INSTITUTE OF AERONAUTICAL ENGINEERING

(Autonomous)

Dundigal, Hyderabad - 500 043

INFORMATION TECHNOLOGY

TUTORIAL QUESTION BANK

Course Name	**AUTOMATA AND COMPILER DESIGN**
Course Code	A50513
Class	III B. Tech I Semester
Branch	Information Technology
Year	2017-2018
Course Coordinator	Mr. D Rahul, Assistant Professor
Course Faculty	Mr. D Rahul, Assistant Professor

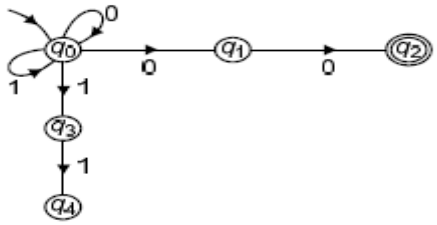
OBJECTIVES

To meet the challenge of ensuring excellence in engineering education, the issue of quality needs to be addressed, debated and taken forward in a systematic manner. Accreditation is the principal means of quality assurance in higher education. The major emphasis of accreditation process is to measure the outcomes of the program that is being accredited.

In line with this, Faculty of Institute of Aeronautical Engineering, Hyderabad has taken a lead in incorporating philosophy of outcome based education in the process of problem solving and career development. So, all students of the institute should understand the depth and approach of course to be taught through this question bank, which will enhance learners learning process.

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
	UNIT-I		
	PART – A (SHORT ANSWER QUESTIONS)		
1	**Define** DFA with example	Remember	1
2	**Explain** transition diagram, transition table with example	Remember	1
3	**Define** Kleene closure.	Remember	1
4	**Explain** the different Operations on the languages	Remember	1
5	**Define** NFA with example	Remember	1

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
6	**Write** any four differences between DFA and NFA	Remember	1
7	**List** any two applications of finite automata	Remember	1
8	**Define** regular expression with example	Understand	1
9	**List** any four identity rules	Understand	1
10	**Define** Complier	Understand	1
11	**List** the phases of a compiler	Understand	1
12	**Explain** a lexeme and define regular sets	Remember	1
13	**Define** the term Symbol table	Understand	1
14	**Define** the term Interpreter?	Remember	1
15	**Explain** about parser and its types	Understand	1
16	**Define** context free grammar	Remember	1
17	**Define** a parse tree?	Understand	1
18	**Explain** an ambiguous grammar with an example	Understand	1
19	**List** the types of derivations	Understand	1
20	**Define** LL(K) grammar	Understand	1
PART – B (LONG ANSWER QUESTIONS)			
1	**Construct** a DFA to accept set of all strings ending with 010. Define language over an alphabet and write for the above DFA.	Remember	1
2	**Construct** a finite automata accepting all the strings over {0, 1} having even number of 0's and even number of 1's.	Remember	1
3	**Construct** the DFA that accepts/recognizes the language $L(M) = \{w \mid w \in \{a, b, c\}^*$ and w contains the pattern $abac\}$. Draw the transition table.	Remember	1
4	**Differentiate** between DFA and NFA with examples	Understand	1
5	**Write** the DFA that will accept those words from $\in \{a, b\}$ where the number of a's is divisible by two and the number of b's is divisible by three. Sketch the transition table of the finite Automaton M.	Remember	1

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
6	**Construct** DFA for the given NFA as shown in fig. below 	Remember	1
7	**Convert** Regular Expression 01* + 1 to Finite Automata.	Understand	1
8	**Construct** a NFA with Є equivalent to the regular expression 10 + (0 + 11)0*1	Understand	1
9	**Define** compiler? State various phases of a compiler and explain them in detail.	Understand	1
10	**Explain** the various phases of a compiler in detail. Also write down the output for the following expression after each phase a: =b*c-d.	Understand	1
11	**Explain** the role Lexical Analyzer and issues of Lexical Analyzer.	Remember	1
12	**Differentiate** the pass and phase in compiler construction?	Remember	1
13	**Construct** Leftmost Derivation. , Rightmost Derivation, Derivation Tree for the following grammar $G = (V, T, P, S)$ with $N = \{E\}$, $S = E$, $T = \{id, +, *, (,)\}$ E→ E+E E→ E*E E→ (E) E→ id Obtain id+id*id in right most derivation, left most derivation	Understand	1
14	**Write** a CFG that generates equal number of a's and b's.	Knowledge	1
15	**Construct** context free grammar which generates palindrome strings ∑={a,b}	Knowledge	1
16	**Explain** the general format of a LEX program with example?	Understand	1
17	**Construct** the predictive parser the following grammar: S->(L)|a L->L,S|S Construct the behavior of the parser on the sentence (a, a) using the grammar specified above	Knowledge	1

