Normal Forms.

Chomsky Normal Form. All productions are of the form $A \rightarrow BC$ or $A \rightarrow a$ (where $a \in T$ and $A, B, C \in V$).
Useless symbols: Symbols which do not appear in any derivation of a string from the start symbol. That is, the symbol does not appear in any derivation $S \Rightarrow_G^* w$, for any $w \in T^*$.

We want to eliminate useless symbols.

Symbol $A$ is said to be useful if it appears as $S \Rightarrow_G^* \alpha A \beta \Rightarrow_G^* w$, for some $w \in T^*$.

We say that a symbol $A$ is generating if $A \Rightarrow_G^* w$, for some $w \in T^*$.

We say that a symbol $A$ is reachable if $S \Rightarrow_G^* \alpha A \beta$, for some $\alpha, \beta \in (V \cup T)^*$.
Surely a symbol is useful only if it is reachable and generating (though vice-versa need not be the case). What we will show is that if we get rid of non-generating symbols first and then the non-reachable symbols in the remaining grammar, then we will only be left with useful symbols.
Theorem: Suppose $G = (V, T, P, S)$ is a grammar which generates at least one string. Then, if
1) First eliminate all symbols (and productions involving these symbols) which are non-generating. Let this grammar be $G_2 = (V_2, T, P_2, S)$. 
2) Remove all non-reachable symbols (and corresponding productions for them) from the grammar $G_2$. Suppose the resulting grammar is $G_3$. Then $G_3$ contains no useless symbols and generates the same language as $G$. 
Generating Symbols

Base Case: All symbols in $T$ are generating.
Induction: If there is a production of the form $A \rightarrow \alpha$, where $\alpha$ consists only of generating symbols, then $A$ is generating. Iterate the above process until no more symbols can be added.
Reachable symbols

Base Case: $S$ is reachable.
Induction Case: If $A$ is reachable, and $A \rightarrow \alpha$ is a production, then every symbol in $\alpha$ is reachable.

A symbol is non-reachable, iff it is not reachable.
Converting a Grammar into Chomsky Normal Form:
1. Eliminate $\epsilon$ productions.
2. Eliminate unit-productions.
3. Convert the productions to productions of length 2 (involving non-terminals on RHS) or productions of length 1 (involving terminal on RHS).
Eliminating ε productions

1. We first find all nonterminals \( A \) such that \( A \Rightarrow^* \varepsilon \). These nonterminals are called nullable.
2. Then, we get rid of \( \varepsilon \) productions, and for each production \( B \rightarrow \alpha \), we replace it with all possible productions, \( B \rightarrow \alpha' \), where \( \alpha' \) can be formed from \( \alpha \) by possibly deleting some of the nonterminals which are nullable.
   Note: If \( S \) is nullable, then our method only generates the language \( L - \{\varepsilon\} \).
Theorem: If we modify the grammar as above, then 
\[ L(G') = L(G) - \{\epsilon\}. \]

Proof: We prove a more general statement:
For all \( A \in V \), for all \( w \in T^* - \{\epsilon\} \), \( A \Rightarrow^*_G w \), iff \( A \Rightarrow^*_G' w \).

Claim: Suppose \( A \Rightarrow^*_G w \). Then we claim that \( A \Rightarrow^*_G' w \).

Proof:
In the derivation \( A \Rightarrow^*_G w \), “drop” each symbol which eventually produces empty string in the derivation.
Claim: For all $A \in V$, for all $w \in T^* - \{\epsilon\}$, if $A \Rightarrow_{G'}^* w$ then $A \Rightarrow_G^* w$.

Proof: Consider the first step in the derivation:
$A \Rightarrow_{G'}^* \alpha \Rightarrow_{G'}^* w$.

Suppose the corresponding production in $G$ was $A \rightarrow \alpha'$.
Then, we have that $\alpha' \Rightarrow_G^* \alpha$, by having the “nulled” symbols generate $\epsilon$.
Now the claim follows by induction.
Identifying nullable symbols

Base: If $A \rightarrow \epsilon$, then $A$ is nullable.
Induction: If $A \rightarrow \alpha$, and every symbol in $\alpha$ is nullable, then $A$ is nullable.
Apply the induction step until no more nullable symbols can be found.
Eliminating Unit Productions

First determine for each pair of non-terminals $A, B$, if $A \Rightarrow^*_G B$. Then we need to add $A \rightarrow \gamma$, for all non unit productions of the form $B \rightarrow \gamma$.

