**Program - 7**

**Aim**

Write a program to remove left recursion from a grammar.

**C Code**

#include <stdio.h>  
#include <string.h>  
  
void removeLeftRecursion(char\* nonTerminal, char\* production) {  
 char alpha[100], beta[100], newProduction[100];  
  
 // Check for left recursion  
 if (nonTerminal[0] == production[0]) {  
 printf("Grammar has left recursion.  
");  
 // Extract alpha and beta  
 strcpy(alpha, production + 1);  
 printf("%s -> %s%s'  
", nonTerminal, beta, nonTerminal);  
 printf("%s' -> %s%s' | ε  
", nonTerminal, alpha, nonTerminal);  
 } else {  
 printf("Grammar does not have left recursion.  
");  
 }  
}  
  
int main() {  
 char nonTerminal[100], production[100];  
  
 // Input the grammar (non-terminal and production rule)  
 printf("Enter Non-Terminal: ");  
 scanf("%s", nonTerminal);  
 printf("Enter Production: ");  
 scanf("%s", production);  
  
 removeLeftRecursion(nonTerminal, production);  
 return 0;  
}

**Output**

Example Input:  
Non-Terminal: A  
Production: Aa|b  
  
Example Output:  
Grammar has left recursion.  
A -> bA'  
A' -> aA' | ε

**Program - 8**

**Aim**

Write a program to remove left factoring from a grammar.

**C Code**

#include <stdio.h>  
#include <string.h>  
  
void removeLeftFactoring(char\* nonTerminal, char\* productions) {  
 char commonPrefix[100], remaining1[100], remaining2[100];  
  
 // Extract common prefix  
 sscanf(productions, "%[^|]", commonPrefix);  
  
 // Assume the productions are in the format of "commonPrefixRest|other"  
 char\* rest = strchr(productions, '|') + 1;  
  
 if (strncmp(commonPrefix, rest, strlen(commonPrefix)) == 0) {  
 strcpy(remaining1, commonPrefix + strlen(commonPrefix));  
 strcpy(remaining2, rest + strlen(commonPrefix));  
  
 printf("Grammar has left factoring.  
");  
 printf("%s -> %s%s'  
", nonTerminal, commonPrefix, nonTerminal);  
 printf("%s' -> %s | %s  
", nonTerminal, remaining1, remaining2);  
 } else {  
 printf("Grammar does not have left factoring.  
");  
 }  
}  
  
int main() {  
 char nonTerminal[100], productions[100];  
  
 // Input the grammar (non-terminal and production rules)  
 printf("Enter Non-Terminal: ");  
 scanf("%s", nonTerminal);  
 printf("Enter Productions: ");  
 scanf("%s", productions);  
  
 removeLeftFactoring(nonTerminal, productions);  
 return 0;  
}

**Output**

Example Input:  
Non-Terminal: A  
Productions: ab|ac  
  
Example Output:  
Grammar has left factoring.  
A -> aA'  
A' -> b | c