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
18	**Explain** the algorithm for finding the FIRST and FOLLOW positions for a given non-terminal. Consider the grammar, D→Type List; List→id Tlist Tlist → ,Tlist/ ε Type→int/float **Construct** a predictive parsing table for the grammar given above. Verify whether the input string id + id * id is accepted by the grammar or not.	Understand	1
19	**Explain** the algorithm for finding the FIRST and FOLLOW positions for a given non-terminal. Consider the grammar, E →TE E→+TE |ε T →FT T→*FT| ε F→(E)|id. **Construct** a predictive parsing table for the grammar given above. Verify whether the input string id + id * id is accepted by the grammar or not.	Understand	1
20	**Analyze** whether the following grammar is LR(1) or not. Explain your answer with reasons. **S→ L,R** **S→ R** **L → * R** **L→ id** **R→ L.**	Knowledge	1
PART – C (PROBLEM SOLVING AND CRITICAL THINKING QUESTIONS)			
1	Consider the following fragment of C code: float i, j; i = i*70+j+2; **Write** the output at all phases of the compiler for above C code.	Knowledge	1
2	**Construct** an NFA for regular expression R= (aa | b) * ab convert it into an equivalent DFA.	Remember	1
3	**Describe** the languages denoted by the following regular expressions. i. (0+1)*0(0+1)(0+1) ii. 0*10*10*10*	Remember	1

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
4	**Explain** with one example how LEX program perform lexical analysis for the following PASCAL patterns Identifiers, Comments, Numerical constants, Keywords, Arithmetic operators?	Knowledge	1
5	**Check** whether the following grammar is a LL(1)grammar S-> iEtS|iEtSeS|a E-> b Also define the FIRST and FOLLOW.	Knowledge	1
6	**Consider** the grammar below E->E+E|E-E|E*E|E/E|a|b Obtain left most and right most derivation for the string a+b*a+b.	Knowledge	1
7	**Define** ambiguous grammar? Test whether the following grammar is ambiguous or not. E->E+E|E-E|E*E|E/E|E↑|(E)|-E|id	Knowledge	1
8	**State** the limitations of recursive descent parser?		1
9	**Convert** the following grammar into LL(1)grammar S->ABC A->aA|C B->b C->c.	Knowledge	1
10	**Write** a recursive descent parser for the grammar. bexpr->bexpr or bterm|bterm bterm->bterm and bfactor|bfactor bfactor->notebfactor|(bexpr)|true|false. Where ,or, and , not,(,),true, false are terminals of the grammar.	Knowledge	1
UNIT – II			
PART – A (SHORT ANSWER QUESTIONS)			
1	**Explain** bottom up parsing	Remember	2
2	**Define** LR(0) items in bottom up parsing	Remember	2
3	**List** different bottom up parsing techniques	Remember	2
4	**Define** LR(k) parsing	Understand	2
5	**Explain** why LR parsing is attractive one and explain	Understand	2
6	**Differentiate** top down and bottom up parsing	Understand	2
7	**Define YACC**	Understand	2
8	**Define** goto function in LR parser with an example	Understand	2
9	**Explain** types of LR parsers?	Understand	2
10	**List** down the conflicts during shift-reduce parsing.	Remember	2
11	**Define** syntax directed translation	Understand	2
12	**Define** synthesized attribute	Understand	2
13	**Define** inherited attribute	Understand	2
14	**Define** abstract syntax tree	Understand	2
15	**List** the forms of intermediate code	Understand	2

