JSS Mahavidyapeetha

JSS Science And Technology University (Established Under JSS Science and Technology University Act No. 43 of 2013) (Formerly Known as SJCE)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING Lesson Plan for the Academic year 2020-21

Faculty: Dr. M A Anusuya, Prof. Sheela N and Prof. Vani Ashok

Class & Section: IV Semester

Subject with Code: System Programming Lab (CS47L)

Pre-requisite: Theory of Computations, Data structure, Programming in C.

1. Count the number of vowels and consonants in a given string. 2. Create pattern to Recognize and develop a LEX program to count the number of i) Positive and Negative integers and ii) Positive and Negative Fractions. 2. 3. Write a LEX program to recognize a valid C program and also check for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 3. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4. 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands c) Unix Shell programming commands	Week	Programs	Remarks
2. Create pattern to Recognize and develop a LEX program to count the number of i) Positive and Negative integers and ii) Positive and Negative Fractions. 3. Write a LEX program to recognize a valid C program and also check for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4			
2. Create pattern to Recognize and develop a LEX program to count the number of i) Positive and Negative integers and ii) Positive and Negative Fractions. 3. Write a LEX program to recognize a valid C program and also check for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4	1	1. Count the number of vowels and consonants in a given string.	
i) Positive and Negative integers and ii) Positive and Negative Fractions. 2			
ii) Positive and Negative Fractions. 3. Write a LEX program to recognize a valid C program and also check for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands		number of	
3. Write a LEX program to recognize a valid C program and also check for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. Design a YACC program to recognize the grammar {a ⁿ b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands		 i) Positive and Negative integers and 	
for valid loop structure in it. 4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4. 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands		ii) Positive and Negative Fractions.	
4. Recognize a valid Arithmetic expression and also Count number of operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 4. 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {a ⁿ b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands	2	3. Write a LEX program to recognize a valid C program and also check	
operators present and print them separately. 5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands			
5. Write a LEX program to recognize and count the number of identifiers in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands			
in a file. 6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {a^n b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands			
6. Program to count the number of operators and operands in a given valid expression. Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {a^n b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands	3		
expression. Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {a^n b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands			
Complier Design: YACC Programs 1. Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5. 3. Design a YACC program to recognize the grammar {an b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7. Shell Programming: a) Unix Commands b) Vi Commands			
 Recognize a valid and Evaluate Arithmetic expression that uses operators +, -, *, /. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. Design a YACC program to recognize the grammar {aⁿ b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. Recognize valid declaration and definition statement in C Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Shell Programming: Unix Commands Vi Commands 			
operators +, -, *, /. 2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5			
2. Recognize nested IF control statements and displays the number of levels of nesting in the nested IF. 5 3. Design a YACC program to recognize the grammar {a^n b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 6 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7 Shell Programming: a) Unix Commands b) Vi Commands	4		
levels of nesting in the nested IF. 3. Design a YACC program to recognize the grammar {a^n b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands			
 3. Design a YACC program to recognize the grammar {aⁿ b \ n>=0}. Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands 			
Verify the following string belongs to this grammar: (i) a (ii) ab (iii) aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: 7 Shell Programming: a) Unix Commands b) Vi Commands	_		
aaab (iv) abb. 4. Recognize valid declaration and definition statement in C 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands	5		
4. Recognize valid declaration and definition statement in C 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands			
 5. Develop C program to construct the top down parsing table for any grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming:			
grammar and also Parse the input generated by the grammar. 6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands	6		
6. Develop C program to construct the any one bottom up parsing table for any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands	U		
any grammar and also Parse the input generated by the grammar. Operating System: Shell Programming: a) Unix Commands b) Vi Commands			
Operating System: 7 Shell Programming: a) Unix Commands b) Vi Commands			
7 Shell Programming:	Operati		
a) Unix Commandsb) Vi Commands	-	<u> </u>	
b) Vi Commands	-		
		,	
		,	

		1
	a) Concatenation of two strings	
	b) Comparison of two strings	
	c) Maximum of three numbers	
	d) Fibonacci series	
	e) Arithmetic operation using case	
8	System Calls	
	a. Process Creation	
	b. Executing a command	
	c. Sleep command	
	d. Sleep command using getpid	
	e. Signal handling using kill	
	f. Wait command	
9	I/O System Calls	
	a) Reading from a file	
	b) Writing into a file	
	c) File Creation	
10	a)Implementation of ls command	
	b)Implementation of grep command	
11	Given the list of processes, their CPU burst times and arrival times,	
	display/print the Gantt chart for FCFS and SJF. Print average. waiting time	
	and turnaround time.	
12	Given the list of processes, their CPU burst times and arrival times,	
	display/print the Gantt chart for Priority and Round robin. Print average.	
	waiting time and turnaround time.	

Course Outcomes: After completion of course, the students are able to:

CO1:	Analyze unix commands, system calls, shell scripts
CO2:	Analyze and implement CPU Scheduling Algorithms.
CO3:	Apply the knowledge of regular expressions to recognize the tokens generated by the
	Lexical analyzer.
CO4:	Implement the LEX and YACC programs to recognize, validate and evaluate arithmetic
	Expressions and grammars.
CO5:	Design and Implement the various types of parsers for the given context free grammar

Text Books:

- 1. Alfred V Aho, Monica S. Lam, Ravi Sethi, Jeffrey D Ullman: Compilers-Principles, Techniques and Tools, 2nd Edition, Pearson education, 2014.
- 2. Abraham Silberschatz, Peter Baer Galvin, Greg Gagne: Operating System Principles, 9th Edition, Wiley India, 2013
- 3. P.C.P. Bhatt: Introduction to Operating Systems: Concepts and Practice, 2nd Edition, PHI, 2008.
- 4. Abraham Silberschatz, Peter Baer Galvin, Greg Gagne: Operating System Principles, 9th Edition, Wiley India, 2013

Signature of faculties

Signature of HOD