

**NAME: S Venkata Sai Naveen Kumar (192425176)**

**COURSE NAME : COMPILER DESIGN FOR CODE ANALYSIS**

**COURSE CODE : CSA1423**

**3.DEVELOP A LEXICAL ANALYZER TO TEST WHETHER A GIVEN IDENTIFIER IS VALID OR NOT.**

**C PROGRAMMING CODE:**

#include <stdio.h> #include <ctype.h> int main() { char s[100]; int i, valid = 1; printf("Enter identifier: "); scanf("%s", s);

if (!(isalpha(s[0]) || s[0]=='\_')) valid = 0; for (i=1; s[i]!='\0' && valid; i++) if (!(isalnum(s[i]) || s[i]=='\_')) valid = 0; if (valid) printf("Valid Identifier\n"); else printf("Invalid Identifier\n"); return 0;

}

**OUTPUT:**



**4. IMPLEMENT A C PROGRAM TO ELIMINATE LEFT RECURSION.**

**C PROGRAMMING CODE:**

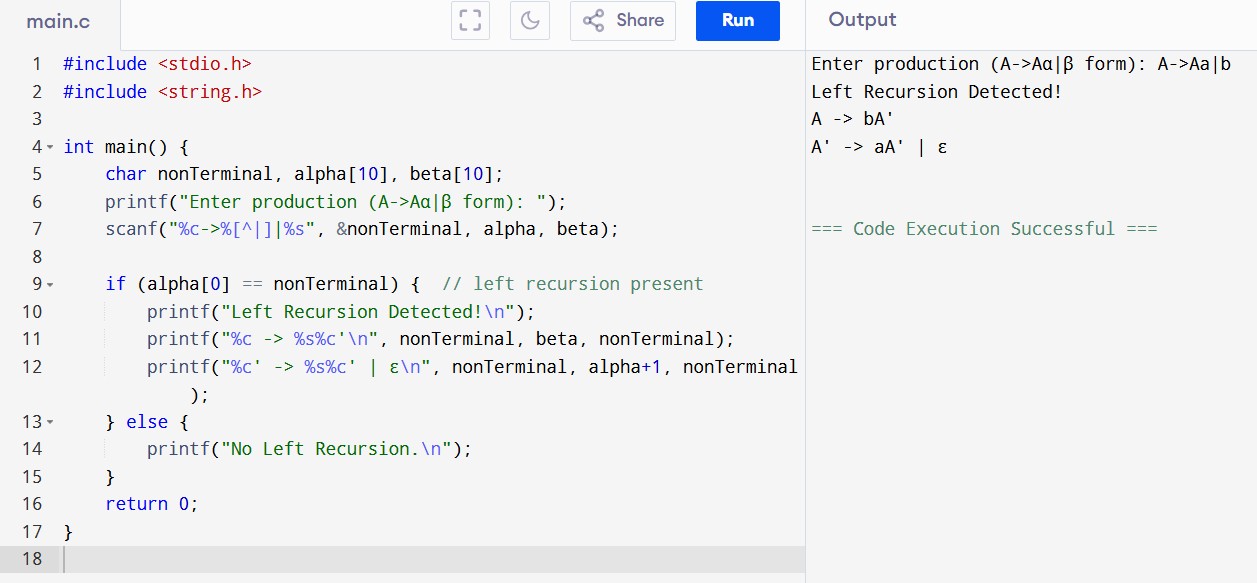
#include <stdio.h> #include <string.h> int main() {

char nonTerminal, alpha[10], beta[10]; printf("Enter production (A->Aα|β form): "); scanf("%c->%[^|]|%s", &nonTerminal, alpha, beta); if (alpha[0] == nonTerminal) { // left recursion present printf("Left Recursion Detected!\n"); printf("%c -> %s%c'\n", nonTerminal, beta, nonTerminal); printf("%c' -> %s%c' | ε\n", nonTerminal, alpha+1, nonTerminal);

} else { printf("No Left Recursion.\n"); } return 0;

}

**OUTPUT:**



**5. IMPLEMENT A C PROGRAM TO ELIMINATE LEFT FACTORING.**

**C PROGRAMMING CODE:**

#include <stdio.h> #include <string.h> int main() { char nt, alpha[10], b1[10], b2[10]; printf("Enter production (A->αβ1|αβ2 form): "); scanf("%c->%[^|]|%s", &nt, b1, b2);

int i=0, j=0;

char common[10]; while(b1[i] && b2[i] && b1[i]==b2[i]) { common[j++] = b1[i]; i++; } common[j] = '\0'; if(strlen(common)==0) { printf("No Left Factoring.\n");

} else { printf("Left Factoring Detected!\n"); printf("%c -> %s%c'\n", nt, common, nt); printf("%c' -> %s | %s\n", nt, b1+i, b2+i);

} return 0;

}

**OUTPUT:**