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
17	**Define** quadruple with example	Understand	2
18	**Define** triple with example	Understand	2
19	**Define** indirect triple with example	Understand	2
20	**Explain** Error recovery in LR parsing	Understand	2
PART – B (LONG ANSWER QUESTIONS)			
1	**Explain** the parsing ambiguous grammar E→E + E/E *E/(E)/id.	Knowledge	2
2	**Prepare** LR(0) items for E → E + T/T T → T F/F F→F*/a/b	Knowledge	2
3	**Prepare** a canonical parsing table for the grammar given below S→Aa/bAc/bBa A→d B→d	Knowledge	2
4	**Prepare** a canonical parsing table for the grammar given below S→ CC C → cC|d	Knowledge	2
5	**Prepare** LR(1) items for the grammar S→L=R S→ R L→ * R L→id R → L	Knowledge	2
6	**Prepare** a LALR parsing table for the grammar given below S→CC C → cC|d	Knowledge	2
7	**Analyze** whether the following grammar is SLR(1) or not. Explain your answer with reasons. S→L,R S→ R L→ * R L→ id R→L.	Knowledge	2

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
8	**Consider** the grammar given below. E → E + T E → T T → T * F T→F F→ (E) F→ id Prepare LR parsing table for the above grammar .Give the moves of LR parser on id * id + id	Knowledge	2
9	**Explain** handle pruning in detail with example	Understand	2
10	**Explain** S-Attributed and L-Attributed grammars	Understand	2
11	**Explain** ways to determine precedence relations between pair of terminals	Understand	2
12	**Explain** types of three address codes	Understand	2
13	**Write** the quadruple, triple, indirect triple for the statement. a :=b * -c + b * -c	Knowledge	2
14	**Define** syntax tree? Draw the syntax tree for the assignment statement. a :=b * -c + b * -c	Knowledge	2
15	**Explain** LR parsers in detail with example	Understand	2
PART – C (PROBLEM SOLVING AND CRITICAL THINKING QUESTIONS)			
1	**Explain** the common conflicts that can be encountered in a shift-reduce parser?	Knowledge	2
2	**Explain** LALR parsing, justify how it is efficient over SLR parsing.	Remember	2
3	Analyze whether the following grammar is CLR(1) or not. Explain your answer with reasons S -> L,R S->R L-> * R L-> id R -> L.	Apply	2
4	Analyze whether the following grammar is SLR or not. Explain your answer with reasons S -> L,R S->R L-> * R L-> id R -> L.	Apply	2
5	**Discuss** error recovery in LL and LR parsing.	Remember	2
6	**Construct** SLR (1) Parsing table for following grammar s-> xAy/xBy/xAz A->as/q B->q	Remember	2
7	**Construct** SLR Parsing table for following grammar s->0s0/1s1/10	Remember	2

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
8	**Construct** SLR Parsing table for following grammar s->aSbS/bsas/	Remember	2
9	**Construct** LALR (1) Parsing table for following grammar s->Aa/bAc/dc/bda A->d	Remember	2
10	**Construct** LALR (1) Parsing table for following grammar s->Aa/aAc/Bc/bBa A->d B->d	Remember	2
UNIT – III			
PART – A (SHORT ANSWER QUESTIONS)			
1	**Define** Chomsky hierarchy of languages	Remember	3
2	**List** the types of grammars	Remember	3
3	**Define** Type 0 grammars with example	Remember	3
4	**Define** Type 1 grammars with example	Remember	3
5	**Define** Type 2 grammars with example	Remember	3
6	**Define** Type 3 grammars with example	Remember	3
7	**Define** context sensitive grammar with example	Remember	3
8	**Define** context free grammar with example	Remember	3
9	**Define** unrestricted grammar with example	Remember	3
10	**Define** left linear grammar	Remember	3
11	**Define** right linear grammar	Remember	3
12	**Define** Regular grammar with example	Remember	3
13	**Explain** the role of type checker	Remember	3
14	**List** the different type checking techniques	Remember	3
15	**Explain** static type checking	Remember	3
16	**Explain** dynamic type checking	Remember	3
17	**Write** SDD for type checking of statements	Remember	3
18	**Write** SDD for type checking of functions	Remember	3
19	**Explain** type conversion	Remember	3
20	**List different** type conversions	Remember	3
21	**List** advantages and disadvantages of type conversion by programmer	Remember	3
22	**Write** a short note on type conversion by programmer	Remember	3
23	**List** advantages and disadvantages of type conversion by compiler	Remember	3
24	**Write** a short note on type conversion by compiler	Remember	3
25	**Define** type Expression with example	Remember	3
26	**List** different user defined type expressions	Remember	3
27	**Define** type Expression for arrays	Remember	3
28	**Define** type Expression for functions	Remember	3
29	**Define** type Expression for records	Remember	3
30	**Define** type Expression for pointers	Remember	3
31	**Explain** representation of type expression	Remember	3
32	**Define** type Expression for products(user defined)	Remember	3
33	**Define** type system	Remember	3

