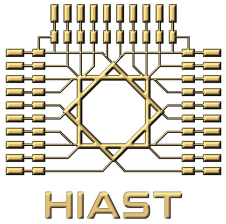
Syrian Arab Republic

Higher Institute for Applied Sciences and Technology



Course Specification Document

Title	Compilers

Credits	5 ECTS

Aims	This course aims to introduce the student to the main principles and theoretical foundations for building a simple compiler. It focuses on the analysis methods implemented by compilers to analyze the source code written in a programming language into its basic components (lexemes, sentences and semantics).

Intended learning outcomes

On successful completion of this course, the student will be able to:

- Understand the structure of a compiler.
- Identify the detailed components of the compiler (lexical analyzer, syntactic analyzer, semantic analyzer, code generator).
- Generate a lexical analyzer using the Flex tool and generate a syntactic analyzer using the Bison tool, achieving integration between the analyzers.
- Apply left and right recursive derivations.
- Establish the precedence of operations with defined rule priorities.

Syllabus

- **Introduction to compiler structure:** Basic components of the compiler, reviewing the concepts of formal languages, linking with lexical analyzer concepts.
- **Lexical analysis:** Function of the lexical analyzer, lexical analyzer tasks, building a lexical analyzer for language tokens, managing error messages resulting from lexical analysis of a program written in the input language.
- **Syntax analysis:** LL Parsing, Bottom-Up Parsing (LR(0), LR(1), SLR, LALR), constructing a syntax analyzer for language rules, resolving ambiguity issues, managing error messages resulting from syntactic analysis of a program written in the input language, designing and building the parsing table, abstract syntax tree for a program written in the input language.
- **Semantic analysis:** Semantic tree, symbol table, pattern verification, constructing a semantic analyzer for the input language capable of detecting semantic errors (patterns, etc.), managing error messages resulting from semantic analysis of a program written in the input language.
- **Equivalent code:** Methods for constructing a semantically equivalent program code for a source code but in a different language.