IT794 Compiler Construction [3 – 2 4]

Course Learning Outcome:

After successful completion of this course, student will be able to

- understand the functionalities of various compilation phases
- apply language theory concepts in various phases of compiler design
- design and develop a miniature compiler

Syllabus:

Overview of the Translation Process, Lexical Analysis, Hard Coding and Automatic Generation Lexical Analyzers

Parsing Theory: Top Down and Bottom Up Parsing Algorithms, Automatic Generation of Parsers

Error Recovery: Error Detection & Recovery, Ad Hoc and Systematic Methods

Intermediate Code Generation: Different intermediate Forms, Syntax Directed Translation Mechanisms and Attributed Mechanisms and Attributed Definition.

Run Time Memory Management: Static Memory Allocation and Stack Memory Allocation Schemes, Symbol Table Management.

Code Generation: Machine Model, Order of Evaluation, Register Allocation and Code Selection.

Code Optimization: Optimization of basic blocks, Flow Graphs and Data Flow Analysis, Code Improving Transformations.

Self-Study:

The self-study contents will be declared at the commencement of semester. Around 10% of the questions will be asked from self-study contents.

Laboratory Work:

Laboratory work will be based on above syllabus with minimum 10 experiments to be incorporated.

References:

1. Alfred V. Aho, Monica S. Lam, Ravi Shethi, Jeffrey D. Ullman, Compilers, Principles, Techniques and Tools, Pearson Education.
2. Keith D Cooper & Linda Torczon, Engineering a Compiler, Morgan Kaufmann Publisher Inc. San Francisco,USA.
3. Jean Paul Trembly & Paul G Sorenson, The Theory and Practice of Compiler Writing, McGraw-Hill computer science series