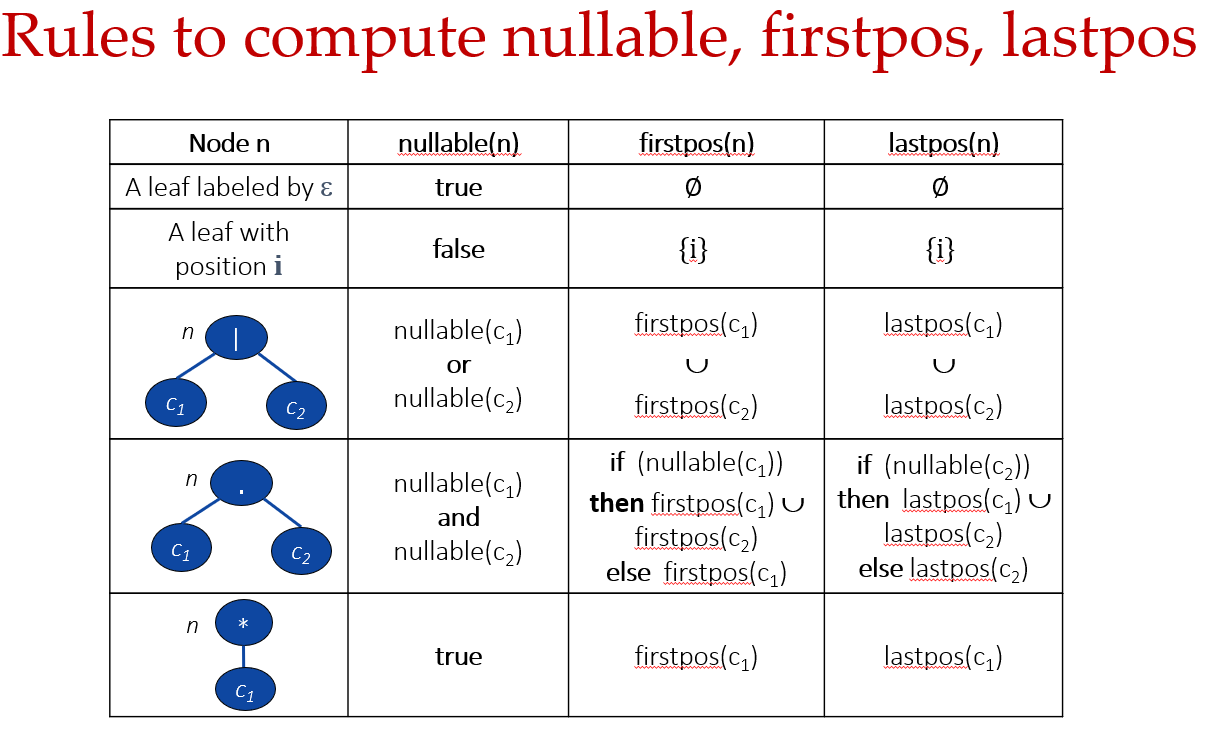
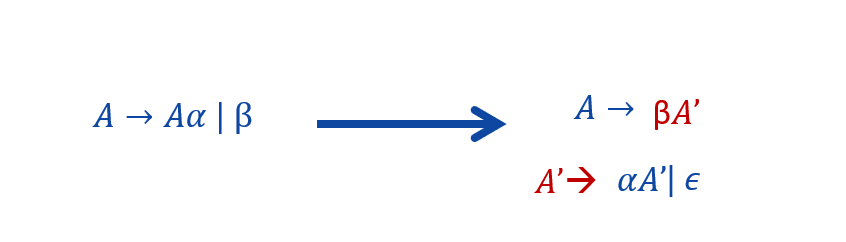
**Compiler Quiz Notes**

1. LLVM = low level virtual machine, compiler toolchains, make frontend of programming languages, backend for instruction set architectures – designed for compile time, link time, run time and idle time optimization, written in cpp, language independent IR
2. Languages->IR->machinecode
3. IR: Does not use fixed set of registers, multiple variables with ids, strongly types RISC, 3 forms – Human readable, in-memory format, dense bitcode format,
4. Translator: converts one form to another, Types: Compiler: translates into equivalent program of target language, Interpreter: same as compiler but line by line, Assembler: for assembly code
5. Code – Analysis Phase – Intermediate representation – Synthesis Phase – Target Code
6. Analysis phase: Breaks code into pieces to convert into IR, Lexical(division into tokens called lexemes with ids), Syntax (Parsing into parse tree, checks for errors in the tokens),Semantic Analysis(checks parenthesis, if else statement, type compatible arithematic operations, scope verify)
7. Intermediate code generation: easy to produce, easy to translate, three address code recommended
8. Synthesis phase: constructs output from intermediate representation, code optimization(enhanced the IR to make it faster for generation) and generation
9. Symbol Table: stores name of tokens and entities along with their addresses, is a linked list or a hash table
10. Compiler phases: preprocessing(decodes macros and headers) – Compiler(translation) - Assembler(translated code to machine level code) – Linker(single program from several files, libraries etc) – Loader(code with relocatable address to machine memory address)
11. Input techniques: Buffer pairs(data divided into n character levels), Sentinel(have special characters to mark end of a part of string)
12. First(A): in the production of A, the first terminal
13. Follow(A): locations in the language where A is occurring in rhs, the next terminals after A, follow does not include epsilon
14. 
15. Left Recursion: Top Down parsers cannot handle left recursive
16. Left Factoring: multiple productions have similar starting terminal symbol, new productions can be added to remove confusion