Base: $(A, A)$ is a unit pair.
Induction: If $(A, B)$ is a unit pair, and $B \rightarrow C$, then $(A, C)$ is a unit pair.
Do the induction step until no more new pairs can be added.
All productions of length $\geq 2$ can be changed to (a set of) productions of length 2 (involving only non-terminals on RHS) or productions of length 1 (involving terminals on RHS) as follows:

Given Production: $A \rightarrow X_1X_2\ldots X_k$

is changed to the following set of productions:

$A \rightarrow Z_1B_2,$

$B_2 \rightarrow Z_2B_3,$ \ldots,

$B_{k-1} \rightarrow Z_{k-1}Z_k,$

$Z_i \rightarrow X_i,$ if $X_i \in T,$

$Z_i = X_i,$ if $X_i$ is a nonterminal,

where $B_i$ (and possibly) $Z_i$ are new non-terminals.
Size of Parse Tree
Theorem: Suppose we have a parse tree using a Chomsky Normal Form Grammar. If the length of the longest path from root to a node is $s$, then size of the string $w$ generated is at most $2^{s-1}$. 
Pumping Lemma

Pumping Lemma for CFL: Let \( L \) be a CFL. Then there exists a constant \( n \) such that, if \( z \) is any string in \( L \) such that \( |z| \geq n \), then we can write \( z = uvwxy \) such that:

1. \(|vwx| \leq n|
2. \( vx \neq \emptyset \)
3. For all \( i \geq 0 \), \( uv^iwx^iy \in L \).
Example: $L = \{a^mb^mc^m : m \geq 1 \}$ is not a CFL. Suppose by way of contradiction that $L$ is a CFL. Then, let $n > 1$ be as in the pumping lemma. Consider $z = a^n b^n c^n$.

Let $z = uvwxy$ be as in the pumping lemma. Now, $|vwx| \leq n$. Thus, $vwx$ cannot contain both $a$ and $c$. In case $vwx$ does not contain an $a$, then $uv^2wx^2y$ contains $n$ a’s, though $|uv^2wx^2y| > 3n$. Thus, $uv^2wx^2y$ is not in $L$.

Similarly, if $vwx$ does not contain a $c$, then $uv^2wx^2y$ contains $n$ c’s, though $|uv^2wx^2y| > 3n$. Thus, $uv^2wx^2y$ is not in $L$.

Thus, in all cases, we have that $L$ does not satisfy the pumping lemma. Hence, $L$ cannot be CFL.
Proof of Pumping Lemma for CFL. Let $L$ be a context free language. Without loss of generality, we assume $L \neq \emptyset$ and $L \neq \{\epsilon\}$. Choose a Chomsky Normal Form grammar $G = (V, T, P, S)$ for $L - \{\epsilon\}$. Let $m = |V|$. Let $n = 2^m$. Suppose a string $z \in L$ of length at least $n = 2^m$ is given. Consider the parse tree for $z$. This parse tree must have a path from the root to a leaf of length at least $m + 1$ (by Theorem proved earlier). Consider the path from the root to a leaf at largest depth. In this path, among the last $m + 1$ non-terminals, there must be two nonterminals which are same (by pigeonhole principle). (See picture: PL-figure)
Then, $z = uvwxy$, where $S \Rightarrow^*_G uAy \Rightarrow^*_G uvAxy \Rightarrow^*_G uvwxy$. Thus, we have $A \Rightarrow^*_G vAx$, $A \Rightarrow^*_G w$.

Thus, $A \Rightarrow^*_G v^iAx^i \Rightarrow^*_G v^ix^i$.

Thus, $S \Rightarrow^*_G uAy \Rightarrow^*_G uv^iAx^iy \Rightarrow^*_G uv^iwx^iy$, for all $i$.

Note that length of $vwx$ is at most $2^m$.

