SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
FACULTY OF ENGINEERING AND TECHNOLOGY
COMPUTER SCIENCE AND ENGINEERING

COURSE PLAN

Course Code : CS2006 **Semester: III Sem M.Tech (CSE)**
Course Title : System Programming **Course Time: Jan. – May 2018**

Day Order	Timing*
Day 1	
Day 2	9.45AM – 11.30AM (Lab) , 2.20PM – 4.05PM (Theory)
Day 3	
Day 4	4.05PM – 4.55 PM (Theory)
Day 5	

*Any Three hours

Location : Theory : TP1302- Tech Park
Lab: TP710 – Tech Park -Compiler Lab

Faculty Details

S.No	Name	Office	Office hour	Mail id
1	Prof.S.S.Sridhar	Tech Park	8.30 – 4.30 P.M	sridhar.s@ktr.srmuniv.ac.in

References

1. Dhamdhere D.M., “Systems Programming”, Tata McGraw Hill Education Pvt. Ltd., 2011.
2. Alfred V Aho , Jeffery D Ullman, Ravi Sethi, "Compilers, Principles Techniques and tools ", Pearson Education ,2011.
3. Srimanta Pal, “Systems Programming”, Oxford University Press, 2011.

Objectives

1. To learn the various system software like assemblers, loaders, linkers and macro
2. To study the features of design phases and parsing techniques of a compiler
3. To learn the various techniques of syntax directed translation & code optimization

Assessment Details

Cycle Test – I	: 20 Marks
Cycle Test – II	: 20 Marks
Surprise Test	: 5 Marks
Assignment	: 5 Marks

Test Schedule

S.No.	DATE	TEST	TOPICS	DURATION
1		Cycle Test - I	Unit I & II	2 Hours
2		Cycle Test - II	Unit III,IV & V	3 Hours

Purpose

To learn the design principles of various system software and its techniques

Instructional Objectives	Course outcomes
1. To learn the various system software like assemblers, loaders, linkers and macro 2. To study the features of design phases and parsing techniques of a compiler 3. To learn the various techniques of syntax directed translation & code optimization	At the end of this course, the student will be able • To understand the basic functioning of various system softwares and its tools • To learn and implement various parsing techniques for the design of a compiler • To learn and implement translation, optimization and generation algorithms for the design of a compiler

Detailed Session Plan

INTRODUCTION AND ASSEMBLERS (13 hours)

Introduction: Language Processor Fundamentals, Data Structures Language Processing, Search data structures, Data Structures, Scanning, Parsing, Assemblers – Elements of assembly language programming, Simple assembly scheme, Pass structure of assemblers, Design of a two pass assembler, single pass assembler for IBM PC.

Session No.	Topics to be covered	Time	Ref	Teaching Method	Testing Method
1	Overview of Language processors	50	1	BB	Discussion
2	Introduction to Data structures	50	1	BB	Discussion
3	Search Data Structures	50	1	BB	Discussion
4	Scanning and Parsing	50	1	BB	Discussion
5	Elements of assembly language programming	50	1	BB	Discussion
6	Simple assembly scheme	50	1	BB	Discussion
7	Assemblers: Pass Structure – Data structures needed for design	50	1	BB	Discussion
8	Assemblers: Design of two pass assemblers	50	2	BB / PPT	Group discussion
9	Single pass assemblers	50	2	BB / PPT	Group discussion
10	Lab Session – Experimenting Data Structures	50	2	Laboratory	Implementation through programs

11	Lab Session – Searching and Sorting	50	2	Laboratory	Implementation through programs
12	Lab Session – Assembler Design	50	2	Laboratory	Implementation through programs
13	Lab Session – Assembler Design	50	2	Laboratory	Implementation through programs
LOADERS AND LINKERS (15 hours) Macro and Linkers: Macro definition and call, Macro expansion, Nested macro calls, Advanced macro facilities, Design of preprocessor, Relocation and linking concepts, Design of a linker, Self relocating program, Linker for MS-DOS, Linking for overlays, Loaders.					
14	MACRO: Macro definition- macro call – macro expansion	50	1	BB	Discussion
15	Nested macro	50	1	BB / PPT	Discussion
16	Advanced macro facilities	50	1	BB / PPT	Discussion
17	Design of Macro pre-processor	50	1	BB / PPT	Group discussion
18	Relocation and linking concepts	50	1	BB	Discussion
19	Design of linker	50	1	BB	Discussion
20	Self relocating programs	50	1	BB / PPT	Discussion
21	Linking in MS-DOS – overlays	50	1	BB	Discussion
22	Linking for overlays- Loaders	50	1	BB / PPT	Discussion
23	Absolute loaders	50	1	BB / PPT	Discussion
24	Relocating loaders	50	1	BB	Discussion
25	Lab Session – Design of Macro processor	50	1	Laboratory	Implementation through programs
26	Lab Session – Design of Macro processor	50	1	Laboratory	Implementation through programs
27	Lab Session – Design of Loader and Linker	50	1	Laboratory	Implementation through programs
28	Lab Session – Design of Loader and Linker	50	1	Laboratory	Implementation through programs
GRAMMARS, EXPRESSIONS & AUTOMATA (16 hours) Context free Language - Context free grammar - regular expression - Recognizing of patterns - finite automation (deterministic & non deterministic) Conversion of NDFA to DFA - Conversion of regular expression of NDFA - minimization of NDFA – Derivation - parse tree - ambiguity – handle – Lexical Analysis.					
29	Introduction to Compiling, Phases of Compiler	50	1	BB	Discussion
30	Languages – Grammars – Types - Linear Grammars	50	1	BB	Discussion
31	Context Free grammar and its usage	50	1	BB	Discussion
32	Regular Expression – Recognition of patterns – Finite Automation	50	1	BB	Illustration by examples
33	NFA, DFA, Conversion of NDFA to DFA	50	1	BB	Illustration by examples

