A context-free grammar $G$ is in *Chomsky Normal Form* (ChNF) if every production has the form $A \rightarrow BC$ or $A \rightarrow c$. Here it is understood that $A, B, C$ are variables and $c$ is a single terminal symbol. Having $G$ in ChNF implies in particular:

- No $\epsilon$-productions $A \rightarrow \epsilon$.
- No unit productions $A \rightarrow B$.
- No mixing of variables and terminals in the RHS of any production.

A Chomsky NF grammar $G$ can never derive its start symbol $S$ to $\epsilon$, so always $\epsilon \notin L(G)$. But we can show that $\epsilon$ is the only string “lost” in this special form:

**Theorem.** Given any CFG $G = (V, \Sigma, P, S)$, we can construct a CFG $G'$ in Chomsky normal form such that $L(G') = L(G) \setminus \{\epsilon\}$.

**Proof.** A step-by-step method for converting any given $G$ into $G'$:

1. First identify the set $NULLABLE = \{ A \in V : A \Rightarrow^* \epsilon \}$. Here is the algorithm to do so:

   $$NULLABLE := \emptyset;$$

   (*) for each $A \in V \setminus NULLABLE$ do

   for each production $A \rightarrow \beta$ do

   if $\beta \in NULLABLE^*$ then $NULLABLE := NULLABLE \cup \{ A \}$;

   end for; end for;

   if (there was no change in $NULLABLE$) then stop; else goto (*);

2. For every production $A \rightarrow \beta$ in $P$, add to $P$ all productions obtained by deleting some subset of occurrences of variables that belong to $NULLABLE$ from $\beta$. Note: If $NULLABLE = \{ A, B \}$ and you have the production $S \rightarrow AS\epsilon BA$, then the 7 added productions include $S \rightarrow ScBA$ and $S \rightarrow AS\epsilon B$, deleting one $A$ but not the other. The other 5 productions you add are $S \rightarrow AS\epsilon A$, $S \rightarrow ScA$, $S \rightarrow ScB$, $S \rightarrow AS\epsilon c$, and $S \rightarrow Sc$.

3. Now delete all $\epsilon$-productions from $P$, including any you may have added in the last step. Call the resulting grammar $G_1 = (V, \Sigma, S, P_1)$.

At this point, we claim that $L(G_1) = L(G) \setminus \{\epsilon\}$. It is pretty clear that adding the productions in step (2) did not add any more strings to the language, because any new production can be simulated by using the old production it came from and then deriving all of the “deleted” variables to $\epsilon$. The issue is whether deleting the productions in step (3) prevents any nonempty strings $x$ from being derived. Well, let $T$ be a derivation tree for $x$ in the original $G$. Now delete all subtrees of $T$ whose yield is $\epsilon$. The resulting tree $T_1$ is a valid derivation tree for $x$ in the new grammar $G_1$. [To study at home. The idea is that the root $B$ of any deleted subtree of $T$ is a $NULLABLE$ variable. Since $B$ is not the root of $T$ (since $x \neq \epsilon$), let $A$ be the parent node of $B$ in $T$, and let $A \rightarrow \beta$ be the production that was used when that $A$ was expanded in $T$. Then $B$ is a $NULLABLE$ variable inside $\beta$, so the new $P_1$ has productions that don’t include that $B$, and one of them gives you how $A$ is expanded in the new tree $T_1$.] That proves the claim. $G_1$ has no more $\epsilon$-productions. Now we move on to the steps for eliminating unit productions from $G_1$.

4. Identify all pairs $A, B \in V$ such that $A \Rightarrow^* B$. Do this by forming a graph $\Gamma$ with variables as nodes and an edge from $A$ to $B$ whenever $A \rightarrow B$ is a production in $P_1$. Then take the transitive closure of $\Gamma$ to form a new graph $\Gamma^*$. 

---
(5) For all pairs $A, B$ such that $A \Rightarrow^* B$ (i.e. for all edges in $\Gamma^*$) and all non-unit productions $B \rightarrow \beta$, add the production $A \rightarrow \beta$. Call the resulting grammar $G_2 = (V, \Sigma, S, P_2)$.

