Code No: R2032052 **R20** SET -1

III B. Tech II Semester Regular/Supplementary Examinations, May/June - 2024
COMPILER DESIGN
(Computer Science and Engineering)

Time: 3 hours Max. Marks: 70

Answer any **FIVE** Questions **ONE** Question from **Each unit**
All Questions Carry Equal Marks
*****

UNIT-I

1. a) What are the different phases of compilation and explain the phases of the compilation with the following C language statement as an input. [7M]
position = initial + rate * 60;
where position, initial, and rate are variables of type double

   b) The programming language may be case sensitive or case in sensitive. Case sensitive languages treat uppercase and lowercase letters as distinct characters in programming whereas case insensitive languages treat both as same. Most of the programming languages are case sensitive, However some languages like SQL are case insensitive so the keyword can be written either in lower case, upper case, or in any mixture case. The SQL keyword **SELECT** can also be written **select**, **Select**, or **SeLeCT**. Show how to write a regular expression for a keyword in a case insensitive language, explain with "select" in SQL [7M]

(OR)

2. a) What is sentinel character in input buffering and explain its significance [7M]

   b) Write a regular expression for recognizing the following patterns where input alphabet {a,b,c} [7M]
      i) Which begins with 'a' and ends with 'c'
      ii) Which begins and ends with same symbol

UNIT-II

3. a) Consider the following grammar which is used for specifying subset of arithmetic expressions in C [7M]
A -> A - id | -A | id
      i) Construct a parse tree for the string id-id-id
      ii) Prove that the above grammar ambiguous

   b) Explain with an example "why left recursion is an issue in top down parsing" and write steps for left recursion elimination [7M]

(OR)

4. Construct a LL(1) parsing table for the following grammar and show the working of parser on input -id-id-id [14M]
A -> A -id | B
B -> -A | id

UNIT-III

5. a) Construct CLR(1) parsing table for the following grammar and show the working of parser on input -id-id-id [7M]
A -> A - id | B
B -> -A | id

   b) Explain about SDD's and also evaluation orders for SDD's. [7M]

(OR)

6. a) Explain shift-reduce and reduce-reduce conflicts in the context of shift-reduce parsing. [7M]

   b) Explain S-attributed and L-attributed definitions with an example. [7M]

|''|''|||''|''|||'|

Code No: R2032052 **R20** SET -1

UNIT-IV

7. a) Explain the contents of an activation record with an example. [7M]
   b) What is an induction variable and identify the induction variables in the following statements in C [7M]

```
for( i=0,j=1,k=2,l=4 ; i< n ; i++){
        j= j+2
        k = k+3
        l = l+4
        sum = sum + i*j*k
}
```

(OR)

8. a) Compare static, stack, dynamic storage allocation strategies [9M]
   b) What is dead code elimination explain with an example [5M]

UNIT-V

9. a) Generate simple assembly language target code for the following intermediate code statements using simple code generator. Assume that target machine has three registers (R1, R2, R3). Initially all registers are empty and no variable is live at the end of all statements [9M]

```
T1 = a*b
T2 = a*c
T3 = T1 + b
T4 = a+T2
T5 = b + T4
```

   b) Explain various object code forms used in compilation. [5M]

(OR)

10. a) Explain how does register allocation techniques affect program performance. [7M]
    b) Explain peephole optimization on target assembly language programs with examples. [7M]

Code No: R2032052 **R20** SET -2

III B. Tech II Semester Regular/Supplementary Examinations, May/June - 2024
COMPILER DESIGN
(Computer Science and Engineering)

Time: 3 hours Max. Marks: 70

Answer any **FIVE** Questions **ONE** Question from **Each unit**
All Questions Carry Equal Marks
*****

UNIT-I

1. a) What is boot strapping in the context of compiler and explain how it helps in language independence and reducing development time. [8M]
   b) Write a regular expression for recognizing the following tokens in C [6M]
      i) Identifier ii) integer constant iii) string constant.