34	Conversion of Regular expression to NDFA – Thompson construction	50	1	BB / PPT	Illustration by examples
35	More examples on NDFA to DFA	50	1	BB / PPT	Illustration by examples
36	Minimization of DFA – Derivation – Parse tree - Ambiguity	50	1	BB	Illustration by examples
37	Lexical analysis, handles , token specification – Design of Lexical analysis	50	1	BB	Discussion
38	Lab Session – Generation of Regular Expressions	50	1	Laboratory	Implementation through programs
39	Lab Session – Regular Expression to NDFA	50	1	Laboratory	Implementation through programs
40	Lab Session – Regular Expression to NDFA	50	1	Laboratory	Implementation through programs
41	Lab Session NDFA to DFA	50	1	Laboratory	Implementation through programs
42	Lab Session - NDFA to DFA	50	1	Laboratory	Implementation through programs
43	Lab Session - Minimization of DFA	50	1	Laboratory	Implementation through programs
44	Lab Session – Lexical Analyzer	50	1	Laboratory	Implementation through programs
SYNTAX ANALYSIS (16 hours) Role of parsers - Top down parsing : Left recursion - left factoring - Handle pruning , predictive parsing - recursive descent parsing – Bottom up parsing: Shift reduce parsing - operator precedence parsing - LR parsing – LR (0) items - SLR parsing – Canonical LR parsing -LALR parsing					
45	Introduction to Parsing, Top down and Bottom up parsing	50	1	BB	Discussion
46	Left Recursion – Left Factoring – Handle Pruning	50	1	BB / PPT	Discussion
47	Illustration by examples	50	1	BB / PPT	Discussion
48	Shift Reduce Parsing	50	1	BB / PPT	Group discussion
49	Operator Precedence Parsing, Precedence Matrix and Precedence functions	50	1	BB / PPT	Group discussion
50	FIRST and FOLLOW procedures	50	1	BB / PPT	Group discussion
51	More examples on FIRST and FOLLOW	50	1	BB / PPT	Group discussion
52	LEADING- TRAILING	50	1	BB	Illustration by examples
53	Predictive parsing, Parsing table construction	50	1	BB / PPT	Group discussion
54	Recursive Descent Parsing	50	1	BB	Group discussion
55	Simple LR Parsing, LR(0) items,	50	1	BB / PPT	Group discussion
56	Parsing table generation - SLR parsing	50	1	BB / PPT	Group discussion
57	Lab Session – Left Recursion & Left Factoring	50	1	Laboratory	Implementation through programs
58	Lab Session – Predictive parsing	50	1	Laboratory	Implementation through programs
59	Lab Session LEADING- TRAILING	50	1	Laboratory	Implementation through programs
60	Lab Session - LR(0) items	50	1	Laboratory	Implementation through programs

SYNTAX DIRECTED TRANSLATION & CODE OPTIMIZATION (15 hours)

Intermediate Languages - Quadruple - triple - indirect triples - three-address code- Introduction – Syntax tree- DAG - S-attribute - R-attributes - Assignment statement schemes - Back patching - Syntax free construction - CASE statements – Symbol Table - Symbol table contents - data structure for symbol tables - storage allocation - Runtime storage management. Sources of optimization - Loop optimization – DAG representation of basic block - Dominators - flow graphs - object program – problems in code generation - machine model - simple code generator - Code generation from DAG - peephole optimization.

61	Intermediate Languages : prefix - postfix - Quadruple - triple - indirect triples –	50	1	BB	Discussion
62	syntax tree- Evaluation of expression - three-address code - DAG	50	1	BB	Discussion
63	Synthesized attributes – Inherited attributes – Conversion of Assignment statements	50	1	BB	Discussion
64	Boolean expressions – Control statements Backpatching – CASE statements	50	1	BB	Discussion
65	Symbol table contents - data structure for symbol tables -	50	1	BB	Discussion
66	storage allocation - Runtime storage management.	50	1	BB	Discussion
67	CODE OPTIMIZATION: Local optimization-	50	1	BB	Illustration by examples
68	Loop Optimization techniques	50	1	BB	Illustration by examples
69	DAG representation – Dominators- Flow graphs	50	1	BB	Illustration by examples
70	Storage allocations	50	1	BB	Group discussion,
71	Peephole optimization	50	1	BB	Illustration by examples
72	Issues in Code Generation - machine model	50	1	BB	Discussion
73	simple code generator - Code generation from DAG	50	1	BB	Discussion
74	Lab Session – Intermediate code generation – Syntax tree	50	1	Laboratory	Implementation through programs
75	Lab Session - Few Optimization techniques	50	1	Laboratory	Implementation through programs

Signature of Faculty Member

Head of the Department