It is fairly easy to see that $L(G_2) = L(G_1)$: any derivation tree $T_1$ of a string $x$ in which some variable node $A$ has only one child $B$ (which is not a leaf) can be edited by splicing out node $B$ and making the children of $B$ the new children of $A$—this corresponds to using the new production $A \rightarrow \beta$ in the new tree $T_2$, which still derives $x$.

At this point, $G_2$ often has a lot of obviously redundant productions that you can delete. Sometimes $G$ has deadwood variables $A$, meaning that no terminal string can be derived from $A$. Any production involving a deadwood variable can be deleted. $G$ may (then) also have unreachable variables $B$, meaning that there is no sentential form $\alpha$ such that $S \Rightarrow \alpha$ and $B$ is part of $\alpha$. Productions involving these variables can be deleted as well. These two steps are optional.

(6) For every terminal symbol $c$, create an “alias variable” $X_c$. For every production $A \rightarrow \beta$ where $\beta$ is not a single terminal symbol, replace every terminal symbol $c$ in $\beta$ by its alias $X_c$. The resulting CFG $G_3$ no longer mixes variables and terminals on the RHS of any production, and needs only one more (rather silly) step for Chomsky NF.

(7) For any production $p = A \rightarrow B_1B_2 \cdots B_m$ in $G_3$, where the $B_i$ are all variables and $m \geq 3$, add new variables $Y_1, \ldots, Y_{m-2}$, and replace $p$ by the productions $A \rightarrow B_1Y_1$, $Y_1 \rightarrow B_2Y_2$, $\ldots$, $Y_{m-2} \rightarrow B_{m-1}B_m$.

Since the variables $Y_1, \ldots, Y_{m-2}$ are used only for this production (make sure you use other variables names for other productions that you simulate in this manner, or superscript $Y_1^p, \ldots, Y_{m-2}^p$ by the production $p$), it is clear that the final resulting grammar $G_4$ gives $L(G_4) = L(G_3) = L(G_2) = L(G_1) = L(G) \setminus \{ \epsilon \}$. And $G_4$ is in Chomsky normal form.

Example. Let $G = \{ S \rightarrow SB|BC, A \rightarrow SAS|a, B \rightarrow CC|BS|b, C \rightarrow DD|AS, D \rightarrow \epsilon|a \}$.

(1) $\text{NULLABLE} = \emptyset$, then $\{ D \}$, then $\{ C, D \}$, then $\{ B, C, D \}$, and finally $\{ S, B, C, D \}$.

(2) Add productions $S \rightarrow B \ [S \rightarrow S \text{ and } S \rightarrow \epsilon \text{ are unnecessary to add}], S \rightarrow C, A \rightarrow AS|SA \ [A \rightarrow A \text{ is unnecessary}], B \rightarrow C|S, \text{ and } C \rightarrow D|A \ [\text{NOT } C \rightarrow S \text{ since } A \text{ is not nullable}].$

(3) Then delete the production $D \rightarrow \epsilon$.

(4) $\Gamma$ has edges $S \rightarrow B, S \rightarrow C, B \rightarrow C|S, \text{ and } C \rightarrow D|A$. $\Gamma^*$ adds $S \rightarrow D|A$ and $B \rightarrow D|A$.

(5) The new CFG has productions $S \rightarrow SB|BC|CC|BS|SAS|AS|SA|b|a|DD, A \rightarrow SAS|AS|SA|a, B \rightarrow CC|BS|b|SB|BC|SAS|AS|SA|a|DD, C \rightarrow DD|a|SAS|AS|SA, \text{ and } D \rightarrow a$. At this point, it is clear that $S$ and $B$ are equivalent variables (since each can derive the other), so we can substitute $S$ for $B$ and delete “$B$” from the grammar.

(6) No productions mix terminals and variables, so no “aliases” are needed.

(7) The only “long” productions are $S \rightarrow SAS$, and $A \rightarrow SAS$, and $C \rightarrow SAS$. Since the RHS is the same in each case, we can use a single “extra variable $Y$” to handle them: $S \rightarrow SY, A \rightarrow SY, C \rightarrow SY, Y \rightarrow AS$.

The final CFG computed by this process has productions

$S \rightarrow SS|SC|CC|SY|AS|SA|DD|b|a,$

$A \rightarrow SY|AS|SA|a, C \rightarrow DD|a|SY|AS|SA, D \rightarrow a, Y \rightarrow AS.$