**Q1. Read and summarize J.A. Whittaker, “What is software testing And why is it so hard?”, IEEE software, January / February, 70-77, 2000” paper in your own words in not more than 300 words.**

The paper talks about why software testing remains one of the toughest parts of software development. And even after careful testing by testers, bugs often get missed and get released into production. It explains the reasons and challenges involved in this.

In the paper, the author highlights that software testing is not just about running programs but also about ensuring that software is working correctly in real environments when we put different inputs, user actions, and operating conditions. Testers need both coding skills and knowledge of languages, graph theory and algorithms in order to do the appropriate testing. They introduce a four-phase approach to testing:

1. Modeling the environment – simulating how software will interact with users, hardware, files, APIs and communication protocols.
2. Selecting test scenarios – choosing in optimal way which test cases to run, since testing every possible input and sequence is practically impossible. Testers use strategies like boundary value partitioning, coverage criteria, and input domain modelling.
3. Running and evaluating those tests – executing tests in either manual or automated ways, and then comparing actual results with the expected ones. This is hard because specifications may be incomplete, and outputs can be complex. Regression testing is also mentioned, where we rerun old tests on the new version to ensure fixes don’t break other features.
4. Measuring progress – interpreting this is quite difficult, deciding when to stop testing, and when it is completely done. Counting test cases or bugs is not enough, instead testers use adequacy criteria, reliability models and concepts like testability to judge completeness.

Conclusion – The paper emphasized that testing is not just about finding bugs, but about making smart choices under constraints. Since it is impossible to test everything, testers need to balance time, cost, and coverage. Testing requires creativity, technical depth, and proper strategies. Therefore, testing is as important and should be taken seriously as development itself, otherwise, software quality will suffer.

**Q2. Write a program to find out current date and generate test cases to verify the correctness of the program using the same model discussed in the class.**

import **datetime**

def **get\_current\_date**():

    today = **datetime**.**date**.**today**()

    return today.**strftime**("%Y-%m-%d")

if \_\_name\_\_ == "\_\_main\_\_":

**print**(**get\_current\_date**())

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Input Condition | Actual Output | Expected Output | Result |
| 01. | System date  e.g – 22-08-2025 | Current Date (2025-08-22) | Current Date (2025-08-22) | PASS |
| 02. | System date mismatch (past) | System Date | Current Date | FAIL |
| 03. | System date mismatch (future) | System Date | Current Date | FAIL |
| 04. | Corrupted datetime module | Error Message | Error Message | PASS |
| 05. | Invalid Date (29 Feb in non leap year) | Shows the date | Error Message (invalid date) | FAIL |

TESTING MATURITY MODEL(TMM)

* Level 1 (Initial): Program written, no structured testing
* Level 2 (Phase Definition): Basic testing done, date retrieval tested. (**Test Case 1**)
* Level 3 (Integration): Controlled and monitored the testing process, integrated into the software life cycle. (invalid dates, leap year check) (**Test Case 2, 3, 4, 5**)
* Level 4 (Management and Measurement): Software quality evaluation done (**Future Scope** – will automate all test cases with unit testing)
* Level 5 (Optimization/Defect Prevention and Quality Control): Test process optimization and quality control. (**Future Scope** - will be optimizing and preventing defects using CI/CD pipelines)