Name: AYUSH KUSHWAHA

Reg No.: CH.EN.U4CSE22077

Date: 28-07-2025

-----------------------------------------------------------------------------------  
  **LAB – 3**  
  
**Aim:** To implement eliminate left recursion and left factoring from the given grammar using C program.  
**Code:**  
**Leftrecursion.c**

#include <stdio.h>

#include <string.h>

int main() {

char gram[100], part1[100], part2[100], modifiedGram[100], newGram[100];

int i, j = 0, k = 0, pos = 0;

printf("Enter Production : A->");

gets(gram);

// Split input into part1 and part2 at '|'

for (i = 0; gram[i] != '|' && gram[i] != '\0'; i++, j++)

part1[j] = gram[i];

part1[j] = '\0';

j++; // skip '|'

for (i = j, j = 0; gram[i] != '\0'; i++, j++)

part2[j] = gram[i];

part2[j] = '\0';

// Find longest common prefix

for (i = 0; i < strlen(part1) && i < strlen(part2); i++) {

if (part1[i] == part2[i]) {

modifiedGram[k++] = part1[i];

pos = i + 1; // position after common prefix

} else {

break;

}

}

// Construct newGram with suffixes after the common prefix

for (i = pos, j = 0; part1[i] != '\0'; i++, j++)

newGram[j] = part1[i];

newGram[j++] = '|';

for (i = pos; part2[i] != '\0'; i++, j++)

newGram[j] = part2[i];

newGram[j] = '\0';

// Append 'X' to modifiedGram

modifiedGram[k++] = 'X';

modifiedGram[k] = '\0';

// Print the result

printf("\nA->%s", modifiedGram);

printf("\nX->%s\n", newGram);

return 0;

}

**Output:**

