SL No of QP 9946

Unique Paper Code:	2343010025
Name of the Paper:	Compiler Design
Name of Course:	B.Sc. (H) Computer Science (DSE)
Semester:	VII
Duration of Examination:	3 Hours
Maximum Marks:	90 Marks

Instructions for Candidates

1. Write your Roll No. on the top immediately on receipt of this question paper.
2. Section A is compulsory.
3. Attempt any four questions from Section B.
4. All parts of a question must be answered together.
5. Make and state necessary assumptions wherever applicable.

SECTION A

1 (a) Describe how token, pattern and lexeme differ and illustrate the distinction with the help of suitable example. **4**

(b) Determine the cost of the following instruction sequences: **4**

i.
```
LD R0, a
LD R1, b
MUL R0, R0, R1
ST c, R0
```

ii.
```
LD R0, x
LD R1, y
ST 0(R0), R1
```

(c) Consider the following grammar: **4**

```
S → A | B
A → aA | a
B → aBb |b
```

Is the above grammar ambiguous? Justify your answer.

(d) Draw the transition diagram for the following patterns: **4**

i. Identifier in C++
ii. Whitespace

(e) Explain the peephole optimization techniques using suitable example. **4**

(f) Consider the following context-free grammar: 4

```
S → ACB | CbB | Ba
A → da | BC
B → g | ∈
C → h | ∈
```

Compute the FOLLOW set for each non-terminal in the set.

(g) Suppose that we have a production `S → ABC`. Each of the four nonterminals `S`, `A`, `B` and `C` have two attributes: `s` is a synthesized attribute, and `i` is an inherited attribute. For each of the sets of rules below, tell whether the rule is S-attributed or L-attributed. Justify your answer. 4

i. `S:s = B:i + C:s`

ii. `A:s = B:s + C:s and B:i = A:i + C:i`

(h) Differentiate between static and dynamic storage allocation. 2

SECTION B

2 (a) Consider the following grammar: 9

```
E → Ea | Ec | c
```

Compute the canonical set of LR (0) items for the above grammar. Also, build the SLR parsing table for the grammar. Using the parsing table check whether the grammar is SLR or not? Justify your answer.

(b) Compare and contrast lexical, syntactic, semantic and logical errors with suitable examples. 6

3 (a) List the various phases of compiler with suitable example. 9

(b) Describe the language denoted by the following regular expressions: 6

i. 0(0|1)*0

ii. a*ba*ba*ba*

iii. (00|11)*((01|10)(00|11)*(01|10)(00|11)*)*

4 (a) Consider the context-free grammar: 9

```
S->AB | ABad
A->d
E ->b
D->b | ε
B->c
```

Construct the predictive parsing table. Check whether the given grammar is LL(1) or not. Justify your answer.

(b) Consider the following code snippet of bubble sort: **6**

```
void bubbleSort(int arr[ ],int n) {
if (n == 1)
 return;
for (int i=0; i<n-1; i++)
 if(arr[i]>arr[i+1])
   swap(arr[i],arr[i+1]);
bubbleSort(arr,n-1);
 }

void printArray(int arr [ ],int n) {
// print the sorted array
}

main ()
{
int arr[ ] = {62,32,23,10,20};
int n = sizeof(arr)/sizeof(arr[0]);
bubbleSort(arr,n);
printArray(arr,n);
}
```

Draw the activation tree to sort the array.

5 (a) Differentiate between synthesized and inherited attribute with the help of suitable example. **4**

(b) Translate the following code statement into three-address code: **5**

$$if(x < 100 \,||\, x > 200 \,\&\&\, x! = y)\; x = 0;$$

(c) Construct the annotated parse tree for the string **8 + 3 * 5** using the following Syntax Directed Definition (SDD): **6**

$L \rightarrow E$	$\{L.val = E.val\ \}$
$E \rightarrow E1 + T$	$\{E.val = E1.val + T.val\}$
$E \rightarrow T$	$\{E.val = T.val\}$
$T \rightarrow T1 * F$	$\{T.val = T1.\ val \times F.val\}$
$T \rightarrow F$	$\{T.val = F.val\ \}$
$F \rightarrow (E)$	$\{\ F.val\ = E.val\ \}$
$F \rightarrow digit$	$\{F.val = digit.lexval\ \}$

6 (a) Generate code for the following three-address statements assuming a and b are arrays whose elements are 8-byte values: **6**

i.
```
x = a[i]
y = b[j]
a[i] = y
b[j] = x
```

ii.
```
a = b * c
```

```
d = a + b
```

(b) Consider the following SLR parsing table below: **6**

	Action					GOTO		
State	a	b	0	1	$	S	A	B
0	S1		S2	S3		11	4	5
1	S1		S2	S3			6	7
2		r4	r4		r4			
3	r6	r6	r6	r6	r6			
4	r1	r1	r1		r1			
5	r2	r2	r2	r2	r2			
6		S8						
7		S9						
8		r3						
9		S10						
10		r5			r5			
11					acc			

Check whether the input string `a1bb` can be accepted or not for the following grammar.

```
S->A
S->B
A->aAb
A->0
B->aBbb
B->1
```

Show the stack contents, input buffer and action at each step.

(c) Define input buffering in compiler design. Why is it necessary during lexical analysis? **3**

7 (a) Construct the syntax tree and DAG for following expression: **4**

```
a+b*c+d+b*c-b
```

(b) Define left recursion. Consider the following grammar: **5**

```
A → Bb | a
B → Ad | c
```

Eliminate left recursion in the above grammar.

(c) Consider the following expression: **6**

$-a + (b * c) / (d - e)$

Write quadruples, triples and indirect triples for the given expression.