** ACS COLLEGE OF ENGINEERINGC:\Documents and Settings\Administrator\Desktop\update kk\New Folder\naac-a-new.png**

**#207, Kambipura, Mysore road, Banglore-74**

**Department of Computer Science & Engineering**

**QUESTION BANK**

Sem/Subject/Code: **6th/SYSTEM SOFTWARE & COMPILER DESIGN/15CS63/17CS63**

Prepared By**: SUNITA CHALAGERI**

Year : **2019-20(EVEN)**

**Module-1**

**Assembler:**

1. Bring out the differences between system software and application software with examples.
2. Briefly explain the SIC/XE machine architecture.
3. Briefly explain the SIC machine architecture.
4. Generate the object code for the source program given below.
5. What is the role of assembler? Write assembler functions.
6. Write and explain the algorithm for a PASS-1 of a two-pass assembler.
7. Write and explain the algorithm for a PASS-2 of a two-pass assembler.
8. What is program relocation? Explain the problems associated with it and their solutions.
9. Give the simple object program format. and explain.

**Sol :**

1. Header record
2. Text record
3. End record
4. Explain the different data structure used in designing SIC assembler.

**Sol :**

1. LOCCTR
2. SYMTAB
3. OPTAB
4. Explain Machine independent assembler features.

**Sol:**

1. Literals
2. Symbol defining statement
3. Expression
4. Program blocks
5. Control section & program linking
6. Explain the multi pass assembler with example.
7. With suitable example, explain the use of LTORG assembler directive.
8. Explain load and go assembler with an example.
9. Differentiate between literal and immediate operand with example.
10. With an example Explain simple I/O operation of SIC/XE.
11. Compare a two-pass assembler with a one-pass assembler. How forward reference are handled in one pass assemblers?
12. Discuss different design options of assembler.
13. What are assembler directives?

Explain assembler directives?

1. Write a note on MASM assembler
2. How is Relocation is done using the following
3. Modification records
4. Bit masks
5. What is the difference between the instructions?

LDA #3 and LDA THREE

**Sol**:

In the first intn immediate addressing is used. Here the value 3 is directly loaded into the accumulator register where as in the second intn the direct addressing is used. Here value assigned for the symbol THREE is loaded into the accumulator register.

1. Write a note on basic macro processor functions.

**Module-2**

**Loader and Linker:**

1. What is a Loader? Develop an algorithm for a boot strap loader.
2. What is dynamic loading? What are its advantages & disadvantages? Explain with a neat diagram loading and calling of a subroutine using dynamic linking.
3. Write and Explain Boot strap loader for SIC/XE.
4. Explain the machine dependent loader features.
5. Explain various data structures used for a linking loader.
6. What is relocating loader? Explain the creation of object program with relocation by bit mask.(p140)
7. What is relocating loader? Explain the creation of object program with relocation by modification records. (p138)
8. With examples explain any five loader options.(p156)
9. What are basic functions of a loader? Develop and algorithm for a boot strap loader.
10. What is the roll of loader? Explain the design of absolute loader.
11. Write algorithm for an absolute loader.(p132)
12. Write the SIC /XE source code for a simple bootstrap loader.(p134)
13. Explain dynamic linking with suitable diagram.(p164)
14. Explain the facilities available in MS-DOS linker for program linking.(p167)
15. Write an algorithm, explain pass 1 of a linking loader.
16. Write an algorithm, explain pass 2 of a linking loader.
17. Explain various loader design options.
18. Linkage editor
19. Dynamic linking
20. Explain machine independent loader features.
21. What is relocating loader? Explain the methods of specifying relocation as a part of object program.

1. Modification of records

2. Use of relocation bits

**MODULE : 3**

**Lexical Analysis**

1. Explain the phases of compiler with examples.
2. Write a note on compiler constructions tools.
3. What are the applications of compiler?
4. What is lexical analysis and explain its role.
5. Define tokens patterns and lexemes with example.
6. Why analysis portions of compiler is separated into lexical analysis and syntax analysis.
7. What is input buffering? Explain input buffering strategy used in lexical phase.
8. Write the algorithm for constructing look ahead code with sentinels, define sentinels.
9. Explain the use of sentinels in recognizing the tokens (or) what is the disadvantage of input buffering with buffer pairs.
10. Define alphabets, strings. Language, regular expressions.
11. What are the rules for constructing regular expressions?
12. Give the regular definitions for an unsigned number and for an identifier.
13. How do you recognize tokens, explain with diagrams, explain about transition diagram (or) define transitions algorithm an construct transition diagram for relational operation ,identifier, keywords, unsigned nos, void spaces and sketch program segment.
14. List the operation on language and explain with example.
15. Explain error recovery strategy.

