Experiment-07
Shift Reduce Parser

Date:21/02/2023 **Anas Ahmed Ather (RA2011031010006)**

Aim: Implementation of Shift Reduce Parser

Algorithm:

1. **Start with an empty stack and a sequence of input tokens.**
2. **Read the next token from the input sequence.**
3. **If the token is a terminal symbol (e.g., an identifier or a keyword), push it onto the stack.**
4. **If the token is a non-terminal symbol (e.g., a production rule), apply a reduction operation by popping the appropriate number of symbols from the stack and replacing them with the non-terminal symbol.**
5. **Repeat steps 2-4 until the entire input sequence has been processed.**
6. **If the final symbol on the stack is the start symbol of the grammar, the input sequence is valid and a parse tree can be constructed. Otherwise, the input sequence is invalid and a parse error should be reported.**

Program:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
struct prodn
{
     char p1[10];
     char p2[10];
};
void main()
{
     char input[20],stack[50],temp[50],ch[2],*t1,*t2,*t;
     int i,j,s1,s2,s,count=0;
     struct prodn p[10];
     FILE *fp=fopen("sr_input.txt","r");
     stack[0]='\0';
     clrscr();
     printf("\n Enter the input string\n");
     scanf("%s",&input);
     while(!feof(fp))
     {
```

```
            fscanf(fp,"%s\n",temp);
            t1=strtok(temp,"->");
            t2=strtok(NULL,"->");
            strcpy(p[count].p1,t1);
            strcpy(p[count].p2,t2);
            count++;
      }
      i=0;
      while(1)
      {
            if(i<strlen(input))
            {
                  ch[0]=input[i];
                  ch[1]='\0';
                  i++;
                  strcat(stack,ch);
                  printf("%s\n",stack);
            }
            for(j=0;j<count;j++)
            {
                  t=strstr(stack,p[j].p2);
                  if(t!=NULL)
                  {
                        s1=strlen(stack);
                        s2=strlen(t);
                        s=s1-s2;
                        stack[s]='\0';
                        strcat(stack,p[j].p1);
                        printf("%s\n",stack);
                        j=-1;
                  }
            }
            if(strcmp(stack,"E")==0&&i==strlen(input))
            {
                  printf("\n Accepted");
                  break;
            }
            if(i==strlen(input))
            {
                  printf("\n Not Accepted");
                  break;
            }
      }
      getch();
}
```

Input File: sr_input.txt

```
E->E+E
E->E*E
E->i
```

Output 1:

```
Enter the input string
i*i+i
i
E
E*
E*i
E*E
E
E+
E+i
E+E
E

 Accepted
```

Output 2:

```
Enter the input string
i*+i
i
E
E*
E*+
E*+i
E*+E

 Not Accepted
```

Result: The implementation of Shift Reduce Parser has been successfully executed.