System Programming (BHCS15B) Discipline Specific Elective - (DSE)

Credit: 06

Course Objective

The course is focused on design of assembler and basic compiler. The course covers topics like absolute loader, relocating loader and dynamic linking.

Course Learning Outcomes

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

1. Describe the working of assemblers and compilers.
2. Use Lex/ Yacc for building basic compiler.
3. Develop a two pass Assemblers.
4. Describe the role of the loaders, linkers and relocatable programs.

Detailed Syllabus

Unit 1

Assemblers & Loaders, Linkers: One pass and two pass assembler, design of an assembler, Absolute loader, relocation and linking concepts, relocating loader and Dynamic Linking.

Unit 2

Introduction: Overview of compilation, Phases of a compiler.

Unit 3

Lexical Analysis: Role of a Lexical analyzer, Specification and recognition of tokens, Symbol table, lexical Analyzer Generator.

Unit 4

Parsing & Intermediate representations: Bottom up parsing- LR parser, yacc,three address code generation, syntax directed translation, translation of types, control statements

Unit 5

Storage organization & Code generation: Activation records, stack allocation, Object code generation

Practical

Projects to implement an assembler for a hypothetical language.

Programs to get familiar with Lex and Yacc

1. Write a Lex program to count the number of lines and characters in the input file.
2. 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.
3. Write a Lex program that finds the longest word (defined as a contiguous string of upper and lower case letters) in the input.
4. Write a Lex program that distinguishes keywords, integers, floats, identifiers, operators, and comments in any simple programming language.
5. Write a Lex program to count the number of identifiers in a C file.
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

a

8. A program in Lex to recognize a valid arithmetic expression.
9. 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.
10. A Program in YACC to evaluate an expression (simple calculator program for addition and subtraction, multiplication, division).
11. Program in YACC to recognize the string „abbb‟, „ab‟ „a‟ of the langauge (an b n , n>=1).
12. Program in YACC to recognize the language (an b , n>=10). (output to say input is valid or not)

References

1. Aho, A., Lam, M., Sethi, R., & Ullman, J. D. (2006). *Compilers: Principles, Techniques, and Tools*. 2nd edition. Addison Wesley.

2. Chattopadhyaya, S. (2011). *System Software*. P H I Learning.

Additional references:

1. Beck, L. & Manjula, D. (1996). *System Software: An Introduction to System Programming*. 3rd edition. Pearson Education.

2. Dhamdhere, D. M. (2015). *Systems Programming*. Tata McGrawHill.

Course Teaching Learning Process

- Use of ICT tools in conjunction with traditional class-room teaching methods
- Interactive sessions
- Class discussions

Tentative weekly teaching plan is as follows:

Week	Content
1-3	Assemblers & Loaders, Linkers: One pass and two pass assembler, design of an assembler, Absolute loader, relocation and linking concepts, relocating loader and Dynamic Linking.
4	Overview of compilation, Phases of a compiler.
5-6	Lexical Analysis: Role of a Lexical analyzer, Specification and recognition of tokens,Symbol table, lexical Analyzer Generator.
7-9	Parsing : Bottom up parsing- LR parser,yacc.
10-11	Intermediate representations: Three address code generation,syntax directed translation, translation of types, control statements
12-15	Storage organization & Code generation: Activation records, stack allocation, Object code generation

Assessment Methods

Written tests, assignments, quizzes, presentations as announced by the instructor in the class.

Keywords

Compilers, lexical analyzer, syntax directed translation, assembler, loader, linker.