SARVAJANIK UNIVERSITY
Sarvajanik College of Engineering and Technology
Bachelor of Technology

Year: B. Tech III (Semester VI)

Subject Name: System Programming **Subject Code:** BTAI14603
Type of course: Professional Elective Course
Prerequisite (if any): Data Structures

Rationale: System programming leads to the development of computer system software that manages and controls the computer operations. The low-level codes are very close to the hardware level and deal with things such as registers and memory allocations. The objective behind this course is to aware the students about various system softwares and their role in translation of high/mid level language program into low-level code. The students are also taught to develop their own system software programs.

Teaching and Examination Scheme:

Teaching Scheme				Theory Marks			Practical Marks		Total
L	T	P	C	TEE	CA1	CA2	TEP	CA3	
3	0	2	4	60	25	15	30	20	150

CA1: Continuous Assessment (assignments / projects / open book tests / closed book tests) CA2: Sincerity in attending classes / class tests / timely submissions of assignments / self-learning attitude / solving advanced problems TEE: Term End Examination TEP: Term End Practical Exam (Performance and viva on practical skills learned in course) CA3: Regular submission of Lab work / Quality of work submitted / Active participation in lab sessions / viva on practical skills learned in course.

Contents:

Sr. No.	Contents	Total Hrs
1.	**Overview of System Software:** Introduction, Software, Software Hierarchy, Systems Programming, embedded system programming, Machine Structure, Interface, Address Space, Computer Languages, Language Translator, Life Cycle of a Source Program, Levels of System Software.	03
2.	**System Files and Library Structures:** Introduction, Library and File Organization, Design of a record, Source Program/Data File structure, Object Code, Object Files, Executable File, Libraries	03
3.	**Assembly Language and Assembler:** Machine Language, Mnemonic Language, Machine vs. Mnemonic Language, Assembly Language and its role, Basis of Assembly language, Assembly language applications, Assembly Statements and basic Operations, Macros and Subroutines in assembly language, Introduction to Assembler, Classification , Working Principles, General procedure for Assembler design, Assembly Process, Assembler design criteria, Types of Assemblers, Two-Pass Assemblers, One-Pass Assembler, Variants of Assemblers	10

4.	**Macros and Macro Languages:** Definition of Macro, Advantages, Macro Languages, Macros with/without parameters, Macro Processors, Features, Two-Pass Macro Processor, One-Pass Macro Processor	**04**
5.	**Linkers and Loaders:** Principles of Linking, A simple Linking Process, Basic Linking Tasks and Procedure, Memory management during linking, Binding, Relocation, Linking Methods, Principles of Loading, Linkers vs. Loaders	**05**
6.	**High-Level Language Translation :** Need of High Level Language, Features of High Level Language, Regular Expression, Finite Automata, Grammars, Context-Free Grammars, Ambiguous and Unambiguous Grammars, Left-Recursive Grammar, Parser, Parsing table construction, Operator Precedence Parser	**10**
7.	**Design of Compilers:** Introduction, Compilation Process, Compiler Construction Tools, Lexical Analysis, Syntax Analysis, Intermediate Code Generation, Code Optimization, Scanner Generator- LEX, Parser Generator-YACC	**10**

Suggested Specification table with Marks (Theory): (For B. Tech only)

Distribution of Theory Marks					
R Level	**U Level**	**A Level**	**N Level**	**E Level**	**C Level**
20	20	15	5	-	-

Legends: R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate C: Create (Revised Bloom's Taxonomy)

Reference Books:

Sr. No	**Title of book /article**	**Author(s)**	**Publisher and details like ISBN**
1	Systems Programming	Pal Srimanta	OUP India
2	Systems Programming	Dhananjay Dhamdhere	McGraw Hill Education
3	System Software	Chattopadhyay Santanu	PHI Learning
4	Systems Programming	John Donovan	McGraw Hill Education

Note: Students should refer to the latest editions of books

Course Outcomes (CO):

Sr. No.	CO statements	Marks % weightage
CO-1	Explain the software hierarchy, importance of system software, different system files and system programming techniques.	15%
CO-2	Write assembly language programs and use various data structures to develop two-pass, one-pass assembler.	25%
CO-3	Discuss macro processor, its usage and compare various loading and linking schemes.	20%
CO-4	Demonstrate compilation process (scanning, parsing, intermediate code optimization) for a given programming construct.	40%

List of Open learning website:

- www.cs.jhu.edu/~scott/pl/lectures/parsing.html
- www.en.wikipedia.org/wiki/System_programming
- https://www.isi.edu/~pedro/Teaching/CSCI565-Fall15/Materials/LexAndYaccTutorial.pdf
- https://developer.ibm.com/technologies/systems/tutorials/au-lexyacc/

List of Experiments:

1. Develop a 'C' Program finds white spaces, number of newline characters from the given input.
2. Implement a lexical analyzer (Scanner program) to recognize identifiers, keywords and constants from the given input file and store them separately.
3. Write a C program to simulate lexical analyzer to validate arithmetic operators, relational operators and logical operators.
4. Convert the following regular expression (R.E.) into DFA and Write a‘C’ program to simulate the DFA for given input Strings.

   (i) a(a|b)* ab. (ii) digit(digit)*(.digit(digit)*|ε)

5. Implement Recursive Descent Parser program in ‘C’ for the following Grammar.

   P ---> E '#'
   E ---> T {('+'|'-') T}
   T ---> S {('*'|'/') S}
   S ---> F '^' S | F
   F ---> D | '(' E ')'

D ---> 0|1|……|9.

Write a program in a way that it will trace the processing of different non-terminals of above grammar for given input string.

6. Simulate a two pass assembler for given assembly program. Show the content of symbol table at the end of pass-one of an assembler and write intermediate code representation of the assembly program.

7. Write a program to remove left recursion from a given grammar.

8. Write a program to left factor the given grammar.

9. Write a program to illustrate steps of LL (1) parser for the given parsing table.

10. Study the following compiler construction tools:
a) LEX b) YACC