Compiler Design
Chapter 1: Compiler Intro

GATE CS PYQ

by Monalisa

- GATE CS 2010,Q13: Which data structure in a compiler is used for managing information about variables and their attributes?
  (A) Abstract syntax tree
  (B) Symbol table
  (C) Semantic stack
  (D) Parse Table
- Symbol table is the abstract data structure use by compiler to store all the information about variable and their attribute used in the program.
- Ans: **(B)** Symbol table

- GATE CS 2011,Q1: In a compiler, keywords of a language are recognized during
- **(A)** parsing of the program
  (B) the code generation
  (C) the lexical analysis of the program
  (D) dataflow analysis
- Lexical analysis is the process of converting a sequence of characters into tokens.
- During scan it will recognize keywords and convert into tokens.
- Ans: **(C)** the lexical analysis of the program

- GATE CS 2011,Q19: The lexical analysis for a modern computer language such as Java needs the power of which one of the following machine models in a necessary and sufficient sense?
  (A) Finite state automata
  (B) Deterministic pushdown automata
  (C) Non-Deterministic pushdown automata
  (D) Turing Machine
- Lexical Analysis use regular expression as pattern for identifying tokens.
- So Lexical analysis for java need power of Finite state automata.
- **Ans :(A)** Finite state automata

- GATE CS 2015 Set-2,Q19: Match the following:

P. Lexical analysis	1.Graph coloring
Q. Parsing	2.DFA Minimization
R. Register allocation	3.Post-order traversal
S. Expression evaluation	4.Production Tree

- (A)P-2, Q-3, R-1, S-4
- (B)P-2, Q-1, R-4, S-3
- (C)P-2, Q-4, R-1, S-3
- (D)P-2, Q-3, R-4, S-1
- P. Lexical Analysis uses Regular expression.
- We can find RE from minimized DFA. [2]
- Q. The parser constructs a production tree. [4]
- R. Register allocation can be done by graph coloring.
- By graph coloring we can reduce number of register. [1]
- S. Expression can be evaluated with postfix traversals.[3]
- Ans : (C)P-2, Q-4, R-1, S-3

- GATE CS 2016 Set-2,Q19: Match the following:

(P) Lexical analysis	(i) Leftmost derivation
(Q) Top down parsing	(ii)Type Checking
(R) Semantic analysis	(iii) Regular expressions
(S) Runtime environment	(iv) Activation records

- (A) P ↔ i, Q ↔ ii, R ↔ iv, S ↔ iii
  (B) P ↔ iii, Q ↔ i, R ↔ ii, S ↔ iv
  (C) P ↔ii, Q ↔ iii, R ↔ i, S ↔ iv
  (D) P ↔ iv, Q ↔ i, R ↔ ii, S ↔ iii
- (P) Lexical analysis uses Regular expression to recognize tokens.[iii]
- (Q) Top down parsing uses Left Most Derivation to generate the string.[i]
- (R) In Semantic analysis phase we do Type checking.[ii]
- (S) Activation records of a function are loaded into stack memory at Run time.[iv]
- Ans : (B) P ↔ iii, Q ↔ i, R ↔ ii, S ↔ iv

- GATE CS 2017 Set-2,Q5: Match the following according to input (from the left column) to the compiler phase (in the right column) that processes it:

(P) Syntax Tree	(i) Code generator
(Q) Character stream	(ii) Syntax analyzer
(R) Intermediate representation	(iii)Semantic analyzer
(S) Token stream	(iv) Lexical analyzer

- **(A)** P → (ii), Q → (iii), R → (iv), S → (i)
  (B) P → (ii), Q → (i), R → (iii), S → (iv)
  (C) P → (iii), Q → (iv), R → (i), S → (ii)
  (D) P → (i), Q → (iv), R → (ii), S → (iii)
- (P) Syntax tree is the input for the semantic analyzer.[iii]
- (Q) Character stream is input to lexical analyzer .[iv]
- (R) Intermediate representation is input for Code generator. [i]
- (S) Token stream is input for Syntax analyzer .[ii]
- **Ans :(C)** P → (iii), Q → (iv), R → (i), S → (ii)

- GATE CS 2018,Q8: Which one of the following statements is FALSE?
  (A) Context-free grammar can be used to specify both lexical and syntax rules.
  (B) Type checking is done before parsing.
  (C) High-level language programs can be translated to different Intermediate Representations.
  (D) Arguments to a function can be passed using the program stack.
- (A)Lexical analysis uses Regular Expression .
- We can design CFG for RE.as RL⊂ CFL.
- Syntax analysis uses CFG for parse tree.Both lexical & syntax rule use CFG.[True]
- (B)Type checking is done in semantic analysis phase after parsing.[False]
- (C)We have various types of Intermediate Code Representations, ex 3-address code, Postfix notation. quadruples [True]
- (D) Program stack holds the activation record of the function called, which stores function parameters, return value, return address etc.[True]
- Ans : (B) Type checking is done before parsing.