(OR)

2. a) What is input buffering in the context of lexical analysis and explain why we take pair of input buffers instead of single buffer in lexical analysis. [7M]
   b) Explain the output of the following Lex specification on input abbbabaa [7M]

```
%%
a*b    {printf("1"); }
ab* {printf("2"); }
b*a {printf("3"); }
ba* {printf("4"); }
%%
```

UNIT-II

3. a) Consider the following grammar which is used for specifying logical expressions in python [7M]

   L ->L *and* L | L *or* L |*not* L | *TRUE* | *FALSE*

      i) Construct parse tree(s) for the string
         not TRUE and FALSE
      ii) Prove that the above grammar ambiguous

   b) Explain with an example "why common left factors is an issue in top down parsing" and write steps for left factoring [7M]

(OR)

4. Construct a LL(1) parsing table for the following grammar and show the working of parser on input*not TRUE and FALSE* [14M]

   L -> L *or*B | B
   B ->B*and*C | C
   C ->*not* L | *TRUE* | *FALSE*

UNIT-III

5. Construct a SLR(1) parsing table for the following grammar and show the working of parser on input *not TRUE and FALSE* [14M]

   L -> L *or* B | B
   B -> B *and*C | C
   C ->*not* L | *TRUE* | *FALSE*

(OR)

6. a) Justify the following statements [9M]
   (i) If there is no shift-reduce conflict in CLR(1) parsing table then there is no shift-reduce conflict in LALR(1) parsing table of same grammar
   (ii)Even If there is no reduce-reduce conflict in CLR(1) parsing table also there maybe reduce-reduce conflict in LALR(1) parsing table of same grammar

|''|''|||''|''|||'|

b) Write quadruple, triple , and indirect triple for the following statement [5M]

x = y*-z + y*-z

UNIT-IV

7. a) Consider the following intermediate code statements numbered from 1 to 12 [7M]

```
1 i=0
2 if i<n goto 4
3 goto 11
4 t1 = 4*i
5 t2 = a[t1]
6 t3 = t2 + 1
7 t4 = 4*i
8 b[t4] = t3
9 i= i+1
10 goto 2
11 t5 = 4*i
12 b[t5] = 0
```

Construct a control flow graph for the given code and explain which of the 4*i computation in statement 7 and statement 11 are redundant.

b) Explain with an example why static allocation strategy is not appropriate for languages like C. [7M]

(OR)

8. a) Draw the activation tree for the function call f(5) where definition of f is given as follows [7M]

```
int f(int n){
if n==0  return 0
else if n==1 return 1
else  return f(n-1) + f(n-2)
}
```

b) Explain different loop optimizations with examples. [7M]

UNIT-V

9. a) Explain about next-use, register descriptor, address descriptor data structures used in simple code generation algorithm. [7M]

b) Write simple code generation algorithm. [7M]

(OR)

10. a) Explain about different forms object code forms used as target code in target code generation. [7M]

b) Explain register allocation by graph coloring. [7M]

|''|''|||''|'''|||'|

Code No: R2032052 **R20** SET - 3

III B. Tech II Semester Regular/Supplementary Examinations, May/June - 2024
COMPILER DESIGN
(Computer Science and Engineering)

Time: 3 hours Max. Marks: 70

Answer any **FIVE** Questions **ONE** Question from **Each unit**
All Questions Carry Equal Marks
*****

UNIT-I

1. a) Compare and contrast compiler and interpreter. [5M]
   b) Define token, pattern, and lexeme. And mark the token, lexeme and pattern required for the following C language statement. [9M]
   printf(" sum of %d and %d is %d",a,b,a+b)

(OR)

2. a) What are the different ways of designing a lexical analyzer and explain each method with an example. [6M]
   b) Write a regular expression for recognizing the following patterns where input alphabet {a,b,c} [7M]
      i) Which contains at least one a and at most one b
      ii) Which contains at lease two a's and at most two b's

UNIT-II

3. a) Consider the following grammar which is used for specifying subset of arithmetic expressions in C, where num is an integer constant [7M]
   E ->num –E | num + E | num
      i) Construct a parse tree for the string
         num – num - num + num
      ii) As per the grammar given above , what is the result of the expression 9-5-2+4
   b) Explain dangling else problem and what is the solution for it as per the C language specification. [7M]

(OR)

4. Construct a LL(1) parsing table for the following grammar and show the working of parser on input aa+a* [14M]
   S-> SS + | SS * | a

UNIT-III

5. Construct a LALR(1) parsing table for the following grammar and show the working of parser on input aa+a* [14M]
   S-> SS + | SS * | a

