Exp No: 7 **Date:**
Name: Pavan Kumar Ch **Reged.no: 22501A05E0**

Experiment - 7

Aim: To implement shift reduce parsing algorithm.

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char ip_sym[15], stack[15];
int ip_ptr = 0, st_ptr = 0, len, i;
char temp[2], temp2[2];
char act[15];
void check();
int main() {
   printf("\n\t\tShift Reduce Parser\n");
   printf("\nGrammar:\n");
   printf("E -> E + E\n");
   printf("E -> E / E\n");
   printf("E -> E * E\n");
   printf("E -> a | b\n");
   printf("\nEnter the input string: ");
   gets(ip_sym);
   printf("\nStack Implementation Table\n");
   printf("Stack\t\tInput Symbol\t\tAction\n");
   printf("------\t\t------------\t\t------\n");
   printf("$\t\t%s$\t\t\t--\n", ip_sym);
   strcpy(act, "Shift ");
   temp[0] = ip_sym[ip_ptr];
   temp[1] = '\0';
   strcat(act, temp);
   len = strlen(ip_sym);
   for (i = 0; i < len; i++) {
      stack[st_ptr] = ip_sym[ip_ptr];
      stack[st_ptr + 1] = '\0';
      ip_sym[ip_ptr] = ' ';
      ip_ptr++;
      printf("$%s\t\t%s$\t\t\t%s\n", stack, ip_sym, act);
      strcpy(act, "Shift ");
      temp[0] = ip_sym[ip_ptr];
      temp[1] = '\0';
      strcat(act, temp);
      check();
      st_ptr++;
   }
   st_ptr++;
   check();
   return 0;
}
void check() {
   int flag = 0;
```

```
    temp2[0] = stack[st_ptr];
    temp2[1] = '\0';
    if ((!strcmp(temp2, "a")) || (!strcmp(temp2, "b"))) {
        stack[st_ptr] = 'E';
        if (!strcmp(temp2, "a"))
            printf("$%s\t\t%s$\t\t\tE -> a\n", stack, ip_sym);
        else
            printf("$%s\t\t%s$\t\t\tE -> b\n", stack, ip_sym);
        flag = 1;
    }
    if ((!strcmp(temp2, "+")) || (!strcmp(temp2, "*")) || (!strcmp(temp2, "/")))
        flag = 1;
    if ((!strcmp(stack, "E+E")) || (!strcmp(stack, "E/E")) || (!strcmp(stack, "E*E"))) {
        strcpy(stack, "E");
        st_ptr = 0;
        if (!strcmp(stack, "E+E"))
            printf("$%s\t\t%s$\t\t\tE -> E + E\n", stack, ip_sym);
        else if (!strcmp(stack, "E/E"))
            printf("$%s\t\t%s$\t\t\tE -> E / E\n", stack, ip_sym);
        else if (!strcmp(stack, "E*E"))
            printf("$%s\t\t%s$\t\t\tE -> E * E\n", stack, ip_sym);
        flag = 1;
    }
    if (!strcmp(stack, "E") && ip_ptr == len) {
        printf("$%s\t\t%s$\t\t\tACCEPT\n", stack, ip_sym);
        exit(0);
    }
    if (flag == 0) {
        printf("$%s\t\t%s$\t\t\tReject\n", stack, ip_sym);
        exit(0);
    }
    return;
}
```

Output:

```
/mnt/c/U/pavan/Doc/G/CompilerDesign/07-shift-reduce 〉main ?2 〉./shiftreduce

               SHIFT REDUCE PARSER

GRAMMER

E->E+E
E->E/E
E->E*E
E->a/b
enter the input string:a+b

       STACK IMPLEMENTATION

stack          input symbol          action
--------       ------------          ------------

$              a+b$                  --
$a              +b$                  shifta
$E              +b$                  E->a
$E+              b$                  shift+
$E+b              $                  shiftb
$E+E              $                  E->b
$E                $                  E->E+E
$E                $                  ACCEPT
```

Result: Developed a shift reduce parser in C language