# SOFTWARE TESTING (IIITB)

**PROF. MEENAKSHI D'SOUZA**

Department of Computer Science and Engineering
IIT Kharagpur

**INTENDED AUDIENCE:** Any Interested Learners

**PRE-REQUISITES:** Programming, Algorithms, Discrete Mathematics (basics)

**INDUSTRY SUPPORT:** The material of this course has been used to offer training for Samsung, ABB and Mindtree. The course will be useful for any firm that does tests their software.

**COURSE OUTLINE:**

This course will cover various techniques for test case design, as used for testing of software artifacts including requirements, design and code. We will discuss algorithms and techniques for test case design based on graphs, logic, syntax of programming languages and on inputs. Special techniques for testing object-oriented features and web applications will also be discussed. The course will end with symbolic testing techniques. These broadly will cover test cases for both white-box and black-box.

**ABOUT INSTRCTOR:**

Prof. Meenakshi D'souza is currently an Associate Professor at IIIT-Bangalore.Meenakshi did her Master TMs in Mathematics from University of Madras, Chennai and her Ph. D. in Theoretical Computer Science from The Institute of Mathematical Sciences, Chennai. She joined the research department of Honeywell Technology Solutions, Bangalore soon after completing her Ph. D. and worked there in the areas of Formal Verification of Software Design, Model Based Development and Physical Access Control before joining IIIT-Bangalore. Her research interests are in Formal Methods, Model Based Development,Software Testing and Automata Theory.

**COURSE PLAN:**

**Week 1:** Techniques and algorithms for test case design: Graphs based testing- structural coverage criteria.

**Week 2:** Graphs based testing: Data flow coverage criteria

**Week 3:** Graphs based testing: Data flow coverage criteria

**Week 4:** Graphs coverage for source code, design elements and requirements

**Week 5:** Techniques and algorithms for test case design: Logic based testing- Predicates, logic based coverage criteria

**Week 6:** Specification based logic coverage, logic coverage on finite state machines

**Week 7:** Input space partitioning: Input domain modeling, combination strategies criteria

**Week 8:** Syntax based testing: Coverage criteria based on syntax, mutation testing

**Week 9:** Test case design (as learnt above) applied to object-oriented applications

**Week 10:** Test case design (as learnt above) applied to web applications

**Week 11:** Symbolic testing

**Week 12:** Concolic testing, Conclusion