Course Code	CSE 503
Course Name	Program Analysis
Credits	4
Course Offered to	UG/PG
Course Description	This course will focus on static and dynamic program analysis techniques that can be used to perform tasks such as program verification, profiling, optimization, repair, and comprehension. The students will learn the concepts behind the techniques, and will apply their learning to develop analyses using the state-of-art tools. Ensuring program correctness can be very challenging. Programmers depend on testing to build some confidence in the expected behavior of programs. Although testing is essential, the complexity as well as the criticality of modern software demands more rigorous techniques, in addition to testing, to ensure software correctness. The properties that programs need to satisfy vary from safety properties to security properties and in terms of expressiveness from simple predicates that check the program states at specific execution points to regular expressions and context-free grammars that check the legality of traversed program paths. In this course, we will focus on program verification and the students will learn about static analysis techniques, i.e., the techniques that can be applied to programs without running the programs as well as about dynamic analysis (or runtime monitoring) techniques that analyze programs during runtime. The students will learn about the strengths and weaknesses of both techniques. In the static analysis, we will primarily focus on dataflow analysis, and learn about some standard analyses that are used in program verification and optimization. However, the course will also introduce students to some advanced static analysis topics such as symbolic execution, and will cover some topics related to security analysis. We will primarily use Java as programming language and "Soot" as a static analysis tool. In the dynamic analysis part, the students will learn to specify properties of interest mainly using regular expressions and finite state automata, and use aspects to build runtime monitors to check those properties. In the process, they will learn to develop “aspects” to instrument and monitor programs.

Pre-requisites

Pre-requisite (Mandatory)	Pre-requisite (Desirable)	Pre-requisite(other)
CSE201 Advanced Programming	CSE322 Theory of Computation	
CSE102 Data Structures and Algorithms		

Post Conditions*(For suggestions on verbs please refer the second sheet)

CO1	CO2	CO3	CO4
Implement a dataflow analysis using Soot static analyzer.	Implement a simple flow-sensitive inter-procedural analysis.	Use a symbolic execution tool to generate test cases.	Develop aspects to analyze a program dynamically.

Weekly Lecture Plan

Week Number	Lecture Topic	COs Met	Assignment/Labs/Tutorial
1-6	Classic problems in static program analysis, monotone data flow framework including lattice concepts, control flow graph, pointer analysis, analysis with Soot.	CO1	A programming assignment to implement basic dataflow analyses using Soot.
7-10	Introduction to advanced topics including interprocedural analysis and symbolic execution.	CO2, CO3	A programming assignment involving an implementation of a simple interprocedural dataflow analysis using Soot. Another assignment will involve using a symbolic execution tool to generate test cases.
11-13	Runtime monitoring concepts, finite state properties, aspects, security automata.	CO4	A programming assignment to implement runtime monitors using aspects.

*Please insert more rows if required

Weekly Lab Plan			
Week Number	**Laboratory Exercise**	**COs Met**	**Platform (Hardware/Software)**

*Please insert more rows if required

Assessment Plan	
Type of Evaluation	**% Contribution in Grade**
End-sem	25
Mid-sem	25
Quiz	10
Programming assignments	40

Resource Material	
Type	**Title**
Reference	Principles of Program Analysis by Nielson, Nielson and Hankin.
Reference	Data Flow Analysis: Theory and Practice by Khedker, Sanyal and Karkare.
Reference	Compilers: Principles, Techniques, and Tools by Aho, Lam, Sethi and Ullman.
Other Materials	Research papers as assigned by the instructor