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
34	**Define** type graph	Remember	3
35	**Explain** the importance of function overloading	Remember	3
36	**Explain** the importance of operator overloading	Remember	3
37	**Explain** the importance of polymorphism	Remember	3
38	**Compare** structural equivalence, name equivalence of type expressions	Remember	3
39	**Explain** importance of equivalence of type expressions	Remember	3
40	**Explain** equivalence of type expressions	Remember	3
41	**Explain** structural equivalence of type expressions	Remember	3
42	**Explain** name equivalence of type expressions	Remember	3
43	**Explain** role of equivalence of type expressions	Remember	3
44	**List** the merits and demerits of structural equivalence of type expressions	Remember	3
45	**List** the merits and demerits of name equivalence of type expressions	Remember	3
46	**Define** polymorphism	Remember	3
47	**Define** operator overloading	Remember	3
48	**Define** function overloading	Remember	3
PART – B (LONG ANSWER QUESTIONS)			
1	**Explain** in detail Chomsky hierarchy of languages with neat diagram	Remember	3
2	**Explain** Type 2 and Type 3 grammars with example	Remember	3
3	**Explain** Type 0 and Type 1 grammars with example	Remember	3
4	**Explain** in detail type checking	Remember	3
5	**Differentiate** static and dynamic type checking	Remember	3
6	**Explain** in detail type conversion with suitable examples	Remember	3
7	**Explain** in detail equivalence of type expressions	Remember	3
8	**Write** short note on function overloading	Remember	3
9	**Write** short note on operator overloading	Remember	3
10	**Differentiate** structural, name equivalence of type expressions	Remember	3
11	**Write** a note on the specification of a simple type checker.	Knowledge	3
12	**Describe** the method of generating syntax directed definition for control Statements?	Understand	3
13	**Define** a type expression? Explain the equivalence of type Expressions with an appropriate example.	Understand	3
14	Explain the importance of type checking	Understand	3
15	Explain the importance of function and operator overloading	Understand	3
16	Explain the importance of type conversion	Knowledge	3
17	**Distinguish** between static and dynamic storage allocation?	Understand	3
18	**Demonstrate** semantic actions in semantic analysis	Understand	3
19	**Explain** translations on parse tree semantic analysis	Understand	3
20	**Explain** type checking in semantic analysis	Understand	3

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
PART – C (PROBLEM SOLVING AND CRITICAL THINKING QUESTIONS)			
1	**Suppose** that the type of each identifier is a sub range of integers, for expressions with operators +, -, *, div and mod, as in Pascal. Write type-checking rules that assign to each sub expression the sub range its value must lie in.	Understand	3
2	**Define** type expression? Write type expression for the following type i. Functions whose domains are functions from integers to pointers to integers and whose ranges are records consisting of an integer and a character.	Understand	3
3	**Write** an S-attributed grammar to connect the following with prefix rotator. L→E E→ E+T|E-T|T T→ T*F|T/F|F F→ P↑F|P P→ (E) P→ ID	Knowledge	3
4	**Construct** triples of an expression: a *- (b + c).	Knowledge	3
5	**Explain** SDD for Boolean expression with and without back patching?	Remember	3
6	**Explain why** are quadruples preferred over triples in an optimizing compiler?	Remember	3
7	**Explain** the following: i) Static and Dynamic Checking of types ii) Over loading of Operators & Functions	Understand	3
UNIT – IV			
PART – A (SHORT ANSWER QUESTIONS)			
1	**Define** the principle sources of optimization	Understand	4
2	**List** different local optimization techniques	Understand	4
3	**List** different Loop optimization techniques	Understand	4
4	**Define** local optimization?	Understand	4
5	**Define** constant folding?	Understand	4
6	**List** the advantages of the organization of code optimizer?	Understand	4
7	**Define** Common Sub expressions?	Understand	4
8	**Explain** Dead Code?	Understand	4
9	**Explain** the techniques used for loop optimization and Reduction in strength?	Understand	4
10	**Define** loop unrolling	Understand	4
11	**List** the different data flow properties	Understand	4
12	**Explain** inner loops, natural loops	Understand	4
13	**List** different parameter passing techniques	Understand	4
14	**Define** activation record	Understand	4

