Roll No.: _

Amrita Vishwa Vidyapeetham

B.Tech. First Assessment Examinations – January 2018

Sixth Semester

Computer Science and Engineering

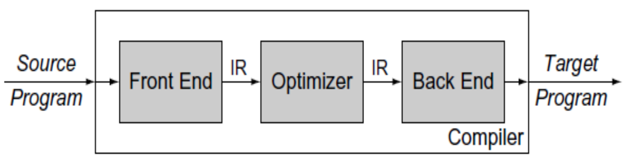
15CSE311 Compiler Design

Time : Two hours Maximum : 50 Marks

Answer all questions

Part-A (10* 2 = 20 Marks)

1. Give the objectives of compiler design and mention if all of them are mandatory.
2. Consider the following compiler structure:



Specify the functionality of front end and back end (each in only one word). What is meant by retargeting the compiler?

3. Fill the missing cells the following table:

Phases	Input	Output	Functionality (max 4 words)
Scanner	Stream of characters		Valid lexeme recognition
Parser		Parse tree	
Semantic analyser			Type checking, Type coercion

4. What is the difference between symbolic register, virtual register and permanent register? Mention the phases in which they are used.
5. Fill in the blanks:
In subset construction, every iteration of the algorithm produce a new subset making it a _______________ increasing function and on termination the function reaches a __________________.
6. Consider the partition p1 = {di, dj, dk}. Each state in p1 has a transition on input symbol “a”: di to dx, dj to dy and dk to dz. In Hopcroft's DFA minimization algorithm, how is p1 split under the following conditions:
   a. When dx, dy and dz are in partitions p2, p3 and p4.
   b. When dx and dy are in partition p2 and dz is in partition p3.

7. In parser design, how are terminals and non terminals specified in programming language specification?
8. Consider the following grammar:
   start → paren | bracket
   bracket → [paren] | [ ]
   paren → (bracket) | ( )
   Check if the following string is derivable: [() . Do the steps of the derivation have a formal name if so, what and if not, why?
9. Consider the following grammar: A→ α | β
   What is FIRST(α) and FIRST^{+}(α)? Mention the LL(1) property.
10. Is it possible to transform all non LL(1) grammars to LL(1) grammar? Mention the transformation methods used to convert non LL(1) grammars to LL(1) grammar.

Part-B (5* 6= 30 Marks)

11. Consider the following grammar:
    S ::= beep | b + B | B | { S }
    B ::= bop + B | bop | { beep }
    a. The following grammar is suspected to be ambiguous. Show that it is, or argue that it can't be.
    b. Does this grammar satisfy the LL(1) condition? Justify your answer. If it does not, rewrite it as an LL(1) grammar for the same language.

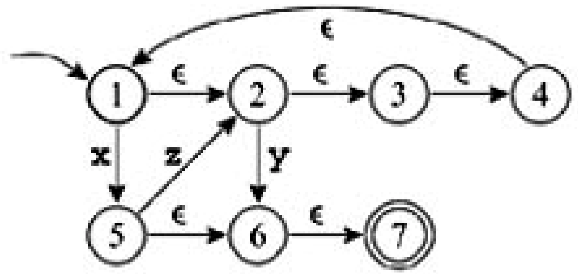
12. Consider the following grammar (non terminals are A, B, C and terminals are d,e,f):
    A → Cd
    B → Ce
    C → A | B | f
    Show the step-by-step of implementation the following indirect left recursion elimination algorithm:

```
impose an order on the nonterminals, A1, A2, ..., An

for i ← 1 to n do;
    for j ← 1 to i - 1 do;
        if ∃ a production Ai→Aj γ
            then replace Ai→Aj γ with one or more
                 productions that expand Aj
    end;
    rewrite the productions to eliminate
         any direct left recursion on Ai
end;
```

Show the rules to eliminate the direct left recursion and remove the left recursion in the evolved grammar.

13. Draw the NFA using Thompson's construction algorithm as in the Keith Cooper textbook for the regular expression: (*0* | [*1. . . 9*] [*0. . . 9*]*) (ε | **.** [*0. . . 9*]*)
14. Construct a DFA for the following NFA using the Subset construction algorithm and minimize the obtained DFA using Hopcroft's algorithm:



15. Suppose an elevator is controlled by two commands: ↑ to move the elevator up one floor and ↓ to move the elevator down one floor. Assume that the building is arbitrarily tall and that the elevator starts at floor *x*. Consider the elevator grammar given below that generates arbitrary command sequences that (1) never cause the elevator to go below floor *x* and (2) always return the elevator to floor *x* at the end of the sequence.

    S → LS / ε

    L → ↑ L'

    L'→ L ↓ / ↓

    Here, ↑↑↓↓ and ↑↓↑↓ are valid command sequences, but ↑↓↓↑ and ↑↓↓ are not. For convenience, you may consider a null sequence as valid. Answer the following questions.

    a. Compute FIRST and FOLLOW Sets
    b. Check whether the given grammar is LL(1) by constructing LL(1) parsing table.
    c. Trace the LL(1) parser for the sentence ↑↓↑↑↓↓
16. Consider the Context-Free Grammar (CFG) depicted below where "begin", "end" and "x" are all terminal symbols of the grammar and Stat is considered the starting symbol for this grammar. Productions are numbered in parenthesis and you can abbreviate "begin" to "b" and "end" to "e" respectively.

    (1) Stat → Block
    (2) Block → begin Block end
    (3) Block → Body
    (4) Body → x

    Write the procedures of the non-terminals of a recursive descent parser.