Example: $A = \{0, 1\}$; start symbol <s>; 2 production rules given by:

1. <s> $\to$ 0<s>1
2. <s> $\to$ 01

Let's see what we generate: via rule 2, <s> $\to$ 01, so we get <s> $\Rightarrow$ 01
Via rule 1, <s> $\to$ 0<s>1, then via rule 2, 0<s>1 $\to$ 0011. We write the process as <s> $\Rightarrow$ 0<s>1 $\Rightarrow$ 0011.
Via rule 1, <s> $\to$ 0<s>1, then via rule 1 again 0<s>1 $\to$ 00<s>11, then via rule 2, 00<s>11 $\to$ 000111.
We got <s> $\Rightarrow$ 0<s>1 $\Rightarrow$ 00<s>11 $\Rightarrow$ 000111.
The language L we generated thus consists of all strings of the form $0^m 1^m$ (m 0's followed by m 1's) for all $m \geq 1, m \in \mathbb{N}$
We saw 2 types of strings that appeared in this process of generating L:

1. terminals, **i.e.** the elements of A
2. nonterminals, **i.e.** strings that don't consist solely of 0's and 1's such as <s>, 0<s>1, 00<s>11, etc.

The production rules then have the form:

nonterminal $\to$ word over the alphabet V = {terminals, non-terminals}

<T> $\to$ w

In our notation, the set of nonterminals is $V \backslash A$, so <T>$\in V \backslash A$ and $w \in V^* = \bigcup_{n=0}^{\infty} V^n$. To the production rule <T>$\to w$, we can associate the ordered pair (<T>$, w) \in (V \backslash A) \times V^*$, so the set of production rules, which we will denote by P, is a subset of the Cartesian product $(V \backslash A) \times V^*$.
Grammars come in two flavours:

1. Context-free grammars where we can replace any occurrence of <T> by w if <T>$\to w$ is one of our production rules.
2. Context-sensitive grammars only certain replacements of <T> by w are allowed, which are governed by the syntax of our language L.

The example we had was of a context-free grammar. We can now finally define context free-grammars.

Definition: A context-free grammar ($V, A,$ <s>$, P$) consists of a finite set V, a subset A of V, an element <s> of $V \backslash A$, and a finite subset P of the Cartesian product $V \backslash A \times V^*$.

Notation: $(\underset{\text{set of terminals and non terminals}}{V}, \underset{\text{set of terminals}}{A}, \underset{\text{start symbol}}{\langle s \rangle}, \underset{\text{set of production rules}}{P})$

Example: $A = \{0, 1\}$; start symbol <s>; 3 production rules given by:

1. <s>$\to$ 0<s>1
2. <s>$\to$ 01
3. <s>$\to$ 0011

We notice here that the word 0011 can be generated in 2 ways in this context free grammar:

By rule 3, <s>$\to$ 0011 so <s>$\Rightarrow$ 0011

$\vee$

By rule 1, <s>$\to$ 0<s>1 and by rule 2, 0<s>1 $\to$ 0011. Therefore, <s>$\Rightarrow$ 0<s>1 $\Rightarrow$ 0011.

Definition: A grammar is called ambiguous if it generates the same string in more than one way.
Obviously, we prefer to have unambiguous grammars, else we waste computer operations.
Next, we need to spell out how words relate to each other in the production of our language via the grammar:

Definition: Let w' and w” be words over the alphabet $V = \{$terminals, nonterminals$\}$. We say that w' directly yields w” if $\exists$ words u and v over the alphabet V and a production rule <T>$\to$ w of the grammar s.t. $w' = u$ <T> v and w” $= uwv$, where either or both of the words u and v may be the empty word.
In other words, w' directly yields w” $\Leftrightarrow \exists$ production rule <T>$\to$ w in the grammar s.t. w” may be obtained from w' by replacing a simple occurrence of the nonterminal <T> within the word w' by the word w.

Notation: w' directly yields w” is denoted by $w' \Rightarrow w$”

Definition: Let w' and w” be words over the alphabet V. We say that w' yields w” if either $w' = w$” or else $\exists$ words $w_0, w_1, ...w_n$ over the alphabet V s.t. $w_0 = w', w_n = w$”$, w_{i-1} \Rightarrow w_i$ for all $i, 1 \leq i \leq n$. In other words, $w_0 \Rightarrow w_1 \Rightarrow w_2 \Rightarrow ... \Rightarrow w_{n-1} \Rightarrow w_n$

Notation: w' yields w” is denotes by $w' \stackrel{*}{\Rightarrow} w$”.

Definition: Let $(V, A, \text{<s>}, P)$ be a context-free grammar. The language generated by this grammar is the subset L or A^* defined by $L = \{w \in A^* \mid \text{<s>} \stackrel{*}{\Rightarrow} w\}$
In other words, the language L generated by a context-free grammar $(V, A, \text{<s>}, P)$ consists of the set of all finite strings consisting entirely of terminals that may be obtained from the start symbol <s> by applying a finite sequence of production rules of the grammar, where the application of one production rule causes one and only one nonterminal to be replaced by the string in V^* corresponding of the right-hand side of the production rule.