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
15	**Define** busy expression	Understand	4
16	**Define** basic block and flow graph	Understand	4
17	**List** different storage allocation strategies	Understand	4
18	**Explain** about live variable analysis	Understand	4
19	**Explain** static storage allocation	Understand	4
20	**Explain** dynamic storage allocation	Understand	4
PART – B (LONG ANSWER QUESTIONS)			
1	**Explain** the principle sources of code optimization in detail	Understand	4
2	**Explain** peephole optimization?	Understand	4
3	**Discuss** about the following i. Copy propagation ii. Dead code elimination iii. Code motion	Understand	4
4	**Explain** in the DAG representation of the basic block with example.	Understand	4
5	**Explain** Local optimization and loop optimization in detail	Understand	4
6	**Write** about Data Flow Analysis of structural programs	Understand	4
7	**Explain** various Global optimization techniques in detail	Understand	4
8	**Explain** organizing the non local data		4
9	**Discuss** algebraic simplification and reduction in strength	Understand	4
10	**Explain** dynamic storage allocation	Understand	4
11	**Explain** parameter passing in detail	Understand	4
12	**Explain** static storage allocation	Apply	4
13	**Explain** the different storage allocation strategies	Understand	4
14	(a) **Write** the procedure to detect induction variable with example? (b) With example **Explain** dead code elimination?	Understand	4
15	(a) **Explain** how loop invariant computation can be eliminated? (b) **Explain** how "Redundant sub-expression eliminates" can be done in a given program?	Understand	4
16	**Explain** reachable code in code optimization	Understand	4
17	**Explain** characteristics of peep hole optimization	Understand	4
18	**Explain** depth first search in data flow analysis	Understand	4
19	**Differentiate** between stack and heap storage allocation	Understand	4
20	**Explain** handling the local data	Understand	4
PART – C (PROBLEM SOLVING AND CRITICAL THINKING QUESTIONS)			
1	**Explain** how loop invariant computation can be eliminated?	Knowledge	4
2	**Describe** the procedure to compute in and out values using data flow equations for reaching definition in structured programs?	Knowledge	4

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome

3	**Consider** the following part of code.
	int main()
	{
	int n,k=0;
	scanf("%d",&n);
	for(i=2;i<n;i++)
	{
	if((n%I)==0)break;
	}
	k=1;
	if(i==n)
	printf("number is prime");
	else
	printf("number is not printed");
	}
	Identify the basic blocks in the given program & Draw the domination tree for the program
		Understand	4

4	**Construct** the DAG for the following basic block. D:=B*C E:=A+B B:=B+C A:=E-D	Knowledge	4

5	**Generate** target code for the given program segments:
	main()
	{
	int i=4,j;
	j = i + 5;
	}
		Knowledge	4

6	**Consider** the following program which counts the prime from 2 to n using the sieve method on a suitable large array,
	begin read n
	for i:=2 to n do
	a[i]:=true
	count=0;
	for i:=2 to n**.5 do
	if a[i]then
	begin
	count:=2*I to n j=j+1 do
	a[j]:=false end
	i. print count end
	ii. Propagate out copy statements wherever possible.
	iii. Is loop jamming possible? If so, do it.
	iv. Eliminate the induction variables wherever possible
		Knowledge	4

7	**Write** an algorithm to eliminate induction variable?	Knowledge	4
8	**Explain** how the following expression can be converting in to DAG. a+b*(a+b)+c+d	Knowledge	4

