UNIVERSITY OF MINES AND TECHNOLOGY, TARKWA

SECOND SEMESTER EXAMINATIONS, MAY 2018

COURSE NO: CE 470
COURSE NAME: **COMPILER CONSTRUCTION**
CLASS: CE IV TIME: 3hrs

Name**:** _____Index Number: _____

Answer all the questions in Section A and any other two in section B. Carefully circle your answers in section A

1. Compiler translates the source code to
   a. Executable code
   b. Machine code
   c. Binary code
   d. Both B and C

2. Which of the following groups is/are token together into semantic structures?
   a. Syntax analyzer
   b. Intermediate code generation
   c. Lexical analyzer
   d. Semantic analyzer

3. Compiler should report the presence of _____ in the source program, in translation process
   a. Classes
   b. Objects
   c. Errors
   d. Text

4. What is the output of lexical analyzer?
   a. A parse tree
   b. A list of tokens
   c. Intermediate code
   d. Machine code

5. How many parts of compiler do we have?
   a. 1
   b. 2
   c. 4
   d. 8

6. Grammar of the programming is checked at _____ phase of compiler
   a. Semantic analysis
   b. Syntax analysis
   c. Code optimization
   d. Code generation

7. _____ is a process of finding a parse tree for a string of tokens
   a. Parsing
   b. Analysing

c. Recognizing
d. Tokenizing

8. What is the action of parsing the source program into proper syntactic classes?

a. Lexical analysis
b. Syntax analysis
c. General syntax analysis
d. Interpretation analysis

9. Compiler can check _____ error

a. Logical
b. Syntax
c. Content
d. Both A and B

10. A grammar that produces more than one parse tree for some sentence is called a

a. Ambiguous
b. Unambiguous
c. Regular
d. All of the above

11. Lexical analysis is about breaking a sequence of characters into

a. Groups
b. Packets
c. Lines
d. Tokens

.

12. _____ is the most general phase structured grammar

a. Context sensitive
b. Regular
c. Context free
d. All of these

.

13. _____ is considered as a sequence of characters in a token

a. Texeme
b. Pattern
c. Lexeme
d. Mexeme

14. What is the name of the process that determines whether a string of tokens can be generated by a grammar?

a. Analysing
b. Recognizing
c. Translating
d. Parsing

15. Three address code involves

a. Exactly 3 address
b. At most 3 address
c. No unary operators
d. None of the above

16. An intermediate code form is

a. Postfix Notation
b. Syntax Free
c. Three address code
d. All of the above

17. In operator precedence parsing , precedence relations are defined

a. For all pair of non terminals
b. For all pair of terminals
c. To delimit the handle
d. Only for certain pair of terminals

18. Synthesized attribute can be easily simulated by

a. LL Grammar
b. Ambiguous Grammar
c. LR Grammar
d. Non of the above

19. The output of a lexical analyzer is

a. Machine Code
b. Intermediate Code
c. A stream of Tokens
d. A Parse Tree

20. Task of the lexical analysis

a. To parse the source program into the basic elements or tokens of the language
b. To build a literal or identifier table
c. To build a uniform symbol table
d. All of the above

21. Shift reduce parsers are

a. Top down Parser
b. Bottom Up Parser
c. Maybe top down or bottom up Parser
d. None of the above

22. The Linker

a. Is similar to interpreter
b. Uses source code as its input

c. Is required to create a load module
d. None of the above

23. A grammar that produces more than one parse tree for some sentence is called

a. Ambiguous
b. Unambiguous
c. Regular
d. None of the above

24. Intermediate code generation phase gets input from

a. Lexical Analyzer
b. Syntax Analyzer
c. Semantic Analyzer
d. Error Handling

25. We can optimize code by

a. Deadcode elimination
b. Coomon subprogam
c. Copy immediate loop
d. All of the above

26. A top down parser generates

a. Rightmost Derivation
b. Rightmost Derivation in Reverse
c. Leftmost Derivation
d. Leftmost Derivation in Reverse

27. _____is a graph representation of a derivation

a. The Parse Tree
b. The Oct Tree
c. The Binary Tree
d. None of the above

28. LR stands for

a. Left to Right
b. Left to Right Reduction
c. Right to Left
d. Left to Right and right most derivation in reverse

29. Grammar of the programming is checked at _____ phase of compiler

a. Semantic Analysis
b. Code Generation
c. Syntax Analysis
d. Code Optimization

30. Which of the following is used for grouping of characters into tokens?

a. Parser
b. Code Optimization
c. Code Generation
d. Lexical Analy

SECTION B

1a. Draw an NFA for the following regular expressions:

i. a (b |c) d

ii. (abc)*

iii. ((a b)* c (d | e) (f | g))*

b. Write context free grammars for the following languages (your grammar does not have to be LR(1), LL(1) etc):

i. All strings open and close parentheses, where the parentheses are balanced.

ii. The language described by the regular expression ((ab)*(c|d))*

iii. Expressions consisting of num, +, and *. You should write your grammar so that * has higher precedence than +

c. Differentiate between a compiler and an interpreter

2a. What qualities would you want in a compiler? Discuss its common uses.

b. Draw a Pass tree for b*b-4*a*c, Hence, Draw a Parse Tree for -10/5*8-4-5

c. Construct the NFA of a (b/c)*. Hence, reduce the NFA to DFA

3a. Consider the following grammar:

```
S -> S a S b
   | c
   | Q q
Q -> Q m
   |
```

i. Which non-terminals (if any) can derive empty?

ii. What are the FIRST sets of Q and S?

iii. What are the FOLLOW sets of Q and S?

b. What is the language *L*(G) of the grammar G = (S,P,t,nt) below?

•bin → bin '+' dig

•bin → bin '–' dig

•bin → dig

•dig → '0'

•dig → '1'

c. The lexical- (scanner), syntactic- (parser), and semantic-analysis phases of a compiler front-

end each process parts of the source program in particular ways and also check certain rules of the language being compiled. For each of the following possible language rules, specify which phase of the compiler should verify that a program conforms to that rule and why that part of the compiler is the best place for that check. If a check could be done equally well in more than one phase of the compiler, briefly discuss the tradeoffs between the alternative implementations. Keep your answers short and to the point.

i. A function is called with the correct number of arguments.

ii. Underscore characters (_) may appear in the middle of identifiers, but not at the beginning or end (i.e., this_identifieris legal, but this_oneis not

iii. Every variable must be declared before it is used in the program (the classic C or Pascal rule).

iv. Assignment statements must end with a semicolon (;).

Examiners: Prof B. K. Alese