III B.Tech II Semester

15ACS38-OBJECT ORIENTED ANALYSIS AND DESIGN & COMPILER DESIGN LAB

L	T	P	C
0	0	3	2

Course Objectives:

- *Practice the notation for representing various UML diagrams*
- *Analyze and design the problem by representing using UML diagrams*
- *Become familiar with all phases of OOAD.*
- *Design, develop and test software systems for engineering applications.*
- *Analyze technical solutions to computational problems and develop efficient algorithms.*

Problems that may be considered are

1. College information system
2. Hostel management
3. ATM system
4. Library management system
5. Passport Automation System
6. Political Administration System.

Compiler Design Experiments

1. Design a lexical analyzer for given language and the lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Simulate the same in C language.
2. Write a C program to identify whether a given line is a comment or not.
3. Write a C program to recognize strings under 'a', 'a*b+', 'abb'.
4. Write a C program to test whether a given identifier is valid or not.
5. Write a C program to simulate lexical analyzer for validating operators.
6. Implement the lexical analyzer using JLex, flex or other lexical analyzer generating tools.
7. Write a C program for implementing the functionalities of predictive parser for the mini Language specified
8. (a) Write a C program for constructing of LL (1) parsing.
   (b) Write a C program for constructing recursive descent parsing.
9. Write a C program to implement LALR parsing.
10. (a) Write a C program to implement operator precedence parsing.
    (b) Write a C program to implement Program semantic rules to calculate the expression that takes an expression with digits, + and * and computes the value.
11. Convert the BNF rules into Yacc form and write code to generate abstract syntax tree for the mini
12. Write a C program to generate machine code from abstract syntax tree generated by the parser.

[illegible]

Course Outcomes:

- *Find solutions to the problems using object oriented approach*
- *Represent using UML notation and interact with the customer to refine the UML diagrams*
  - *(i)* *Use Case Diagram.*
  - *(ii)* *Class Diagram.*
  - *(iii)* *Sequence Diagram.*
  - *(iv)* *Collaboration Diagram.*
  - *(v)* *State Diagram*
  - *(vi)* *Activity Diagram.*
  - *(vii)* *Component Diagram*
  - *(viii)* *Deployment Diagram.*
  - *(ix)* *Test Design.*

[illegible] [illegible]