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
9	**State** loop invariant computations? Explain how they affect the efficiency of a program?	Understand	4
10	**Explain** how "Redundant sub-expression Eliminates" can be done at global level in a given program?	Understand	4
11	**Explain** role of DAG in optimization with example?	Understand	4
12	**Explain** the use of Symbol table in compilation process? List out various attributes stored in the symbol table?	Understand	4
13	**Explain** the data structure used for implementing Symbol Table?	Understand	4
14	**List** the advantages and disadvantages of Static storage allocation strategies?	Understand	4
15	**Explain** about reusing the storage space for names?	Remember	4
16	**Define** self-organizing lists? How can this be used to organize a symbol table? Explain with an example?	Knowledge	4
17	**Discuss** and analyze about all allocation strategies in run-time storage environment?	Understand	4
18	**Define** activation records? Explain how it is related with run-time storage organization?	Remember	4
19	Only one occurrence of each object is allowable at a given moment during program execution. **Justify** your answer with respect to static allocation?	Knowledge	4
UNIT – V			
PART – A (SHORT ANSWER QUESTIONS)			
1	**Define** about machine dependent code generation	Remember	5
2	**Explain** the role of code generator in a compiler	Understand	5
3	**List** the issues in the design of code generator	Understand	5
4	**List** different object code forms	Understand	5
5	**Explain** the instructions and address modes of the target machine	Understand	5
6	**Define** absolute machine code	Understand	5
7	**Define** relocatable machine code	Understand	5
8	**Define** assembly language code	Understand	5
9	**List different** machine dependent optimization techniques	Remember	5
10	**Define** address descriptor with example	Remember	5
11	**Define** code descriptor with example	Remember	5
12	**Explain** how do you calculate the cost of an instruction	Understand	5
13	**Generate** the code for x: =x+y-z for target machine	Understand	5
14	**Generate** the code for x: =x+1 for target machine	Understand	5
15	**Explain** the input taken by code generation algorithm	Understand	5
16	**Define** DAG with example	Understand	5
17	**Mention** the applications of DAG	Knowledge	5
18	**Define** register allocation and assignment	Understand	5

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
19	**List** different register allocation techniques	Understand	5
20	**Demonstrate** global register allocation with example	Understand	5
PART – B (LONG ANSWER QUESTIONS)			
1	**Explain** the concept of object code forms	Understand	5
2	**Explain** different issues in design of code generator	Understand	5
3	**Explain** Machine dependent code optimization in detail with an Example	Understand	5
4	**Write** a short note on code generating algorithms	Understand	5
5	Explain in detail register allocation and assignment	Understand	5
6	**Write** about target code forms and explain how the instruction forms effect the computation time	Understand	5
7	**Explain** in detail about DAG for register allocation with example	Knowledge	5

8	**Generate** optimal machine code for the following C program.
	main()
	{
	int i, a[10];
	while (i<=10)
	a[i] =0;
	}
		Knowledge	5

9	**Show** the code sequence generated by the simple code generation Algorithm u := a – c v := t + u d := v + u//d	Knowledge	5
10	**Identify** the register descriptor target code for the source language statement "(a-b) + (a-c) + (a-c);" The 3AC for this can be written as t := a – b		5
11	**Demonstrate** register allocation by graph coloring	Understand	5
12	**Explain** the steps involved in Dag construction	Understand	5
13	**Demonstrate** code generation algorithm in detail	Understand	5
14	**Generate** optimal machine code for the X:=(A-B)+(A-C)+(A-C)	Understand	5
15	**Explain** code generation from DAG using rearranging order algorithm	Understand	5
PART – C (PROBLEM SOLVING AND CRITICAL THINKING QUESTIONS)			
1	**Explain** how the instruction forms effect the computation time?	Knowledge	5

S. No.	Questions	Bloom's Taxonomy Level	Course Outcome
2	**Explain** how the nature of the object code is highly dependent on the machine and the operating system?	Knowledge	5
3	**Explain** why Next-use information is required for generating object code?	Knowledge	5
4	Efficient code generation requires the Remember of internal architecture of the target machine. **Justify** your answer with an Example?	Understand	5

5	**Generate** optimal machine code for the following wing c program.
	main()
	{
	int i,a[10];
	while(i<=10)
	a[i]=0;
	}
		Knowledge	5

6	**Generate** 3 address code for below code X = (a+b)- /((c+d)-e)	Knowledge	5

7	**Generate** 3 address code for below code
	For(i=1;i<=10;i++)
	If(a<b) then x = y + z
		Knowledge	5

8	**Generate** 3 address code for below code
	If a < b then
	While c > d do
	x = x+y
	else
	do
	p = p+q
	while e<=f
		Knowledge	5

9	**Generate** 3 address code for below code X = 1 X = y X = x++	Knowledge	5

10	**Generate** 3 address code for below code
	main( )
	{
	int i;
	int a[10];
	While(i<=0)
	a[i]=0;
	}
		Knowledge	5

Prepared by: Mr D Rahul, Assistant Professor
Date: 07/07/2017 **HOD, IT**