The Terminology of CS-321

abstract syntax tree
ACTION / GOTO tables
Ada (a programming language)
alphabet (□)
ambiguous
assembler / assembly language
associative / associativity
AST
attributes
back end of compiler
basic types (primitive types)
BNF
boolean
bottom-up
CFG
checker / type-checking phase
code generation
commutative / commutativity
compiler / compilation
concatenation
constructed types / type constructors
context free grammar
cycles (in graphs)
DAG
declaration (vs. definition)
definition (vs. declaration)
derivation
deterministic finite state automaton
DFA
directed acyclic graph
dynamically typed language
empty set
empty string
epsilon
epsilon edges / □-edges
equivalence (or regular expressions)
expression / term / factor
final state (accept state) of an FSA
finite state machine / finite state automaton
FIRST set
front end of compiler
FOLLOW set
FSA (finite state automaton)

function type (DomainType □ RangeType)
grammar
graph / node / edge
handle
infix
inherited attributes
interior node
interpreter
item
Kleene closure
L □ M (“union of languages”)
L^*
L M (“concatenation of languages”)
L(G)
LALR
language (as set of strings)
leaf node
lefthand / righthand side (e.g., of CFG rule)
leftmost derivation
left-recursion / left-recursive rule
lexeme
lexer
lexical / syntactic
lexical analyzer
lexical level
LL
LL(1)
LL(k)
lookahead
LR
LR(0) item
LR(1)
LR(1) item
LR(k)
l-value
machine language / machine code
minimal DFA (minimum state DFA)
ML (a programming language used in examples)
NFA
non-deterministic finite state automaton
non-terminal
operand
operator
optimization (a phase of the compiler)
overloaded operator / function
parameter vs. argument

parse / parsing / parser
parse tree / derivation tree
Pascal / Fortran / C / C++ / Java
PCAT
phases (of compiler)
polymorphic types
positive closure
postfix notation
precedence (of operators)
predictive parsing
prefix (expression notation)
prefix (of a string)
production (in CFG)
recursive (routine / function)
recursive descent parsing
recursive transition diagrams
recursion
regular expression
regular language
regular set
return type (of a function)
rightmost derivation
right-recursion / right-recursive rule
rule (in CFG)
runtime vs. compile-time
r-value
S-attributed definitions
scope
semantics
set / intersection / union / member / subset
shift / reduce / accept / error
short-circuit operators
SLR
source language
stack
start state (of a finite state machine)
start symbol (of a CFG)
statement / loop / body / if-stmt / while-stmt / etc.
states (in DFA)
states (in LR parser)
static vs. dynamic
statically typed language
string (of symbols from an alphabet)
string table
strongly typed language
structural type equivalence vs. name equivalence

subset / proper subset / superset / proper superset
substitution
suffix
symbol table
syntax
syntax-directed definitions
synthesized attributes
target language
terminal
Thompson's Construction
token
top-down
transitions (edges in FSA)
transition diagrams
translation scheme
tree / node / parent / children
type checking
type coercion
type conversion
type expressions
type inference
type variables
unifier / most-general unifier
unify / unification
viable prefix
void / non-void (functions)
white space
YACC
□(empty string)
□-transitions
□-closure
□ (universal quantification)
□ (existential quantification)