**Module:4**

**Syntax analysis:**

1. Give the rules for constructing FIRST and FOLLOW sets.
2. Practice problems on construction of predictive parsing label (LL(1) parsing label) eg. Construct the predictive parsing label by making necessary changes to the grammar given.
3. E->E+T|T
4. T->T\*F|F
5. F->(E)|id
6. Give formal definition of a context free grammar.
7. Give notational conventions for grammars.
8. CFG vs RE
9. Consider the CFG

S->SS+|SS\*| a and string aa+a\*

* 1. Give a left most derivation for the string
  2. Give a rightmost derivation for the string
  3. Give a parse tree for the string
  4. Is the grammar ambiguous or unambiguous? Justify your
  5. Describe the language generated by this grammar.

1. Eliminating ambiguity from the if then else grammar.
2. Describe algorithm used for eliminating the left recursion. Eliminate left recursion from the grammar.
3. Show the following grammar is ambiguous E->E+E|E\*E|(E)|id . write an equivalent unambiguous grammar for the same.
4. What are the key problems with top down parser? Write a recursive descent parser for the grammar. S->cAd A->ab|a
5. Describe algorithm used for left – factoring. Do left factoring for the given grammar.
6. Algorithm for construction of predictive parsing table.
7. Algorithm for table- driven predictive parsing. (Predictive parsing algorithm).
8. Error recovery in predictive parsing.
9. Definition of handle pruning. Construct bottom – up parse tree for the input string

W = aaa\*a++ using the grammar

S->ss+|ss\*|a.

1. What is shift –reduce parser? Explain the conflicts that may occur during shift reduce passing. List the actions of shift - reduce parser.
2. Solve problems on shift reduce parser.
3. With a diagram, explain the model of an LR- parser.
4. Algorithm to find CLOSURE AND GOTO or definitions of GOTO AND CLOSURE
5. Algorithm for construction of canonical collection of sets of LR(O) items .
6. Consider the grammar.

S->L=R|R

L+\*R|ID

R->L

1. Obtain LR(O) ITEMS
2. Compute FIRST & FOLLOW
3. Obtain SLR passing table
4. Check whether the given grammar is SLR (or) not ?

[note : practice problems like above]

1. Algorithm for constructing SLR – Parsing Table

**Module :5**

**Syntax Directed Transaction (SDT)**

1. Define inherited and synthesized attributes. Give SDT for a simple desk calculator.
2. What is syntax directed definitions? & write annotated parse tree for a-4+c using top-down approach.
3. Give the syntax directed definition to process a simple variable declaration in c & construct dependency graph for the input **float x,y,z.**
4. What is syntax Directed Transaction? Write the applications of SDT.
5. Write the grammar & syntax directed definitions for a simple desk calculator & show annotated parse tree for the expressions 3\*5+4n.
6. Write the grammar & give the syntax directed transaction of type **int [2][3]** ant also give the semantic rules for the respective productions.
7. Write the grammar & SDD that generates either a **basic type (or) an array type.**
8. Write a note on the evaluation order of SDD’s (or) what are dependency graph ? explain
9. Explain the concepts of syntax directed definition .

Discuss S-attributes L-attributes with respect to SDD.

1. For the given productions shown below, write semantic rules & construct annotated parse tree for 3\*5+4n.

L🡪 En

E🡪E1+T

E🡪T

T🡪 T1\*F

T🡪 F

F🡪(E)

F🡪digit

1. Obtain the SDD for **simple type declaration** construct a dependency graph for the declaration **int a,b,c** along with evaluation order

**Intermediate code generation:**

1. Explain how DAG will help in intermediate code generation. construct a DAG and a write three-address code for the expression a+a\*(b-c)+(b+c)\*d.
2. Explain the following with an example

Quadruples, triples and Indirect triples.

1. Explain syntax directed translation of switch statement.
2. What are three address codes? explain different ways of representing 3-address codes with examples.

**Code generation:**

1. Discuss the issues in the design of code generator.
2. Write an intermediate code to set a 10\*10 matrix and identify basic blocks.

OR

Write intermediate code for the following code

For i=1 to 10 do

For j=1 to 10 do

A[i,j]=0.0

For i=1 to 10 do

A[i,j]=1.0

1. What are basic blocks and flowgraphs and how do you partition a three address code into basic block. Write algorithm for it.
2. Write a note on simple target machine model.