Also, note that $vx \neq \epsilon$, as $A \Rightarrow^*_G vAx$, using 1 or more steps in the derivation, and $G$ is a Chomsky Normal Form grammar (which does not have unit productions or $\epsilon$ productions).
Example: $L = \{\alpha\alpha : \alpha \in \{a, b\}^*\}$ is not a CFL.
Suppose by way of contradiction that $L$ is a CFL.
Then, let $n > 1$ be as in the pumping lemma.
Now consider $z = a^{n+1}b^{n+1}a^{n+1}b^{n+1}$.
Let $z = uvwxy$ be as in the pumping lemma.
Now consider the following cases based on where $v$ and $x$ lie in $a^{n+1}b^{n+1}a^{n+1}b^{n+1}$:
Case 1: $vwx$ is contained in the first $a^{n+1}b^{n+1}$.
In this case, $wxy$ is of the form $a^{n+1-k}b^{n+1-s}a^{n+1}b^{n+1}$, where, $vx = a^k b^s$, and thus $0 < k + s \leq n$.
This string cannot be written as $\alpha \alpha$. Suppose otherwise.
Then, the second $\alpha$ must end with $b^{n+1}$ (as $|\alpha| = \frac{4n+4-k-s}{2} > n$).
Thus, the first $\alpha$ ends somewhere in the first sequence of $b$'s: $b^{n+1-s}$.
Thus, the second $\alpha$ ends with $a^{n+1}b^{n+1}$.
But this means $|\alpha| \geq 2n + 2$, and thus $k + s \leq 0$, a contradiction.
Case 2: $vwx$ is contained in $b^{n+1}a^{n+1}$ part of $z$.
Thus, $wxy$ is of the form $a^{n+1}b^{n+1-k}a^{n+1-s}b^{n+1}$, where,
$vx = b^ka^s$, and thus $0 < k + s \leq n$.
This string cannot be written as $\alpha\alpha$. Suppose otherwise.
Then, $\alpha$ must start with $a^{n+1}$ and end with $b^{n+1}$ (as
$|\alpha| = \frac{4n+4-k-s}{2} > n$).
But then $|\alpha| \geq 2n + 2$, and thus $k + s \leq 0$, a contradiction.
Case 3: $vwx$ is contained in the second $a^{n+1}b^{n+1}$ part of $z$. Thus, $wy$ is of the form $a^{n+1}b^{n+1}a^{n+1-k}b^{n+1-s}$, where, $vx = a^kb^s$, and thus $0 < k + s \leq n$. This string cannot be written as $\alpha\alpha$. Suppose otherwise. Then, $\alpha$ must start with $a^{n+1}$ (as $|\alpha| = \frac{4n+4-k-s}{2} > n$). Thus, the second $\alpha$ starts somewhere in the second sequence of $a$’s: $a^{n+1-k}$. Thus, the first $\alpha$ starts with $a^{n+1}b^{n+1}$. But this means $|\alpha| \geq 2n + 2$, and thus $k + s \leq 0$, a contradiction.
Thus, in all cases, we have that $L$ does not satisfy the pumping lemma. Hence, $L$ cannot be CFL.
Closure Properties:

Substitution:
Consider mapping each terminal $a$ to a CFL $L_a$.
$s(a) = L_a$.
For a string $w$ define $s(w)$ as follows:
$s(\epsilon) = \{\epsilon\}$.
$s(wa) = s(w) \cdot s(a)$, for $a \in \Sigma$, $w \in \Sigma^*$.
That is, $s(a_1a_2\ldots a_n) = s(a_1) \cdot s(a_2) \cdot \ldots \cdot s(a_n)$.

Theorem: Suppose $L$ is CFL over $\Sigma$ and $s$ is a substitution on $\Sigma$ such that $s(a) = L_a$ is CFL, for each $a \in \Sigma$. Then, $\cup_{w \in L} s(w)$ is a CFL.
Let $G = (V, T, P, S)$ be a grammar for $L$. For each $a$, let $G_a = (V_a, T_a, P_a, S_a)$ be a grammar for $L_a$.
Assume without loss of generality that $V_a$’s are pairwise disjoint among themselves as well as with $V$.
Then, $G' = (V', T', P', S)$ is a grammar for $\bigcup_{w \in L} s(w)$, where $V'$ is $V \cup \bigcup_{a \in T} V_a$.
$T'$ is $\bigcup_{a \in T} T_a$.
$P' = P_{\text{new}} \cup \bigcup_{a \in T} P_a$
where $P_{\text{new}}$ is formed using the productions in $P$, where in each of the productions, terminal $a$ is replaced by $S_a$.
Now, $(V', T', P', S)$ is a grammar for $\bigcup_{w \in L} s(w)$. $S \Rightarrow^*_G w$ iff $S \Rightarrow^*_G \alpha$, where $\alpha$ has each symbol $a$ in $w$ replaced by $S_a$. That is, if $w = a_1a_2 \ldots a_n$, then $\alpha = S_{a_1}S_{a_2} \ldots S_{a_n}$. 
Reversal

$L^R = \{w^R : w \in L\}$

If $L$ is CFL, then $L^R$ is CFL.