8.1 Phrase Structure Grammars

Definition: A phrase structure grammar $(V, A$ <s>$, P)$ consists of a finite set V, a subset A of V, an element <s> of $V \backslash A$, and a finite subset P of $(V^* \backslash A^*) \times V^*$
In a context-free grammar, the set of production rules $P \subset (V \backslash A) \times V^*$.
In a phrase structure grammar, $P \subset (V^* \backslash A^*) \times V^*$. In other words, a production rule in a phrase structure grammar $r \to w$ has a left-hand side r that may contain more than one nonterminal. It is required to contain <u>at least one</u> nonterminal.
For example, if $A = \{0, 1\}$ and <s> is the start symbol in a phrase structure grammar grammar, 0<s>0<s>0 $\to$ 00010 would be an acceptable production rule in a phrase structure grammar but not in a context-free grammar.
The notions $w' \Rightarrow w"$ (w' directly yields $w"$) and $w' \overset{*}{\Rightarrow} w"$ (w' yields $w"$) are defined the same way as for context-free grammars except that our production rules may, of course, be more general as we saw in the example above.

Definition: Let $(V, A$ <s>$, P)$ be a phrase structure grammar. The language generated by this grammar is the subset L or A^* defined by $L = \{w \in A^* \mid$ <s >$\overset{*}{\Rightarrow} w\}$

Remark: The term phrase structure grammars was introduced by Noam Chowsky.

Definition: A language L generated by a context-free grammar is called a <u>context-free language</u>.

We now want to understand a particularly important subclass of context-free languages called regular languages.

8.2 Regular Languages

Task: Understand when a language is regular and how regular languages are produced. Understand basics of automata theory.

History: The term <u>regular language</u> was introduced by Stephen Kleene in 1951.
A more descriptive name is <u>finite-state language</u> as we will see that a language is regular $\Leftrightarrow$ it can be recognised by a finite state acceptor, which is a type of finite state machine.
The definition of a regular language is very abstract, though. First, describe what operations the collection of regular languages is closed under:

Let A be a finite set, and let A^* be the set of all words over the alphabet A. The regular languages over the alphabet A constitute the smallest collection C of subsets of A^* satisfying that:

1. All finite subsets of A^* belong to C.

2. C is closed under the Kleene start operation (if $M \subseteq A^*$ is inside C, **i.e.** $M \in C$, then $M^* \in C$)
3. C is closed under concatenation (if $M \subseteq A^*, N \subseteq A^*$ satisfy that $M \in C$ and $N \in C$, then $M \circ N \in C$)
4. C is closed under union (if $M \subseteq A^*$ and $N \subseteq A^*$ satisfy that $M \in C$ and $N \in C$, then $M \cup N \in C$)

Definition: Let A be a finite set, and let A^* be the set of words over the alphabet A. A subset L of A^* is called a regular language over the alphabet A is $L = L_m$ for some finite sequence $L_1, L_2, ..., L_m$ of subsets of A^* with the property that $\forall i, 1 \leq i \leq m, L_i$ satisfies one of the following:

1. L_i is a finite set
2. $L_i = L_j^*$ for some $j, 1 \leq j < i$ (the Klenne star operation applied to one of the previous $L_j's$)
3. $L_i = L_j \circ L_k$ for some j, k such that $1 \leq j, k < i$ (L_i is a concatenation of previous $L_j's$)
4. $L_i = L_j \cup L_k$ for some j, k such that $1 \leq k, j < i$ (L_i is a union of previous $L_j's$)

Example 1: Let $A = \{0, 1\}$. Let $L = \{0^m 1^n \mid m, n \in \mathbb{N} \quad m \geq 0, n \geq 0\}$
L is a regular language. Note that L consists of all strings of first 0's, then 1's or the empty string ε. $0^m 1^n$ stands for m 0's followed by n 1's, **i.e.** $0^m \circ 1^n$. Let us examine $L' = \{0^m \mid m \in \mathbb{N}, m \geq 0\}$ and $L'' = \{1^n \mid n \in \mathbb{N}, n \geq 0\}$

Q: Can we obtain them via operatons listed among 1-4?

A: Yes! Let $M = \{0\} \qquad M \subseteq A \subseteq A^*$ and $M^* = L' = \{0^m \mid m \in \mathbb{N} \quad m \geq 0\}$. Let $N = \{1\} \quad N \subseteq A \subseteq A^*$ and $N^* = L'' = \{1^n \mid n \in \mathbb{N}, n \geq 0\}$. In other words, we can do $L_1 = \{0\}, L_2 = \{1\}, L_3 = L_1^*, L_4 = L_2^*, L_5 = L_3 \circ L_4 = L$. Therefore, L is a regular language.

Example 2 Let $A = \{0, 1\}$. Let $L = \{0^m 1^m \mid m \in \mathbb{N}, m \geq 1\}$. L is the language we used as an example earlier. It turns out L is NOT regular. This language consists of strings of 0's followed by an equal number of strings of 1's. For a machine to decide that the string $0^m 1^m$ is inside the language, it must store the number of 1's, as it examines the number of 0's or vice versa. The number of strings of the type $0^m 1^m$ is not finite, however, so a finite-state machine cannot recognise this language.