SPRING END SEMESTER EXAMINATION-2023

6th Semester B.Tech

COMPILER DESIGN

CS 3008

(For 2021 (L.E), 2020 & Previous Admitted Batches)

Time: 3 Hours Full Marks: 50

Answer any SIX questions.
Question paper consists of four SECTIONS i.e. A, B, C and D.
Section A is compulsory.
Attempt minimum one question each from Sections B, C, D.
The figures in the margin indicate full marks.
Candidates are required to give their answers in their own words as far as practicable and all parts of a question should be answered at one place only.

SECTION-A

1. Answer the following questions. [1 × 10]

(a) What are the positives of non-inclusion of lexical analyzers in parser?

(b) Find the number of tokens in the following C code.

```
main()
{
char c= 'B';
int a,b;
a=b=20;
a++;
printf("%d %d",a,b);
}
```

(c) Find the FIRST and FOLLOW for all the non terminals for the following given grammar,

S→ ACB | cbB | Ba

A→ da | BC

B→ g | ϵ

C→ h | ϵ

(d) Differentiate synthesized and inherited attribute in SDT.

(e) What is the maximum number of reduce moves that can be taken by a bottom-up parser for a grammar having no epsilon and unit production to parse a string with "n" tokens?

(f) Write the quadruples and triples for the following expression considering the width=4:

x[i]=y

(g) Define handle and how is it useful in bottom up parsing?

(h) Draw DAG for the expression
a+a+a+a+a+a+a+a

(i) Find out the output of the syntax directed translation for the given input "aaaabcc".

S → aaX { print(" 9")}
S → b { print(" 2")}
X → Sc { print(" 1")}

(j) How machine dependent code optimization is different from machine independent code optimization?

SECTION-B

2. (a) Left factor the following grammar: [4]

X-> xYw | xZC

Y -> x | xy | xyZy | xYy

Z -> zzw | zwZ | z | ϵ

Discuss the importance of Left Factoring and Removal of left recursion in top down parsing.

(b) Construct the LL(1) parsing table for the following grammar: [4]

$S \rightarrow xXZ \mid Yy$

$X \rightarrow qP$

$Y \rightarrow m \mid n$

$Z \rightarrow o \mid i$

$P \rightarrow yQ \mid \epsilon$

$Q \rightarrow qP \mid pP$

3. (a) Describe in detail the importance of symbol table in compiler design. Describe the various data structures to represent the symbol table. [4]

(b) What is peep-hole optimization? Discuss various peep-hole optimization techniques with example. [4]

SECTION-C

4. (a) Show that the following grammar [4]

$A \rightarrow BaBb \mid CbCa$

$B \rightarrow \epsilon$

$C \rightarrow \epsilon$

Is LR(1) but not SLR(1).

(b) Consider the following syntax directed translation (SDT), [4]

$S \rightarrow A$ { if A.sign = POS then print ("positive") else print ("negative")}

$A \rightarrow A_1 * A_2$ { If A_1 .sign = A_2 .sign then A.sign = POS A.sign = NEG}

$A \rightarrow + A_1$ {A.sign = A_1 .sign}

$A \rightarrow -A_1$ { if A_1 .sign = POS then

A.sign =NEG

else A.sign = POS}

$A \rightarrow$ integer { A.sign= POS}

(i) What is the type of the attribute "sign"?

(ii) What is the objective of the given SDT?

(iii) Evaluate the expression " 3 * -2 * 5 " using the given SDT.

5. (a) Consider the following grammar: [4]

$S \rightarrow aAd \mid bBd \mid aBe \mid bAe$

$A \rightarrow c$

$B \rightarrow d$

(i) Construct the canonical collection of LR(1) items

(ii) Construct the LR(1) parsing table

(iii) Simulate the working of the parsing table with a string derivable from the given grammar.

(b) Write the three-address code for the following code snippet: [4]

```
if (a < b + c)
    a = a - c;
c = b * c;
```

Represent the three-address code in the following forms:

i. Quadruples.

ii. Triples.

6. (a) Evaluate the value of the string 4+2*6+3 using the following SDT. Also provide the topological sorting representing the evaluation order. [4]

$E \rightarrow E+T$ { E.val = E.val + T.val }

$E \rightarrow T$ { E.val = T.val }

$T \rightarrow T*F$ { T.val = T.val * F.val }

$T \rightarrow F$ {T.val = F.val }

$F \rightarrow id$ { F.val = id.lexval }

(b) What are the different types of conflicts in bottom-up parsing? Explain each with example. [4]

Let m1, m2 and m3 be the number of states in SLR, CLR, LALR parser for the grammar $A \rightarrow (A) \mid a$ respectively.

Find the relationship among m1, m2 and m3.

SECTION-D

7. (a) Write the three-address code for the following code snippet: [4]

```
for (step = 1; step < size; step++) {
  key = array[step];
  j = step - 1;
  while (key < array[j] && j >= 0) {
    array[j + 1] = array[j];
    --j;
  }
  array[j + 1] = key;
}
```

(b) Define the principal uses of registers in code generation. [4]

What is register descriptor and address descriptor?

Consider the following basic block:

t = a – b

u = a – c

v = t + u

a = d

d = v + u

Show the contents of register and address using the *getreg()*.

8. (a) What is syntax directed translation? Write the semantic rules for the following given grammar (Where symbol **T** is associated with a synthesized attribute ***"type"*** *and* Symbol **L** is associated with an inherited attribute ***"in"***). [4]

$D \rightarrow T\ L$

$T \rightarrow int$

$T \rightarrow real$

$L \rightarrow L_1\ id$

$L \rightarrow id$

Give an attribute dependency graph for the statement **"int a"**

(b) Consider the code sequence, [4]

```
x= 20;
while (x<10)
{
  x=x-1;
```

```
    if(x>4)
    {
       x=x-2
    }
}
y=x+5;
```

(i) Write the three-address code.

(ii) Construct the flow graph by considering the basic blocks.

(iii) Optimize the code by applying all appropriate optimization techniques.

*****