|  |  |
| --- | --- |
| No. of Pages | 5 |
| No. of Questions | 9 |

**Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Department of Computer Science and Engineering**

**Final Examination FALL 2015**

**CSE420: Compiler Design**

**Total Marks: 70 Time Allowed: 2.45 Hour**

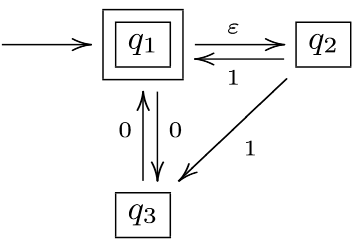
|  |
| --- |
| * You HAVE TO RETURN this question paper and the answer script at the end of the exam. Your script will not be checked unless you do so. * You are not allowed to communicate with any other candidate in any way what so ever. |

**Section 01 (There are 4 questions, answer any 3 out of them) [12 \*3 = 36]**

1. a. Draw the block diagram of **basic compilation phases**. [3]

b. Define the difference between **Lexical Analyzer** and **Syntax Analyzer**. [3]

c. Convert following NFA to DFA using **subset construction methodology**. [3]



d. Convert the regular expression mm(m|n)\*n over the alphabet Σ = {m,n} **directly** to **DFA**. [3]

2. a. What is meant by **left recursion** in a grammar? [1]

b. Consider the grammar with the set of terminals:

S → (L) | a | b

L → L,S | S

**Remove left-recursion** from the grammar and find the **First** and **Follow** sets for each non-terminal of the modified grammar. [1+1+2]

c. For the following grammar, construct the **LL(1) parse table**. [5]

X → X + X | Y++

Y → a

d. What do you understand by **shift-reduce** and **reduce-reduce conflicts**? [2]

3. Construct the **LR(1) parser table** for the following grammar.

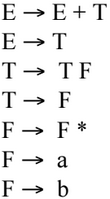
Show all the necessary steps. [12]

P → PaQ | Q

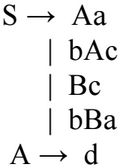
Q → QR | R

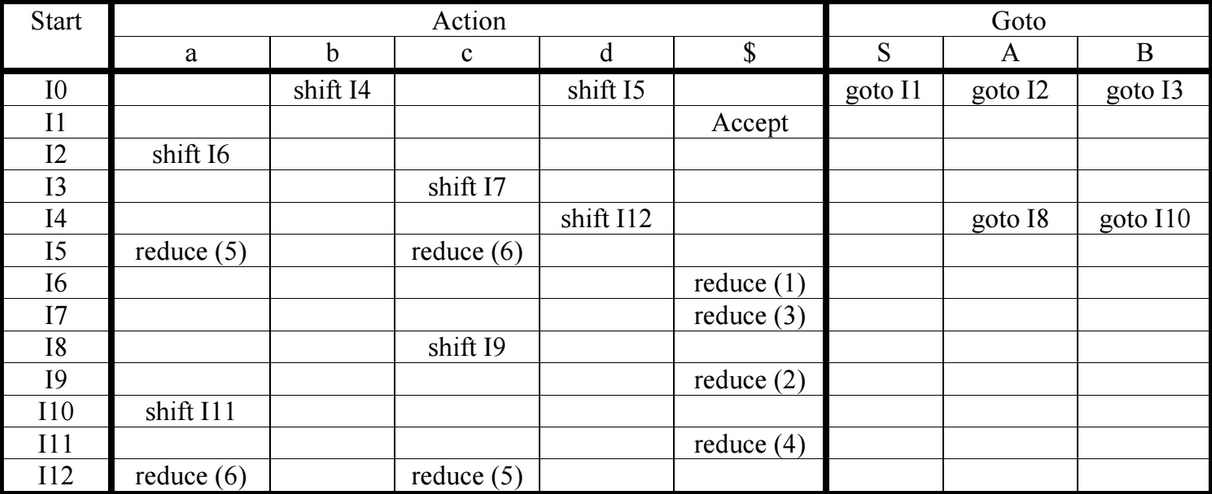
R → Rb | c | d

4. a. Construct the **LR (0) Automation** of following grammar. [6]



b. **Parse input string** “bda$” using following grammar and parsing table: [6]





**Section 02: (There are 3 questions, answer any 2 out of them) [10\*2 =20]**

1. a. Define **synthesized** and **inherited** attributes. [2]

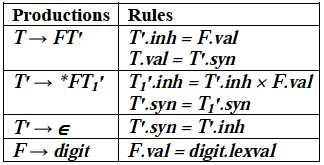
b. Translate the arithmetic expression a\*-(b+c) into: [3]

i) A syntax tree

ii) DAG

iii) Three-address code

c. Consider the following SDD: [3+2=5]



i) Draw the **annotated parse tree** for the expression 3\*5\*7 using the semantic rules given above.

ii) Draw the **dependency graph**.

2. a. Consider the following grammar: [3+4=7]

T → FT’

T’ → +FT’

T’ → ε

F → 1 | 2 | 3 | . . . | 9

i) Construct an **SDD** for the grammar.

ii) Using SDD constructed in (i) give an **annotated parse tree** for the expression: 2+3+4.

b. Discuss the comparative **advantages and disadvantages** of the following three representations: [3]

i) Quadruples

ii) Triples

iii) Indirect triples

3. a. What is **back-patching**? What is the advantage of back-patching? Explain with an example. [1+1+2]

b. Consider the following code fragment:

do i=1+1;

f=i\*5000; while (a[i] < v);

Write the **three address code** and its **quadruple** representation. [2+2]

c. Determine the equation to determine: [1+1]

i) i’th element of a **1-dimensional array**.

ii) (i, j)’th element of a **2-dimensional array**.

**Section 03: (There are 2 questions, answer any 1 out of them) [1\*14 = 14]**

1. a. Write short notes on the following: [2+2]

i) Basic blocks

ii) Peephole optimization

b. Draw the **flow graph** for the following program: [4]

begin

prod := 0;

i := 1;

do begin

prod := prod + a[i] \* b[i];

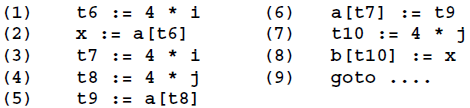
i := i + 1;

end

while (i <= 20);

end

c. For the following code fragment, determine the **next-use information**: [6]



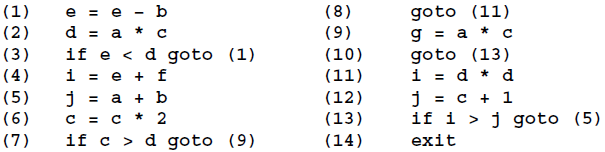
2. a. Consider the following fragment of intermediate code: [4]

|  |  |
| --- | --- |
| w = 2  u = z  y = w + 1  v = y \* y  r = v \*\* 2 //this is exponentiation | t = u \* u  s = u \* t  x = y \* y |

Assume the only variables live at the exit are s, x. Show the result of applying constant propagation, algebraic simplification, common sub-expression elimination, constant folding, copy propagation and dead code elimination as much as possible to this code. You should explain the changes in each step.

b. Draw a block diagram showing the caller-callee responsibilities in the construction of activation record. [3]

c. Consider the following sequence of 3-address codes: [2 + 5]



i) Draw the **flow graph**.

ii) Compute **live variables** at the end of each block using the iterative solution to dataflow equations for live variable analysis.