**Babatunde Tobi Samuel**
**16/SCI01/064**
**CSC312 ASSIGNMENT**
**400L**

## Grammar

A grammar lets us transform a program, which is normally represented as a linear sequence of ASCII characters, into a syntax tree. Only programs that are syntactically valid can be transformed in this way. This tree will be the main data-structure that a compiler or interpreter uses to process the program.

## Derivation

In computer science, program derivation is the derivation of a program from its specification, by mathematical means.

To *derive* a program means to write a formal specification, which is usually non-executable, and then apply mathematically correct rules in order to obtain an executable program satisfying that specification. The program thus obtained is then correct by construction. Program and correctness proof are constructed together.

The approach usually taken in formal verification is to first write a program, and then provide a proof that it conforms to a given specification. The main problems with this are that

- the resulting proof is often long and cumbersome;
- no insight is given as to how the program was developed; it appears "like a rabbit out of a hat";
- Should the program happen to be incorrect in some subtle way, the attempt to verify it is likely to be long and certain to be fruitless

Program derivation tries to remedy these shortcomings by

- keeping proofs shorter, by development of appropriate mathematical notations;  Making design decisions through formal manipulation of the specification.

Terms that are roughly synonymous with program derivation are: transformational programming, algorithmic, deductive programming.

# Production

A production system (or production rule system) is a computer program typically used to provide some form of artificial intelligence, which consists primarily of a set of rules about behavior but it also includes the mechanism necessary to follow those rules as the system responds to states of the world. Those rules, termed productions, are a basic representation found useful in automated planning, expert systems and action selection.

Productions consist of two parts: a sensory precondition (or "IF" statement) and an action (or "THEN"). If a production's precondition matches the current state of the world, then the production is said to be *triggered*. If a production's action is executed, it is said to have *fired*. A

production system also contains a database, sometimes called working memory, which maintains data about current state or knowledge, and a rule interpreter. The rule interpreter must provide a mechanism for prioritizing productions when more than one is triggered.

# Null Symbol

A null character refers to any character that has a numeric value of zero. It is termed a null character as it doesn't carry a value and all its bit are set on 0. A null character is also known as a null terminator.

Although a null character is valueless, it has wide implementation in programming languages and applications. A null character is a type of control character and is part of most character sets. Typically, a null character is represented by a "space" or empty data set in applications such as a word processing database and is used for filling empty spaces and padding. In programming languages/context, a null character is represented by the escape sequence \0, and it marks the end of a character string.

# Sentence

In computer programming, a statement is a syntactic unit of an imperative programming language that expresses some action to be carried out.[1] A program written in such a language is formed by a sequence of one or more statements. A statement may have internal components (e.g., expressions).

Many imperative languages (e.g. C) make a distinction between statements and definitions, with a statement only containing executable code and a definition instantiating an identifier, while an expression evaluates to a value only. A distinction can also be made between simple and compound statements; the latter may contain statements as components.