To see this, suppose $G = (V, T, P, S)$ is a grammar for $L$.

Then, grammar for $L^R$ is obtained by considering

$G^R = (V, T, P^R, S)$, where $P^R$ consists of productions obtained by “reversing” the productions in $P$. That is, $A \rightarrow \alpha$ is a production in $P$ then $A \rightarrow \alpha^R$ is a production in $P^R$, where $\alpha^R$ is the reverse of $\alpha$. 
If $L$ is CFL and $R$ is regular, then $L \cap R$ is CFL.

For this, one can run the PDA for $L$ and DFA for $R$ in parallel. Note that for this, one needs only one stack for the PDA: DFA can be run without using the stack.

Suppose $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a PDA for $L$, and $A = (Q', \Sigma, \delta', q'_0, F')$ is a DFA for $R$.

Then, form PDA $P'' = (Q'', \Sigma, \Gamma, \delta'', q''_0, Z_0, F'')$ as follows:

- $Q'' = Q \times Q'$
- $q''_0 = (q_0, q'_0)$
- $F'' = F \times F'$

For $Z \in \Gamma, p \in Q, q \in Q'$:

$\delta''((p, q), \epsilon, Z) = \delta(p, \epsilon, Z) \times \{q\}$

For $a \in \Sigma, Z \in \Gamma, p \in Q, q \in Q'$:

$\delta''((p, q), a, Z) = \delta(p, a, Z) \times \{\delta'(q, a)\}$. 

Example: $L = \{ w : w \in \{ a, b, c \}^* \text{ and } \#_a(w) = \#_b(w) = \#_c(w) \}$ is not a CFL.

If $L$ were a CFL, then $L \cap a^*b^*c^* = \{ a^n b^n c^n : n \geq 0 \}$ would also be a CFL, contradicting a result proved earlier.
Note that CFLs are not closed under intersection in general:

$L_1 = \{a^nb^nc^m : m, n \geq 1\}$

and

$L_2 = \{a^mb^n c^n : m, n \geq 1\}$

are both context free. However, their intersection

$L_3 = L_1 \cap L_2 = \{a^nb^nc^n : n \geq 1\}$

is not context free.
Testing whether CFL is $\emptyset$ or not.

We can check if $S$ is a useless symbol or not. If $S$ is useless, then the language is $\emptyset$. Otherwise it is non-empty.
Testing membership in a CFL.
CYK algorithm.
Using Chomsky Normal Form.
We use a dynamic programming algorithm.
For $w = a_1 \ldots a_n$, we determine the set $X_{i,j}$ of nonterminals which generate the string $a_ia_{i+1}\ldots a_j$.
Base Case: Note that $X_{i,i}$ is just the set of non-terminals which generate $a_i$.
Induction step: $X_{i,j}$ then contains all $A$ such that $A \rightarrow BC$ and $B \in X_{i,k}$, $C \in X_{k+1,j}$, for $i \leq k < j$. That is, $B$ generates $a_ia_{i+1}\ldots a_k$ and $C$ generates $a_{k+1}\ldots a_j$.
Now, $w = a_1 \ldots a_n$ is in the language iff $X_{1,n}$ contains $S$.
Running Time of the algorithm is $O(n^3)$. 
For $i = 1$ to $n$ do

Let $X_{i,i} = \{A : A \rightarrow a_i\}$.

EndFor

For $s = 1$ to $n - 1$ do

For $i = 1$ to $n - s$ do

Let $j = i + s$.

Let $X_{i,j} = \{A : A \rightarrow BC, B \in X_{i,k}, C \in X_{k+1,j}, i \leq k < j\}$.

EndFor

EndFor

Note that in the above algorithm, $X_{i,k}$ and $X_{k+1,j}$ are already computed by the time $X_{i,j}$ is computed, since $k - i$ and $j - (k + 1)$ are both $< j - i$. 