(OR)

6. a) Compare and contrast SLR(1), CLR(1), LALR(1) parsers. [7M]
   b) Write triples for the following statement. [7M]
   c[i] = a[i] + b[i]
   where a,b,c are arrays of type integers

UNIT-IV

7. a) Show the contents of the activation record for the function call f(6) where definition of f is given as follows and f is called from main function [7M]

```
int f(int n){
if n==0  return1
else  returnn* f(n-1)
}
```

b) Explain in detail common sub expression elimination, copy propagation, dead code elimination optimizations with examples. [7M]

(OR)

8. a) Explain the purpose of live variable data flow analysis with an example. [7M]

b) Write a brief note on structure preserving transfromations. [7M]

UNIT-V

9. a) Explain different addressing modes and how does address modes helpful in improving the performance of the target program. [7M]

b) Explain different machine dependent code optimizations with examples. [7M]

(OR)

10. a) Explain which phases of compilation are machine independent and which are machine dependent? [7M]

b) Explain how relocatable object code form helps in cross-platform compatibility. [7M]

Code No: R2032052 **R20** SET -4

III B. Tech II Semester Regular/Supplementary Examinations, May/June - 2024
COMPILER DESIGN
(Computer Science and Engineering)

Time: 3 hours Max. Marks: 70

Answer any **FIVE** Questions **ONE** Question from **Each unit**
All Questions Carry Equal Marks
*****

UNIT-I

1. a) Define compiler and explain which phases of the compilation aremachine-dependent and which are machine-independent. [6M]
   b) Languages like Fortran ignores spaces and spaces having no significance, Consider the following the statements A and B in Fortan [8M]
   A: DO 5 I = 1.50
   B: DO 5 I = 1,50
   In statement A DO5I is an identifier and In statement B DO is a keyword. How a lexical analyzer differentiates DO in statement A and B explain.

(OR)

2. a) Explain the purpose of lexeme beginning and forward pointers in buffer pairs with an example. [7M]
   b) Explain the output of the following Lex specification on inputbbaabaab [7M]

```
%%
a*b    {printf("1"); }
ab*    {printf("2"); }
b*a    {printf("3"); }
ba*    {printf("4"); }
%%
```

UNIT-II

3. a) Consider the following grammar which is used for specifying subset of arithmetic expressions in C, where num is an integer constant [7M]
   E -> num * E| num/E | num
   i) Construct a parse tree for the string num / num / num * num
   ii) As per the grammar given above, what is the result of the expression 12 / 12 / 2 * 3
   b) Explain with an example "why ambiguity is an issue in parsing" and write an intuitive argument why it is difficult to solve. [7M]

(OR)

4. Construct a LL(1) parsing table for the following grammar and show the working of parser on input ((a,a),a,(a)) [14M]
   S-> (L) | a
   L-> L, S | S

UNIT-III

5. Construct a SLR(1) parsing table for the following grammar and show the working of parser on input ((a,a),a,(a)) [14M]
   S-> (L) | a
   L-> L, S | S

(OR)

6. a) Compare and contrast top-down parsing and bottom-up parsing. [7M]
   b) Explain synthesized and inherited attribute with examples. [7M]

|''|''|||''|''|||'|

Code No: R2032052 **R20** SET -4

UNIT-IV

7. a) Explain the stack allocation strategy with an example. [7M]
   b) Define basic block and write algorithm for constructing control flow graph from the intermediate code. [7M]

(OR)

8. Perform available expression analysis on the following intermediate code statements numbered from 1 to 9 [14M]
   i) Argue whether a*b expression is available at statement 6 and statement 8
   ii) Argue whetherb*c expression is available at statement5 and statement 9

```
1 x = a*b
2 y = b*c
3 if a>20  goto 6
4 z = a*b
5 w = b*c
6 p = a*b
7 a = a+20
8 q = a*b
9 r = b*c
```

UNIT-V

9. a) How register assignment will be done? Explain in detail. [7M]
   b) Explain how assembly code as target code helps in understanding program execution. [7M]

(OR)

10. a) Explain peephole optimization on target assembly language programs with examples. [7M]
    b) Discuss various issues in the design of code generator. [7M]