- GATE CS 2018,Q37: A lexical analyzer uses the following patterns to recognize three tokens T_1, T_2, and T_3 over the alphabet $\{a,b,c\}$.
  T_1: $a?\ (b|c)^*a$
  T_2: $b?\ (a|c)^*b$
  T_3: $c?\ (b|a)^*c$
  Note that 'x?' means 0 or 1 occurrence of the symbol x. Note also that the analyzer outputs the token that matches the longest possible prefix.If the string $bbaacabc$ is processed by the analyzer, which one of the following is the sequence of tokens it outputs?
  (A) $T_1T_2T_3$ (B) $T_1T_1T_3$ (C) $T_2T_1T_3$ (D) T_3T_3
- T_1 : (b+c)*a + a(b+c)*a
- T_2 : (a+c)*b + b(a+c)*b
- T_3 : (b+a)*c + c(b+a)*c
- String= bbaacabc
- T_1 :bba , T_2 :bb , T_3 :bbaac
- Longest possible prefix is : bbaac generated by T_3.
- T_3 :abc
- String= bbaac abc= T_3T_3
- Ans: (D) T_3T_3

- GATE CS 2020,Q9: Consider the following statements.
- **I.** Symbol table is accessed only during lexical analysis and syntax analysis.
- **II.** Compilers for programming languages that support recursion necessarily need heap storage for memory allocation in the run-time environment.
- **III.** Errors violating the condition ‘any variable must be declared before its use’ are detected during syntax analysis.
- Which of the above statements is/are TRUE ?

  (A) I only **(B)** I and III only **(C)** II only **(D)** None of I, II and III
- I is wrong as Symbol table is also accessed during all the phases.
- II is wrong as compilers which supports recursion require stack memory in run time environment not heap.
- III is wrong “any variable must be declared before its use” is a semantic error and not syntax error.
- Ans (D) None of I,II,III

- GATE CS 2021,Set-2,Q3: Consider the following ANSI C code:
- int main(){
- Integer x;
- return 0;
- }
- Which one of the following phases in a seven-phase C compiler will throw an error?
- (A) Lexical analyzer
- (B) Syntax analyzer
- (C) Semantic analyzer
- (D) Machine dependent optimizer
- int x;
- Integer x; <Identifier 1><identifier 2>
- Fine for lexical analyzer & syntax analyzer.
- “any variable must be declared before its use” is a semantic error.
- Compiler will assume x is not declared .So it’s a semantic error.
- Ans: (C) Semantic analyzer

- **GATE CS 2023 | Question: 1**
- Consider the following statements regarding the front-end and back-end of a compiler.
- **S1:** The front-end includes phases that are independent of the target hardware.
  S2: The back-end includes phases that are specific to the target hardware.
  S3: The back-end includes phases that are specific to the programming language used in the source code.
- Identify the CORRECT option.
- (A)Only S1 is TRUE. (B)Only S1 and S2 are TRUE.
- (C)S1,S2, and S3 are all TRUE. (D)Only S1 and S3 are TRUE.
- **Front end :** analysis (machine independent)
- These include lexical and syntactic analysis, the creation of the symbol table ,semantic analysis and the generation of intermediate code.
- It convert source code into intermediate code.It also includes error handling that goes along with each of these phases.
- **Back end :** synthesis (machine dependent)
- It includes code optimization phase and code generation along with the necessary error handling and symbol table operations.It convert 3 address code into assembly language.
- S1:True ,S2:True ,S3: False
- Ans : (B)Only S1 and S2 are TRUE.

- **GATE CS 2024 | Set 2 | Question: 11**
- Consider the following two sets:

Set X	Set Y
P. Lexical Analyzer	1. Abstract Syntax Tree
Q. Syntax Analyzer	2. Token
R.Intermediate Code Generator	3. Parse Tree
S. Code Optimizer	4. Constant Folding

- Which one of the following options is the CORRECT match from ***Set X to Set Y***?
- (A)P−4;Q−1;R−3;S−2 (B)P−2;Q−3;R−1;S−4
- (C)P−2;Q−1;R−3;S−4 (D)P−4;Q−3;R−2;S−1
- **Ans :** (B)P−2;Q−3;R−1;S−4