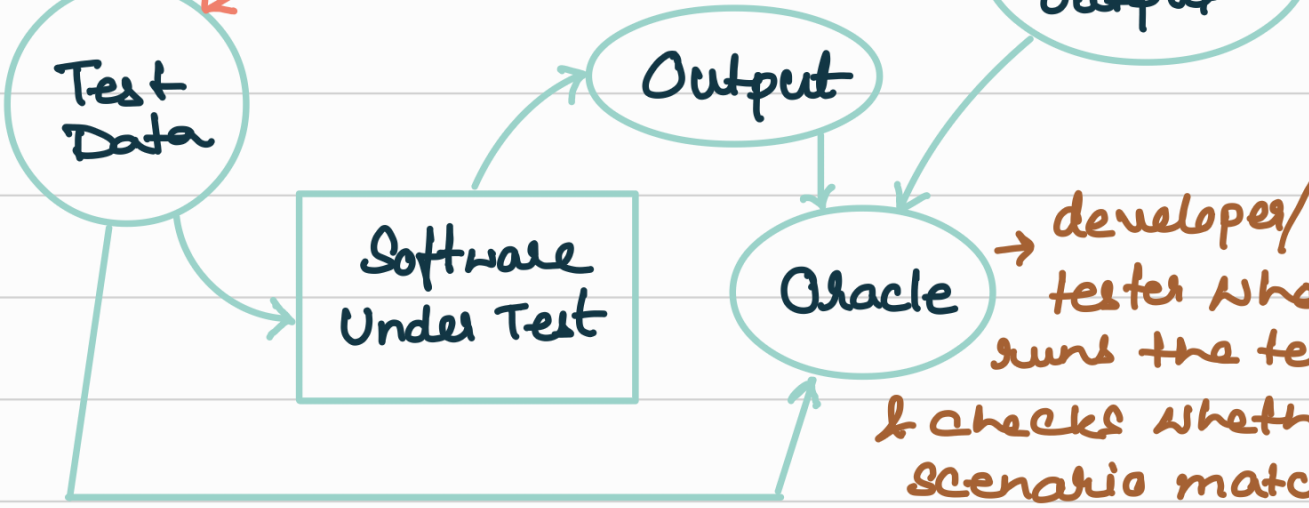
Iteration : You will anyway spend most of the time retesting.

Purpose :

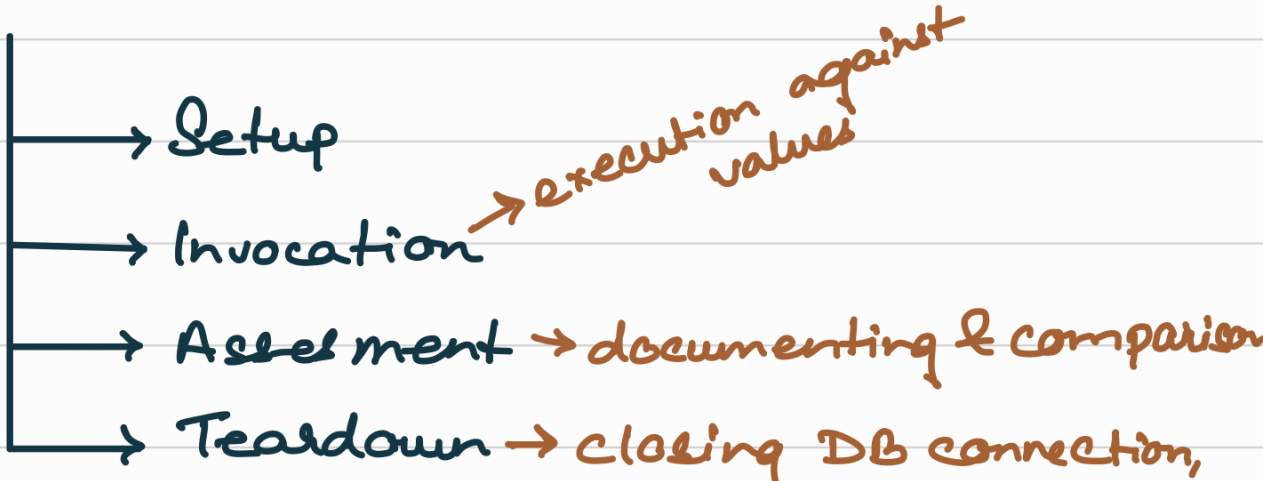
- Functional
- Performance
- Security
- Usability
- Availability testing

* What is a test?





Dissecting the Test anatomy :



- Testing a method is much more easier than testing a main() method.

- if you want to use your input in JUnit in main() for System.setIn(in);

```
String input = "1\n";
```

```
ByteArrayOutputStream in =
        new ByteArrayInputStream
                (input.getBytes());
System.setIn(in);
```

and —

```
ByteArrayOutputStream out =
        new ByteArrayOutputStream();
System.setOut(new PrintStream(out));
```

- Another reason why "main" methods are not used because for every env. type, input & outputs are different and it is very difficult to check every time, which is not the case for methods.