Noakhali Science and Technology University

Department of Computer Science and Telecommunication Engineering

3rd Year 1st Term B.Sc. (Engg.) Final Examination-2024

Course Code: CSTE 3111 **Course Title: Compiler Construction**

Time: 4 hours. **Total Marks: 70**

[Answer any seven of the following questions. Figures in the right hand margin indicate full marks]

1. a) What do you mean by compiler design architecture? "Multi-pass compiler can solve two basic problems."-what are they? Explain with examples. 1+3=4

   b) What are the roles of symbol table and error handler in compiler design? Explain. 3

   c) Differentiate between compiler and interpreter. 3

2. a) What is language processing system? Briefly describe the phases of a language processing system. 5

   b) Suppose a source program contains the following assignment statement: 2

   position = initial + rate * 60

   Now draw the syntax tree that shows the translation of above statement.

   c) Why is it necessary to divide compilation process in to various phases? 3

3. a) Draw an NFA for the regular expression **(a|b)*abb** using Ɛ-transition and construct an equivalent DFA for the expression. 5

   b) Consider the following parsing table: 5

Nonter-minal	Input Symbol					
	Id	+	*	(	)	$
E	E->TE'			E->TE'		
E'		E'->+TE'			E'->ε	E'->ε
T	T->FT'			T->FT'		
T'		T'->ε	T'->*FT'		T'->ε	T'->ε
F	F->id			F->(E)		

   Now show the moves made by predictive parser on input *id*(id + id)*.

4. a) Show the comparisons among token, pattern and lexeme with examples. 3

   b) What are the different kinds of error and describe error recovery techniques? 4

   c) Consider the following program of Quicksort: 3

```
main() {
        int n;
        readarray();
        quicksort(1,n);
}

quicksort(int m, int n) {
      int i= partition(m,n);
      quicksort(m,i-1);
      quicksort(i+1,n);
}
```

   Generate an activation tree for the given program.

5. a) List the rules for computing FOLLOW for a grammar. Compute FIRST and FOLLOW for the following grammar: 2+3=5

$S \rightarrow (L) \mid a$
$L \rightarrow L, S \mid S$

b) "A grammar G is $LL(1)$ iff whenever $A \rightarrow \alpha \mid \beta$ are two distinct productions of G, three conditions must hold."-what are they? 3

c) Write down the conditions for a grammar to be an operator grammar. 2

6. a) What is the rule for eliminating left factoring? Eliminate left recursion from the following grammar: 1+3=4

$S \rightarrow Aa \mid b$
$A \rightarrow Ac \mid Sd \mid \varepsilon$

b) Prove the following grammar is not SLR(1): 6

$S \rightarrow Aa \mid bAc \mid dc \mid bda$
$A \rightarrow d$

7. a) Show an annotate parse tree for the input expression 110.101 according to the following syntax-directed definition that converts binary to decimal with fraction: 4

Production	Semantic Rule
$S \rightarrow L1.L2$	$\{S.dv = L1.dv + \frac{L_2.dv}{2^{L_2.nb}}\}$
$L \rightarrow L.B$	$\{L.dv = 2*L.dv + B.dv$ $L.nb = L.nb + B.nb\}$
$L \rightarrow B$	$\{L.dv = B.dv$ $L.nb = B.nb\}$
$B \rightarrow 0$	$\{B.dv = 0$ $B.nb = 1\}$
$B \rightarrow 1$	$\{B.dv = 1$ $B.nb = 1\}$

b) What do attribute values typically represent? Differentiate between synthesized and inherited attributes with examples. 1.5+1.5= 3

c. "Any topological sort of a dependency graph gives a valid evaluation order of the semantic rules."-How? Explain with an example. 1+2=3

8. a) Define DAG. Draw DAG for the following statements: 1+3=4

1. $a = b + c$
2. $t1 = a * a$
3. $b = t1 + a$
4. $c = t1 * b$
5. $t2 = c + b$
6. $a = t2 + t2$

b) Evaluate three address code for the following code using backpatching: 4

```
prod = 0;
i = 1;
do

{
prod = prod + a[i] * b[i];
i = i + 1;
} while (i <= 10);
```

c) Show the comparisons among access link and control link of an activation record. 2

Follow(A)

9. a) What are the types of the machine independent code optimization techniques? Illustrate how the peephole optimization performed in the code generation phase of a compiler. 1.5+2.5= 4

b) What the purpose of program flow graph (PFG)? Consider the following three address code and then design PFG: 1+2=3

```
1. if (A < C) goto 3
2. goto 15
3. if (B > D) goto 5
4. goto 15
5. if (A = 1) goto 7
6. goto 10
7. T1 = c + 1
8. c = T1
9. goto 1
10. if (A <= D) goto 12
11. goto 1
12. T2 = A + B
13. A = T2
14. goto 10
15.
```

c) How can you allocate registers for the following code without spilling? 3

```
a = 1
b = 10
c = 20
d = a + b
e = c + d
f = c + e
b = c + e
e = b + f
d = 5 + e
Return (d + f)
```