2068 (I)

Attempt all questions. **(10x6=60)**

1. What do you mean by compiler? How source program analyzed? Explain in brief.
2. Discuss the role of symbol table in compiler design.
3. Convert the regular expression **'0 + (1 + 0)* 00'** first into NFA and then into DFA using Thomson's and Subset Construction methods.
4. Consider the following grammar:

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

   (a) Eliminate left recursion.
   (b) Computer FIRST & FOLLOW for the symbol in the grammar.

5. Consider the grammar

   $$C \rightarrow AB$$
   $$A \rightarrow a$$
   $$B \rightarrow b$$

   Calculate the canonical **LR(0)** items.

6. Describe the inherited and synthesized attributes of grammar using an example.
7. Write the type expressions for the following types.
   (a) An array of pointers to real's, where the array index range from 1 to 100.
   (b) Function whose domains are function from two characters and whose range is a pointer of integer.
8. What do you mean by intermediate code? Explain the role of intermediate code in compiler design.
9. What is operation of simple code generator? Explain.
10. Why optimization is often required in the code generated by simple code generator? Explain the unreachable code optimization.

2068 (II)

Attempt all questions. **(10x6=60)**

1. Explain the phase of a compiler with block diagram. (6)
2. Define token, pattern and lexeme with suitable example. How input buffering can be implemented for scanner, explain. (6)
3. Give the regular expression (0+1)*011, construct a DFA equivalent to this regular expression computing follow pos (). (6)
4. Explain the role of the parser. Write an algorithm for non-recursive predictive pursing. (6)
5. Construct the grammar

   E → E+T|T

   T → T*F|F

   F→ (E) |id

   Compute the complete LR(0) collection of item set from above grammar. (6)
6. Show that the following grammar is not in a LL(1) grammar. (6)

   S → cAd , A → Ab/a
7. What do you mean by Kernel and non-kernel items? Compute the Kernel items for LR(0) for the following grammar. (6)

   S → CC

   C →bC/d
8. What do you mean by S-attributed definition and how they are evaluated? Explain with example. (6)
9. What do you mean by three-code representation? Explain with example.(6)
10. How next-use information is useful in code-generation? Explain the steps involved on computing next-use information. (6)

2069

Attempt all questions. **(10x6=60)**

1.) What do you mean by compiler? Explain the semantic analysis phase of compiler construction.

2.) Why are regular expressions used in token specification? Write the regular expression to specify the identifier like in C.

3.) Discuss the specification of lexical analyzer generator Lex.

4.) Consider the grammar :

S……Sbs | bsas | ε

a.)Show that this grammar is ambiguous by constructing two different leftmost derivations for sentence abab.

b.) Construct the corresponding rightmost derivations for abab.

c.)Construct the corresponding parse trees for abab.

5.) Consider the grammar :

E→E+T | T
T→T*F|F
F→(E) | id

a.) Show steps of shift-reduce parsing for the input string id+id*id.
b.) Identify conflicts during the parsing

6.) Describe the L-attributed definitions. How L-attributed definitions are evaluated ?

7.) Write the type expressions for the following types :
a.) An array of pointers to reals where the array index ranges from 1 to 100.
b.) Funtion whose domains are functions from two characters and whose range is a pointer of integer.

8.) What do you mean by three address code? Write the syntax diected definition for following grammar to produce the three address codes for assignments

>if = E
>id

9.) Discuss the issues in design of simple code generator.

10.) Define the following optimization techniques :
a.) Unreachable code elimination
b.) Flow-of-control optimization

2071

Attempt all questions. **(10x6=60)**

1.) Explain the various phases of compiler in detail with practical example.

2.) Explain about design of lexical analyzer generator with its suitable diagram.

3.) What are the problem with top down parsers ? Explain the LR parsing algorithm.

4.) Define finite automata. Construct a finite automata that will accept a string at zeros and ones that contains an odd number of zeros and an even number of ones.

5.) What are the different issues in the design of code generator ? Explain with example about the optimization of basic blocks.

6.) What are the main issues involved in designing lexical analyzer ? Mention the various error recovery strategies for a lexical analyzer.

7.) Define a context free grammar. What are the component of context free grammar? Explain.

8.) What are the various issues of code generator ? Explain the benefits of intermediate code generation.

9.) Explain the peephole organization. Write a three address code for the expression r; = 7*3+9.

10.) Differentiate between Pascal compiler and C++ compiler.

Tribhuvan University
Institute of Science and Technology
Bachelor of Computer Science and Information Technology

Course Title: Compiler Design and Construction
2071 (II)

Course No.: CSC-352 **Time: 3 hours**

Full Marks: 60
Pass Marks: 24

Attempt all questions. **[10x6=60]**

1.) Define the compiler. Explain the phases of compiler.
2.) Design a lexical analyzer generator and explain it.
3.) Differentiate between top-down parsing and bottom-up parsing.
4.) Translate the arithmetic expression a*-(b+c) into syntax tree. Explain the ambiguous grammar.
5.) Explain the dynamic programming code generation algorithm with example.
6.) What do you mean by code optimization ? Explain the basic blocks and their optimization.
7.) What are the generic issues in the design of code generators? Explain.
8.) What are the compiler construction tools? Explain.
9.) Explain the principle sources of code optimization with example.
10.) Differentiate between C compiler and Pascal compiler.