**BSc. (Hons) Discipline Specific Core Semester VII**

**(NEP-UGCF 2022)** (Guidelines)

**Compiler Design**

**(Effective for Academic Year 2025-26)**

Name of the Teacher: - Dr. Dharmendera Kumar Meena

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr.****No.** | **Topics** | **Chapter** | **No. of Hours** | **Reference** |
| **1** | **UNIT I:** Overview of compiler, Phases of compiler | 1.1, 1.2 | **2** | **[1]** |
| **2** | **UNIT 2: Lexical analysis:** role of Lexical analyser, specifications and recognition of tokens,symbol table, error reporting, regular expressions and definitions, lexical analyser - LEX | 3.1, 3.2,3,3, 2.7,3.4(3.4.1, 3.4.2,3.4.3), 3.5, | **9** | **[1]** |
| **3** | **UNIT 3:** CFGs, left recursion, left factoring, Top down parsing - LL parser, Bottom-up parsing - LR parser, Parser generator -YACC | 4.1.1, 4.1.2,4.1.3, 4.2(4.2.1 to4.2.5), 4.3, 4.4,4.5, 4.6, 4.9 | **15** | **[1]** |
| **4** | **UNIT 4:** Syntax directed definitions, evaluation orders for syntax directed definitions, Intermediate languages: syntax tree, three address code, Types and declarations, translation of expressions, loops and conditional statements, type checking | 5.1,5.26.1.1,6.2(6.2.1 to6.2.3), 6.3(6.3.1to 6.3.5), 6.4,6.5(6.5.1 & 6.5.2),6.6(6.6.1 to 6.6.4) | **10** | **[1]** |
| **5** | **UNIT 5:** Activation records, stack allocation, Issues in code generation- design of a code generator | 7.1, 7.28.1, 8.2, 8,3(onlyintroduction), 8.4(8.4.1 to 8.4.3) | **6** | **[1]** |
| **6** | **UNIT 6:** Principal sources of optimization, peephole optimization | 8.7 | **3** | **[1]** |

# References Text Books

1. Aho, A., Lam, M., Sethi, R., & Ullman, J. D. (2006). *Compilers: Principles,*

*Techniques, and Tools*. 2nd edition. Addison Wesley.

[https://theswissbay.ch/pdf/Gentoomen%20Library/Programming/Compiler/](https://theswissbay.ch/pdf/Gentoomen%20Library/Programming/Compiler/Aho%20-%20Compilers%20-%20Principles%2C%20Techniques%2C%20and%20Tools%202e.pdf) [Aho%20-%20Compilers%20-](https://theswissbay.ch/pdf/Gentoomen%20Library/Programming/Compiler/Aho%20-%20Compilers%20-%20Principles%2C%20Techniques%2C%20and%20Tools%202e.pdf)

[%20Principles%2C%20Techniques%2C%20and%20Tools%202e.pdf](https://theswissbay.ch/pdf/Gentoomen%20Library/Programming/Compiler/Aho%20-%20Compilers%20-%20Principles%2C%20Techniques%2C%20and%20Tools%202e.pdf)

**Reference Books**

 V Raghvan (2010), “ Principles of Compiler Design”, TMH.

Santanu Chattopadhayay (2005), “Compiler Design”, PHI.

**Suggested Practical List:**

1. Write a Lex program to count the number of lines and characters in the input file.
2. Lex program to count the number of vowels and consonants in a given string
3. Write a Lex program that implements the Caesar cipher: it replaces every letter with the one three letters after in in alphabetical order, wrapping around at Z. e.g. a is replaced by d, b by e, and so on z by c.
4. Write a Lex program that finds the longest word (defined as a contiguous string of upper and lower case letters) in the input.
5. Write a Lex program that distinguishes keywords, integers, floats, identifiers, operators, and comments in any simple programming language.
6. Write a Lex program to count the number of words, characters, blank spaces and lines in a C file.
7. Write a Lex specification program that generates a C program which takes a string “abcd” and prints the following output:

abcd

abc

ab

a

1. A program in Lex to recognize a valid arithmetic expression.
2. Write a YACC program to find the validity of a given expression (for operators + - \* and /) A program in YACC which recognizes a valid variable which starts with letter followed by a digit. The letter should be in lowercase only.
3. A Program in YACC to evaluate an expression (simple calculator program for addition and subtraction, multiplication, division).
4. Program in YACC to recognize the string „abbb‟, „ab‟ „a‟ of the langauge (an b n , n>=1).
5. Program in YACC to recognize the language (an b , n>=10). (output to say input is valid or not)